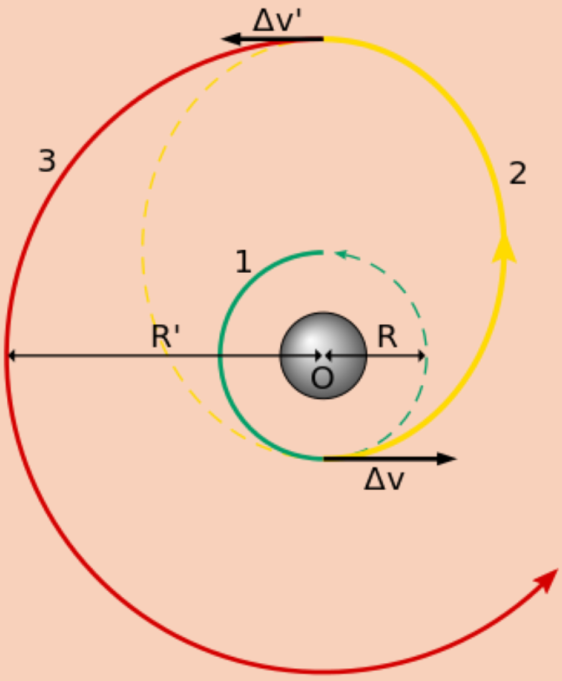
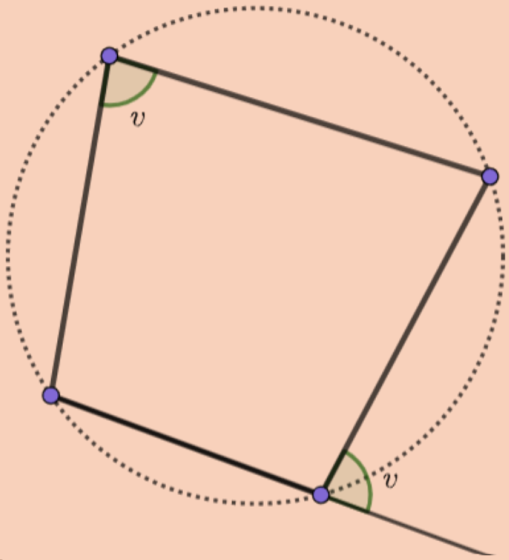
MATTEKÖLLÖ

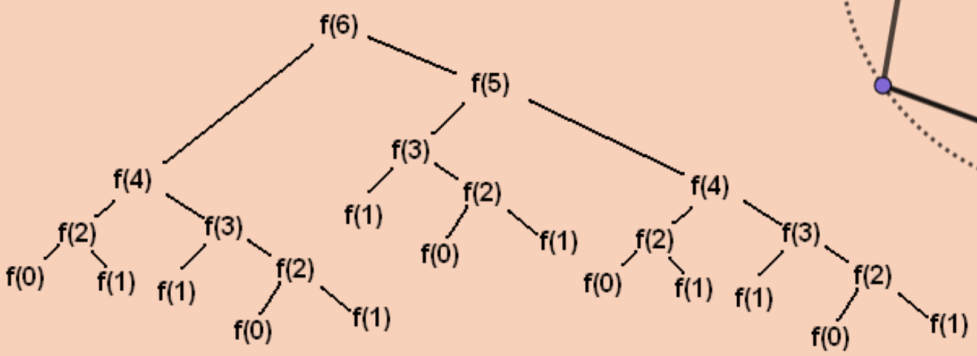
Lektionsmaterial

åk 6–gy2



五 × 五 = 二十五
二 × 九 = 十八
七 + 一 = 八
四 × 九 = 三十六
三 × 六 = 十八
七 + 二 = 九





Mattekollo 2023

Lektionsmaterial

E. Bryland, V. Chapovalova, T. Glimmerfors, K. Haagensen Strömberg, W. Kraft,

L. Lokteva, E. Lundell, F. Löfgren, J. Mårtensson, B. Verbeek, S. Westerlund, H. Zhang

Linköping, juli 2023

Innehåll

VI Programmering - Curie

Matematik - Ung

1. Logik 1

17 juli

Definition. — **Påstående.** Ett påstående i logiken är något som antingen är sant eller falskt.

Definition. — **Negation.** Negationen till ett påstående är falskt när det ursprungliga påståendet är sant och vice versa. Negationen blir alltså ett eget påstående.

Definition. — **Språklig konjunktion.** En språklig konjunktion sätter ihop två påståenden och skapar ett nytt. Några vanliga är:

- "och" - Som är sann om alla påståendena är sanna
- "eller" - Som är sann om något av påståendena är sanna
- "antingen eller" - Som är sann om exakt ett av påståendena är sanna

1. Är dessa påståenden?
   a) Cirklar har inga hörn och kvadrater har 3 hörn
   b) 1, 2, 3, 4 och några andra tal
   c) Antingen är Python ett programmeringsspråk eller så är Java det
2. Är dessa påståenden sanna eller falska?
   a) Man ska vara på sitt rum från 22:00 och frukost serveras 07:00
   b) Mattekollo är inte i september eller oktober
   c) Påståenden är antingen sanna eller falska
3. Skriv ner ett påståenden som du inte vet om det är sant eller falskt.
4. Vad är negationen till "Fredrik älskar alla robotar"?
5. Kan man använda färre "inte"? Påståenden är samma om de alltid är sanna samtidigt.
   a) Det finns inga brädspel som Valentina inte älskar
   b) Inte alla jämna tal delas inte av tre
   c) $x \neq 2$ och $x \geq 2$
6. Kan man förenkla dessa påståenden?

a) x är större än 2 eller mindre än -2 och definitivt större än 3

b) Mattekollo hålls antingen i Linköping eller någon annanstans i Sverige

7. Är negationen till "Idag är tisdag och klockan är efter 8:00" att "Idag är inte tisdag och klockan är inte efter 8:00"?

8. Vad är påståendet som bestämmer om du tror att du kommer gilla ett brädspel? Lägg till minst en av varje konjunktion!

9. — **Extra.** Det finns två till vanliga konjunktioner, "om" och "om och endast om". När borde dessa konjunktioner vara sanna? Vad låter rimligt? Är dessa påståenden sanna?

a) Om ett tal är delbart på 6 är det delbart med 2 och 3

b) Om det är jullov är det inte Mattekollo

c) Om och endast om det regnar ska man använda paraply

10. — **Extra.** Är "Denna mening är falsk" ett påstående?

11. — **Extra.** Om vi har två påståenden, vilka är alla kombinationer av sant och falskt? När är konjunktionen "och" sann?

12. — **Extra.** För att göra logik lättare att skriva kan man använda symboler istället. Då brukar man skriva påståenden som en bokstav, till exempel kan man låta A vara påståendet "Det är söndag". Sedan kan man ersätta "inte" med $\neg$, "och" med $\wedge$ och "eller" med $\vee$. Pröva att skriva några påståenden med dessa symboler.

2. Logik 2

18 juli

Definition. — **Slutledning.** En slutledning innebär att man vet att ett påstående, konsekvensen, är sant om ett annat påstående, villkoret, är sant. Slutledningen i sig är falsk bara om villkoret är sant men inte konsekvensen.

Definition. — **Sanningstabell.** En sanningstabell berättar exakt när ett påstående är sant. Varje rad motsvarar ett fall och varje kolumn ett påstående. De kan till exempel se ut såhär:

"Det regnar"	"William är inne"	"William är inne om det regnar"
F	F	S
F	S	S
S	F	F
S	S	S

1. Varför är slutledningen "Det är sommarlov om det är Mattekollo" sann även under sportlovet?

2. Är slutledningarna garanterat sanna?

   a) Eftersom Julia är en ledare på Mattekollo och Julia tycker om astronomi, finns det en ledare på Mattekollo som gillar astronomi.

   b) Eftersom Harry gillar Mao och Mao är ett kortspel, gillar Harry alla kortspel.

3. Vad är sanningstabellen för "Om det är lektioner eller kvällsaktiviteter så är jag med"?

4. Låt X och Y vara påståenden. Vad blir sanningstabellen för "X om Y"? Vad blir sanningstabellen för "inte X om inte Y".

Notera! Människor talar alltid sanning och varulvar ljuger alltid!

5. Efter en fullmåne under Mattekollo inser du att några av ledarna är varulvar! Du träffar Lisa och Sebastian. Sebastian påstår att de båda är människor, men Lisa påstår att Sebastian är en varulv? Vem är vad, och varför?

6. Du springer vidare och träffar Erik och Tobias. Erik säger att man stoppar ledare från att vara varulvar genom att ge dem kaffe. Tobias påstår att Erik är en varulv om och endast om han själv är det.

   Skapa en sanningstabell med påståendena "Erik är människa" "Tobias är människa" "Eriks påstående stämmer" "Tobias påstående stämmer" "Människor talar alltid sanning, varulvar ljuger alltid" där du testa alla kombinationer av de två första. Kan denna berätta vem som är vad? Är kaffe räddningen?

7. På väg in i köket möter du Harry, Elias, och William. Du tycker dig höra Harry säga "Elias är varulv", Elias säga "William är varulv" och William säga "Harry är varulv". Kan du ha hört rätt?

8. **— Extra.** Du stöter på Julia men vet inte om hon är varulv eller människa. Hon håller två kaffepaket, ett smakar gott och ett illa, men du vet inte vilket som är vilket. Du får ställa en fråga för att lista ut vilket kaffepaket du ska ta. Vilken fråga?

9. Är påståendet "Benjamin är varulv och Valentina är varulv" negationen till "Benjamin är människa eller Valentina är människa"? Två påståenden är samma om de har samma sanningstabell och negationer om de har motsatta.

10. Innan du hinner sätta igång kaffemaskinen kommer de sista Mattekolloledarna fram. Fredrik säger "Alla Mattekolloledare dricker kaffe" medan Kevin säger att "William dricker inte kaffe". Kan båda vara varulvar? Är det rimligt att ställa upp en sanningstabell där man undersöker varje ledares kaffe vanor individuellt?

11. **— Extra Extra.** Några månader senare möter du tre allvetande matematikprofessorer, Sann, Falsk och Slump. Sann talar alltid sanning, Falsk ljuger alltid och vad Slump säger är helt slumpmässigt. Ditt mål är att bestämma vem som är vem genom att ställa ja och nej frågor. Varje fråga måste ges till en specifik proffessor. De förstår svenska men svarar på sitt eget språk där orden för ja och nej är "Bouba" och "Kiki", men det är oklart vilken som är vilken. Hur bestämmer du vem som är vem givet 100 frågor? Hur gör du med bara 3 frågor?

3. Logik 3

19 juli

3.1 Mängdlära

Definition. — Mängd. En mängd är en samling element där ordning och antal inte spelar någon roll. Dessa element kan vara vad som helst till exempel färgen blå eller talet 2.

Man kan definiera en mängd antingen genom att lista upp dess element inom måsvingar, till exempel {flygplan, hatt, pannkaka}, eller genom en regel för elementen, till exempel mängden av alla färger.

Definition. — Mängdoperationer. En mängdoperator tar en eller flera mängder och skapar en ny. Några av de vanligaste är

Operation	Symbol	Innehåller alla element som
Union	$M \cup N$	Är i M eller N
Snitt	$M \cap N$	Är i M och N
Komplement	M^C	Inte är i M

Notera! Komplement fungerar bara om det finns en universalmängd som berättar vad alla relevanta element är, den är oftast intuitiv.

1. Är $\{1,2,3\}$ samma som $\{3,1,1,2\}$?
2. Är $\{\text{Elliot}, \text{Tage}, \text{Tom}\}$ samma som $\{\text{Elliot}, \{\text{Tage}, \{\text{Tom}\}\}\}$?
3. Skriv upp mängden av alla heltal x som uppfyller påståendet "$x > 2$ och $x < 5$".
4. Om universalmängden är {Mattekollo, UVS och Kodsport}, vad blir då:

   a) {Mattekollo, Kodsport} $\cup$ {Kodsport}

   b) {UVS, Mattekollo}^{C}

   c) {Mattekollo, UVS} $\cap$ {UVS, Kodsport} $\cap$ {Kodsport, Mattekollo}

   OBS! UVS = Ung Vetenskapssport.

5. Vi har mängden av alla heltal som är delbara med 7. Vad blir dess komplement? (Universalmängden är alla heltal)
6. Du har mängden av alla tidigare Mattekollodeltagare M och mängden av alla Mattekolloledare N . Hur skapar man mängden av alla som är Mattekolloledare och tidigare varit deltagare?

7. Vi har mängden vars element uppfyller påståendet "Elementet är rött" och den vars element uppfyller påståendet "Elementet är ett klädesplagg". Vilket påstående uppfyller deras snitt?

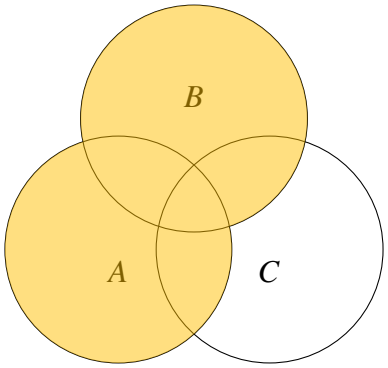
8. Finns det en koppling mellan mängdoperationerna och påståendet deras element uppfyller? Dessa beror kanske särskilt på konjunktionerna?

9. Stämmer det att $(A \cup B) \cap (A \cup C) = A \cup (B \cap C)$ för alla mängder A, B, C? Ställ upp en sanningstabell som för varje element berättar om det är i mängden, givet om det är i A, B, C.

10. — Extra. Det är 9 dagar på Mattekollo, 7 dagar har lektioner, 2 är måndagar, och en är varken måndag eller har lektioner. Hur många måndagar har lektioner?

11. — Extra. Hur många element är i $(A \cup B) \cap C$ om $A =$ mängden av alla tal delbara på 2, $B =$ mängden av alla tal delbara med 5 och $C =$ mängden av alla tal från 0 till 20.

12. — Extra. Ett Venndiagram är ett sätt att illustrera mängder. Man ritar först upp cirklar motsvarande alla mängder, där överlappen representerar element mängderna har gemensamt. Man kan sedan markera de områden en annan mängd består av. Till exempel blir Venndiagrammet för mängden $A \cup B$



a) Rita ett Venndiagram för $(A \cap B) \cup C$

b) Bevisa fråga 9 med Venndiagram

13. — Extra Extra. När man hade lagt upp en omgång med kortspelet Set fanns det 12 kort totalt. 7 var röda, 6 var randiga och 6 hade romber. Vidare vet du att 4 var röda och randiga, 3 hade randiga romber och 2 hade röda romber. Till sist fanns det ett kort med en röd randig romb.

a) Hur många kort var röda och randiga men saknade romber?

b) Hur många hade romber men var varken röda eller randiga?

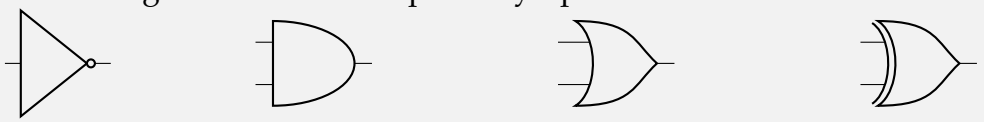
c) Hur många var varken röda, romber eller randiga?

14. — Extra Extra. Kan du se några problem med vår definition av mängder? Kan vi skapa mängder som blir paradoxala (likt påståendet "Denna mening är falsk")? Kan vi ändra definitionen för att skydda oss från detta, kanske med hjälp av en universalmängd?

4. Logik 4

20 juli

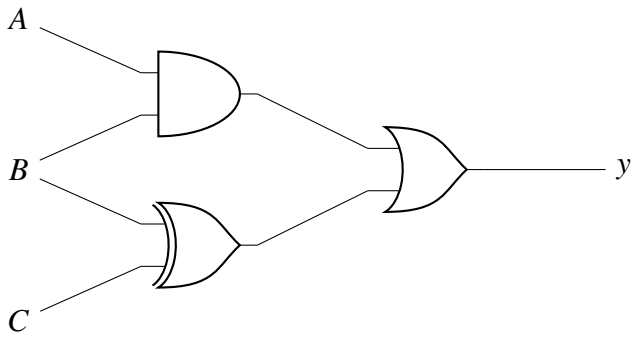
Definition. — Krets. En logisk krets är ett annat sätt skriva påståenden. Man låter några påståenden representeras med variabler, och använder sedan konjunktioner och negationer för att skapa ett nytt påstående. Noderna är



"inte" "och" "eller" "antingen eller"

De kopplas ihop genom att dra en linje från det högra utvärdet på en nod till det vänstra invärdet.

1. Vilka av påståendena A, B, C ska vara sanna för att y ska vara sann?



2. Rita upp en egen krets för påståendet "Antingen A och B eller C".

3. I en masugn måste man bestämma huruvida värmen ska vara antingen på eller av. I en förenklad model säger vi att värmen ska vara på om temperaturen är under 1000° eller om den är inställd på "maximal effekt". Så klart måste värmen vara avstängd om "Nödstopp" knappen är aktiverad. Skapa en logisk krets för detta.

Att bygga en miniräknare

4. Ställ upp additionerna med binära tal:

   a) $1_2 + 11_2$

   b) $101_2 + 1010_2$

   c) $111_2 + 1_2$

5. Vi vill beräkna summan av två ensiffriga binära tal. Vi har påståendena "Ena talet är 1_2" och "Andra talet är 1_2" (istället för 0_2). Hur kan man skriva påståen-

dena "Andra siffran i summan är 1_2" respektive "Första siffran i summan är 1_2". Rita en krets för dessa!

6. Nu vill vi bestämma en godtycklig siffra i det binära talet. Vi vet motsvarande siffror i termerna och har påstående för att dessa är ettor. Vi har också ett påstående för om det finns en "minnesetta" från siffran innan. Rita en krets för om siffran är 1 och om den leder till en "minnesetta".

7. Skapa en symbol för din tidigare krets, en förkortning där bara invärde och utvärde syns men logiken blir samma som tidigare. Sätt ihop dessa till att kunna beräkna summan av två stycken fyra bitars tal.

8. **— Extra.** Hur kan vi bäst representera negativa tal? Fungerar det i vår krets? Kan vi göra det på ett sätt som fungerar i vår krets?

9. **— Extra.** Logiska kretsar kan inte spara någon information, men i datorer vill vi kunna göra detta. Låt oss utvärdera samma krets flera gånger, men låta ett av invärdena påverkas av ett av förra utvärderingens utvärden. Notera att vi fortfarande har vanliga logiska kretsar, men kan ha "minne" genom in- och utvärdena.

För att förtydliga notationen kan vi skriva vårt par av invärde och utvärde som en symbol (kanske en kvadrat?). För att förenkla kan vi göra att invärdet (det värde minnet ger ifrån sig) är samma som förra utvärderingen om utvärdet (det värde minnet fick in) var falskt, och motsatsen om det var sant.

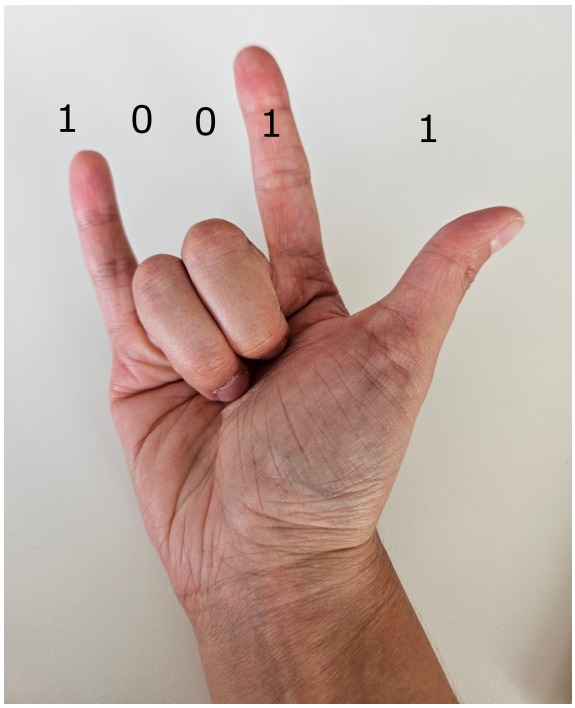
Vad kan vi skapa med detta? Kanske en binär räknare som ökar varje utvärdering? Eller ett stort lager av minne där vi kan välja vilken vi får ut?

5. Talbaser

18 juli

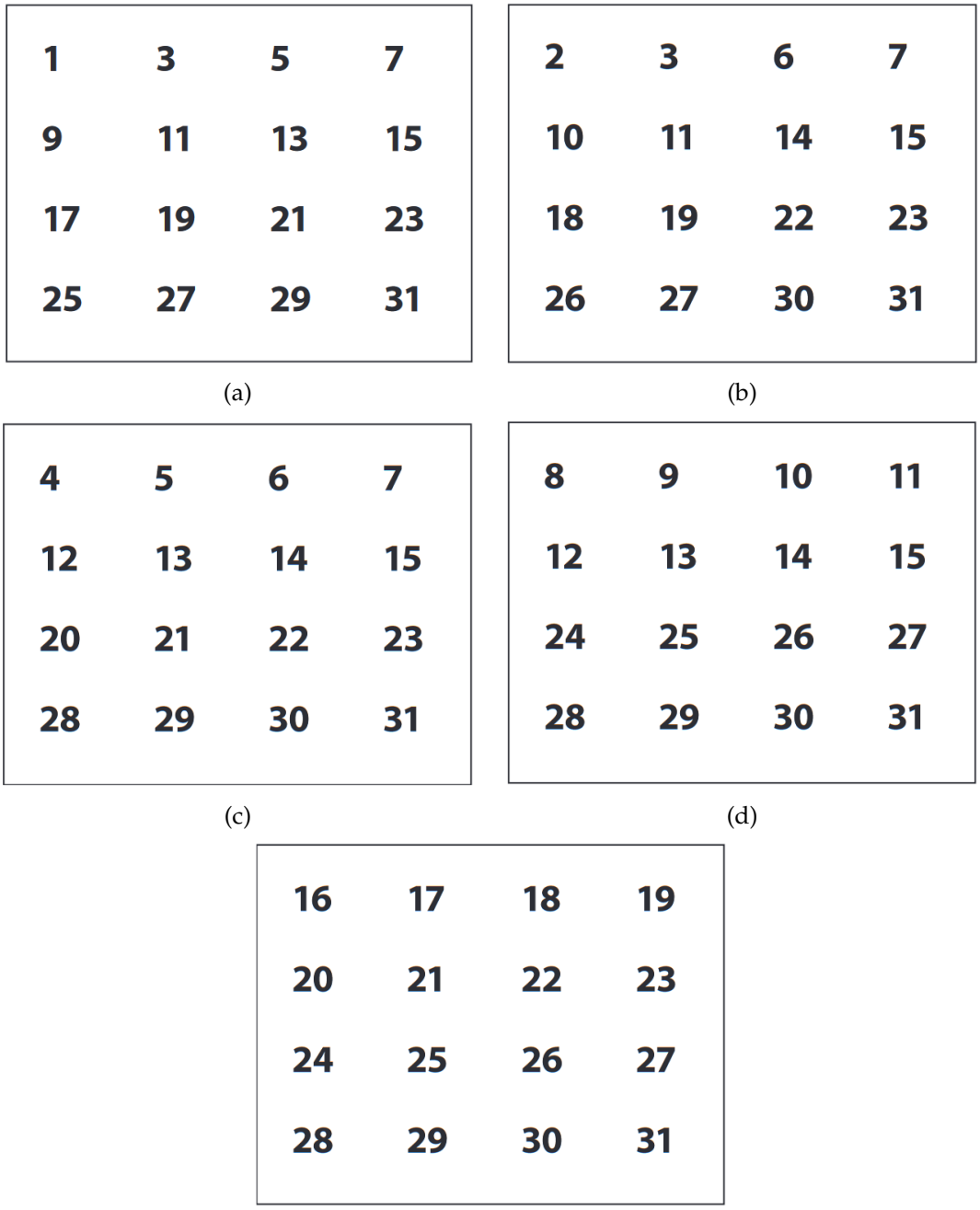
Andra talbaser

1. (Nyckelproblem) Binära tal är praktiska att representera med fingrar. Man representerar en etta med ett utsträckt finger och en nolla med ett böjt finger som i Figur 5.1. Räkna från 1 så långt du kan på fingrarna. Hur stora tal kan man representera på en hand? På två händer?



Figur 5.1: Talet 10011_2 representerat på en hand.

2. (Nyckelproblem) Skriv 25 i baserna 2, 4, 8 och 13.
3. Skriv 2023 i bas 12 och bas 16.
4. (Nyckelproblem) Förklara mitt tankeläsartrick. Tänk på ett tal mellan 1 och 31. Berätta för mig vilka lådor i Figur 5.2 talet ligger i. Hur kan jag snabbt bestämma talet du tänkte på?



Figur 5.2: Tankeläsartrick.

6. Talbaser, operationer och slutsiffror

19 juli

Operationer i andra talbaser

1. Beräkna $239A7_{11} + 3855A_{11}$ helt i bas 11.
2. Beräkna $1001_2 + 1011_2$ helt i bas 2. Svara både i bas 2 och bas 10.
3. Beräkna $1001_2 + 1011_2$ på fingrarna.
4. Fyll i gångertabellen i bas 8.

·	0	1	2	3	4	5	6	7
0								
1								
2								
3								
4								
5								
6								
7								

5. Beräkna $23_8 \cdot 23_8$, $25_8 \cdot 25_8$ och $14_8 \cdot 14_8$ helt i bas 8. Svara i bas 8.
6. Vad är hälften av 1010_2? Vad är hälften av 1011_2?

Slutsiffror

7. Bestäm sista siffran i $5634905734458 + 6432876578$.
8. Bestäm sista siffran i $5634905734458 \cdot 6432876578$.
9. Bestäm sista siffran i 9^{2023}.
10. Bestäm sista siffran i 3^{12}.
11. Bestäm sista siffran i 3^{2023}.
12. Bestäm sista siffran i 2023^{2023}.

Slutsiffror i andra talbaser

13. Bestäm sista siffran i $(A_{11})^{(2023_{11})}$ i bas 11.
14. Bestäm sista siffran i $(2023_8)^{(2023_8)}$ i bas 8.
15. Vilka slutsiffror kan kvadrattal ha i bas 8?
16. Vilken rest har 2023 vid division med 8?
17. Vilken slutsiffra har 2023 i bas 8?
18. Kan 2023 skrivas som summan av två kvadrattal? Tre kvadrattal? Fyra kvadrattal?
19. Kan 20232023 skrivas som summan av tre kvadrattal?
20. Kan 21 skrivas som summan av två kvadrattal?

Extrauppgifter

21. Beräkna $10010/11$. Beräkna $10010_2/11_2$.
22. Visa att $10010_n/11_n$ går jämnt ut för alla heltal $n \geq 2$. Vad blir kvoten?
23. Bestäm sista siffran i $1^4 + 2^4 + 3^4 + \cdots + 2023^4$.

7. Bas 8 och summor av två kvadrater

20 juli

Bas 8 och summor av kvadrater

1. Vilka slutsiffror kan kvadrattal ha i bas 8?
2. Vilka slutsiffror kan en summa av två kvadrattal ha i bas 8?
3. Vilka rester ger talen 230_8, 604_8 och 12_8 vid division med $8 = 10_8$?
4. Förklara sambandet mellan att bestämma sista siffran i ett tal i talbas n och att hitta resten vid division med n. Beskriv sambandet i rutan nedanför.

5. Vilken slutsiffra har 2023 i bas 8?
6. Kan 2023 skrivas som summan av två kvadrattal? Tre kvadrattal? (Fyra kvadrattal?)

8. Moduloräkning

21 juli

Modulonotation

Vi har precis löst ett problem med hjälp av att titta på slutsiffror i en annan talbas. Det kan kännas jobbigt att arbeta i en annan talbas än den man är van vid och framför allt omvandla tal mellan talbaser. Moduloräkning är en sorts språk för att tala om slutsiffror i andra talbaser, som man kan använda när man inte behöver bry sig om de andra siffrorna.

Definition. Om a och b är icke-negativa heltal säger vi att

$$a \equiv b \pmod{n}$$

(uttalas ”a är kongruent med b modulo n”) om och endast om a och b har samma slutsiffra i bas n.

Sats 8.1. Vi har att $a \equiv b \pmod{n}$ om och endast om a och b har samma rest vid division med n.

Exempel 1

- Det är lätt att se att $2023 \equiv 123 \equiv 3 \pmod{10}$
- Modulo 12 (eller 24) använder vi dagligen när vi tittar på klockan. Det är nämligen så att $13 \equiv 1 \pmod{12}$.
- Alla udda tal är kongruenta modulo 2 och alla jämna tal är kongruenta modulo 2.

■

Definition. När vi säger ”Beräkna a modulo n.” menar vi oftast ”Hitta resten av divisionen av a med n.” Vi vill alltså ha ett tal mellan 0 och $n-1$.

Exempel 2 Vi vet ju att $2023 \equiv 1003 \equiv 3 \pmod{10}$, men när uppgiften är ”Beräkna 2023 modulo 10.” så vill vi nog ha svaret 3. ■

1. Beräkna 2340 modulo 10.
2. Beräkna 2023 modulo 8.
3. Beräkna 2100 modulo 11.

Sats 8.2. Antag att $a \equiv b \pmod{n}$ och $c \equiv d \pmod{n}$. Då gäller:

1. $a + c \equiv b + d \pmod{n}$,

2. $a - c \equiv b - d \pmod{n}$,

3. $a \cdot c \equiv b \cdot d \pmod{n}$ och

4. $a^m \equiv b^m \pmod{n}$

OBS Att $m_1 \equiv m_2 \pmod{n}$ betyder inte nödvändigtvis att $a^{m_1} \equiv a^{m_2} \pmod{n}$. Till exempel slutar 3^{12} på 1, men 3^2 slutar på 9. Slutsiffran för 3^m beror inte på ms rest vid division med 10, utan 4.

4. Förklara för din kompis varför Sats 8.2, som ser jätteläskig ut, faktiskt är uppenbart sann och varför den är viktig.

Moduloräkning är jättepraktiskt för att hantera slutsiffror, klockan och delbarhet.

■ **Exempel 3** Säg att vi vill räkna ut slutsiffran i $51234 + 3452 \cdot 134$. Sats 8.2 gör att vi helt enkelt kan skriva

$$51234 + 3452 \cdot 134 \equiv 4 + 2 \cdot 4 \equiv 4 + 8 \equiv 12 \equiv 2 \pmod{10}.$$

Slutsiffran är alltså 2. ■

■ **Exempel 4** Vi bevisar med hjälp av modulospråket att 2023 inte är summan av två kvadrattal.

Varje tal är kongruent med 0, 1, 2, 3, 4, 5, 6, eller 7 modulo 8. Om vi ska bestämma $29^2 \pmod{8}$ behöver vi inte räkna ut 29^2, utan vi kan räkna ut att $29 \equiv 24 + 5 \equiv 5 \pmod{8}$ och $29^2 \equiv 5^2 \pmod{8}$. Vi räknar ut alla kvadraterna modulo 8 och får följande tabell:

x	0	1	2	3	4	5	6	7
x^2	0	1	4	9	16	25	36	49
$x^2 \pmod{8}$	0	1	4	1	0	1	4	1

Ett kvadrattal är alltså alltid kongruent med 0, 1 eller 4 modulo 8. Summor då?

+	0	1	4
0	0	1	4
1	1	2	5
4	4	5	0

Vi ser i tabellen att summan av två kvadrattal endast kan vara kongruent med 0, 1, 2, 4 eller 5. Men

$$2023 = 2000 + 16 + 7 = 2 \cdot 2^3 \cdot 5^3 + 2\dot{8} + 7 = 2 \cdot 8 \cdot 5^5 + 2 \cdot 8 + 7,$$

och

$$2 \cdot 8 \cdot 5^5 + 2 \cdot 8 + 7 \equiv 2 \cdot 0 \cdot 5^3 + 2 \cdot 0 + 7 \equiv 7 \pmod{8}.$$

Eftersom 2023 ger en rest vid division med 8 som summor av två kvadrater inte kan ha, så är inte 2023 summan av två kvadrater. ■

Räkna med modulo

5. Beräkna $30 \cdot 42 + 25 \cdot 5 \pmod 3$.

6. Är 3023_{11} jämnt eller udda? (Är du säker?)

7. Beräkna $1002^3 \pmod{100}$.

8. Beräkna $9^{4000} \pmod 8$.

9. **— Extra.** Beräkna $9^{4000} \pmod{10}$. (Om du kan, använd potensregeln $a^{bc} = (a^b)^c$. Annars kan du använda metoden från Lektion 2.)

▪ **Exempel 5** Säg att klockan är 11 och vi vill veta vad hon kommer vara om 22 timmar. Vi hade kunnat svara 33, men det är inte så vi brukar ange klockan. Klockan anges oftast modulo 12. Vi måste alltså räkna ut 33 modulo 12. Men $33 = 2 \cdot 12 + 9 \equiv 2 \cdot 0 + 9 \equiv 0 + 9 \equiv 9 \pmod{12}$. ▪

10. Säg att klockan är 11. Vad kommer hon vara om 54 timmar?

11. Säg att klockan är 11. Vad var hon för 14 timmar sedan?

12. Vi har definierat modulo för icke-negativa tal. Hur tycker du att man ska definiera modulo för negativa tal? Hur ska vi till exempel tolka $(-1) \pmod{12}$?

13. **— Extra.** Kungen hade guldmynt som vägde 1 g, 3 g, 4 g, 6 g, 8 g, 9 g, 11 g, 12 g respektive 16 g. Robin Hood stal fyra av mynten, och lille John stal också fyra stycken. Det visade sig att Robin Hoods stulna guld vägde 3 gånger så mycket som lille Johns. Vilket mynt fick kungen ha kvar? (Tips: Kolla modulo 4.)

Delbarhet

▪ **Exempel 6** Notera att $a \equiv 0 \pmod n$ om och endast om a är delbart med n. Vi ska visa att summan av tre på varandra följande tal är delbar med 3. Kalla första talet x. Då är andra talet $x+1$ och det tredje $x+2$.

$$x + (x+1) + (x+2) = 3x + 3 \equiv 0 \cdot x + 0 \equiv 0 \pmod 3$$

▪

14. Visa att summan av tre på varandra följande udda tal är delbar med 3.

15. Bevisa att ett tal i bas 10 är delbart med 5 om och endast om slutsiffran är 0 eller 5.

16. Bevisa att $10^n - 1$ är delbart med 9 för alla n.

17. Bevisa att ett (tresiffrigt) tal är delbart med 9 om och endast om dess siffersumma är delbar med 9. (Ledtråd: Skriv talet abc som $a \cdot 100 + b \cdot 10 + c$.)

18. Bevisa att ett (tresiffrigt) tal är delbart med 3 om och endast om dess siffersumma är delbar med 3. (Ledtråd: Skriv talet abc som $a \cdot 100 + b \cdot 10 + c$.)

19. Hur vet man om ett tal är delbart med 4?

20. Hur vet man om ett tal är delbart med 11?

21. Hur vet man om ett tal i binär form är delbart med 3?

22. I vilka baser är det lätt att se om ett tal är delbart med 7?
23. Har bas 10 eller bas 12 bäst delbarhetsregler?

9. Lingvistik I

23 juli

1. **Kinesiska** är ett språk som använder egna tecken för ord. Nedan följer några matematiska uttryck skrivna med kinesiska tecken. Skriv likheterna med arabiska siffror:

1. 五 × 五 = 二十五
2. 二 × 九 = 十八
3. 七 + 一 = 八
4. 四 × 九 = 三十六
5. 三 × 六 = 十八
6. 七 + 二 = 九

2. Skriv med kinesiska tecken:
(a) $1+2=3$
(b) 10
(c) 50
(d) 92

3. Givet att 100 skrivs
一百, skriv med vanliga siffror:
(a) 三百
(b) 六百五
(c) 一百二十三

Mer talteori

5. Hitta alla primtal mindre än 100. Hur många finns det?

6. Avgör om följande tal är primtal:
(a) 91 **(b)** 101 **(c)** 143 **(d)** 347
(e) 12345 **(f)** 387878 **(g)** 1437004797 **(h)** 3599

7. Talen $3,5,7$ är alla primtal. Händer det någonsin igen att tre tal på formen $n, n+2, n+4$ alla är primtal?

8. Hitta det minsta tresiffriga primtalet där varje siffra är ett primtal.

9. Hitta 100 på varandra följande positiva heltal som alla är sammansatta.

10. Visa att det finns oändligt många primtal.

11. **(a)** Visa att $\sqrt{2}$ är irrationellt (dvs. det finns inga heltal a, b sådana att $\sqrt{2} = \frac{a}{b}$).
(b) Visa att alla rötter av heltal n är irrationella då n inte är ett kvadrattal.

10. Lingvistik II

24 juli

10.1 Malagassiska

I följande uppgifter (1-3) ska du med hjälp av logisk slutledningsförmåga och rimliga antaganden försöka lära dig mer om *Malagasiska*, ett språk som talas på Madagaskar av runt 25 miljoner personer.
Detta är en introducerande uppgift för att bekanta dig med en lingvistikuppgifts uppbyggnad. Ställ frågor eller diskutera med personen bredvid dig om du fastnar.

1. Nedan följer en lista med ordgrupper på malagassiska och deras oordnade översättningar. Para ihop rätt ordgrupp med rätt översättning.

1. davenona votsy	a. brun lemur
2. gidro volontany	b. röd hatt
3. gidro votsy	c. röd jord
4. satroka mena	d. vit aska
5. satroka votsy	e. vit hatt
6. tany mena	f. vit lemur

2. Med dina lärdomar från uppgift 1, översätt till svenska:

(a) **gidro mena**

(b) **satroka volontany**

3. Översätt till malagassiska:

(a) röd aska

(b) grå lemur

[Lingolympiaden 2020 U1, *Åke Wettergren*]

10.2 Nen

Nen är ett papuanskt språk som talas i södra Nya Guinea av omkring 250 personer. När vi räknar på svenska använder vi talbas 10. Det betyder att vi har speciella ord för 10 (tio), $10 \cdot 10$ (hundra), $10 \cdot 10 \cdot 10$ (tusen), och så vidare, och tio olika siffror (0-9). Även om väldigt många språk har talbas tio, så gäller det faktiskt inte för alla. Ett sådant exempel är nen.

4. Nedan ges 12 tal skrivna med siffror i talbas 10, samt deras översättningar till nen. Vilken är talbasen i nen?

1	**ambás**
2	**sombés**
3	**nambis**
4	**sombés a sombés**
6	**ambás pus**
7	**ambás pus ambás**
8	**ambás pus sombés**
15	**sombés pus nambis**
30	**widmátandás pus**
34	**widmátandás pus sombés a sombés**
37	**ambás prta ambás**
80	**sombés prta ambás pus sombés**

5. Skriv med vanliga siffror (bas 10):

(a) **widmátandás**

(b) **ambás pus sombés a sombés**

(c) **ambás prta ambás pus ambás**

6. Översätt till nen:

(a) 16

(b) 26

(c) 180

7. Skriv det största tal du vet hur man skriver i nen, och hur stort det är i vanliga siffror.

[Lingolympiaden 2019 U:B, *Amanda Kann & Åke Wettergren*]

10.3 Quechua

Quechua är ett språk som talas av ungefär 9 miljoner personer i och omkring bergskedjan Anderna i Sydamerika. Nedan följer 10 ord på quechua, samt deras översättningar till svenska i en slumpmässig ordning.

1. laqha	a. mörker
2. laqha wasi	b. cykel
3. jatun tata	c. farfar; morfar
4. jatun yachay wasi	d. flygplan
5. kita uywap wasin	e. fängelse
6. kita wuru	f. metallhus
7. lata pisqu	g. pappa
8. lata wasi	h. universitet
9. lata wuru	i. vildåsna
10. tata	j. zoo

8. Para ihop orden på quencha (1-10) med sina svenska motsvarigheter (a-j). *Orden **wasi** och **wasin** betyder samma sak.*

9. Översätt följande ord till svenska:

   (a) pisqu

   (b) yachay wasi

[Lingolympiaden 2019 U:D, *Hampus Lane & Åke Wettergren*]
Sammanställning och anpassning av Benjamin Verbeek

11. Lingvistik III

25 juli

Ni har nu introducerats till lingvistik och har provat på några olika uppgiftstyper. Dagens uppgift kräver alla era nyförvärvade kunskaper inom lingvistik, samt matematisk slutledningsförmåga. Även fonetik kan vara till hjälp här. Arbeta i grupp.

11.1 Cherokesiska

Läs igenom hela uppgiften innan du börjar.
Cherokesiska är ett irokesiskt språk som talas av cirka 2000 personer i de amerikanska delstaterna Oklahoma och North Carolina. Teckenkombinationen **qu** betecknar en konsonant och **v** betecknar en vokal.

Nedan följer ett antal matematiska likheter med tal på cherokesiska. De sex första är skrivna med det latinska alfabetet, medan de sex efterföljande är skrivna med det cherokesiska skriftsystemet. En likhet finns med i båda grupperna.

1. Skriv likheterna med siffror. Ni kan anta att cherokesiska använder bas 10.

1. ***tali*** × ***galiquogi*** = ***nigadu***
2. ***nvgi*** × ***sonela*** = ***tsoisgo sudali***
3. $sonela^{tali}$ = ***tsunelasgo sawo***
4. ***tsunela*** + ***galiquogi*** = ***hisgadu***
5. ***sowo*** + ***tali*** = ***tsoi***
6. ***sudali*** × ***galiquogi*** = ***nvgisgo tali***
7. $\text{O}\cdot\text{Y}^{\text{KT}}$ = ᏔᏓᏢᏍA O·Y
8. KᏚS + ᎯᏍᎩᏍA ᏚᏢᏔᎩ = ᏚᏢᏆᏍAᎯ
9. KT × ᏍAᎯ = KTᏍAᎯ
10. ᎲS + ᏁWS = ᏔᏢᏍA ᎭᏁW
11. ᎯᏍᎩ × ᎭᏁWS = ᎭᏁWᏍA ᎯᏍᎩ
12. O·Y × ᎭᏁW = KTᏍA ᏔᏓᏢ

2. Skriv följande tal med siffror:

(a) ***hisgi*** (b) ***sadu*** (c) ***galiquasgohi*** (d) ***tsoisgohi*** (e) ***tsogadu***

(f) ᏔᎵᏚ (g) ᏦᏞᎵᏍᎪᎯᏐ (h) ᏚᎵᏆᏍᎪ ᎭᏫ (i) ᏚᎵᏆᏕ

3. Skriv föjande tal på cherokesiska (både med det cherokesiska skriftsystemet och med det latinska alfabetet):

(j) 1

(k) 15

(l) 18

(m) 40

(n) 56

(o) 91

Alla tal i problemet är mindre än 100. Observera att det finns två olika varianter av W, se till exempel likhet 10 i uppgift 1, efter likhetstecknet längst till vänster och höger; dessa två tecken är alltså *olika*. Om du använder något av tecknen, se till att det inte går att förväxla med det andra. Om du blir färdig snabbt finns extrauppgifter att få. Ställ frågor om du fastnar.

[Lingolympiaden 2020 U4, *Emil Ingelsten*]
Sammanställning och anpassning av Benjamin Verbeek

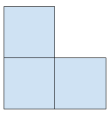
12. Klippgeometri - Rutor

23 juli

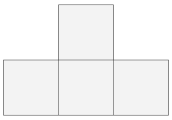
När vi säger dela upp menar vi att hela figuren ska delas upp, ingenting ska bli över och delarna får inte ha någon som helst överlapp.

1. Dela upp ett 8×8 schackbräde där man tagit bort alla hörnrutorna i 1×2-brickor.

2. Dela upp följande figur i trerutiga vinkelhakar:
(a) Ett schackbräde där man tagit bort en hörnruta.
(b) Ett schackbräde där man tagit bort en mittruta.



3. Hur kan man bygga ihop en 60×60-kvadrat utav T-tetraminos (se bild)?

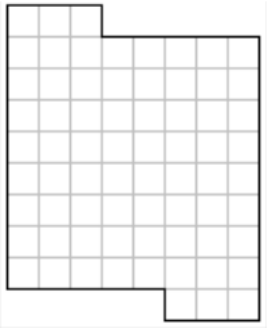


4. Utav en 17×17-kvadrat lämnade bara kvar kanten som var 1 ruta bred. Dela upp ramen i 8 delar och bygg ihop dessa delar till en kvadrat utan hål.

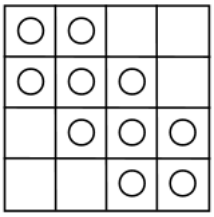
5. Dela upp ett schackbräde utan ett hörn i färre än 10 likadana rektanglar.

6. Dela upp en 5×5-kvadrat längs med rutgränserna i 7 olika rektanglar.

7. På hur många sätt kan figuren på bilden delas upp i 1x5-rektanglar längs med rutgränserna?



8. I en 4×4-kvadrat markerade man 10 rutor (se bild). Dela upp kvadraten i fyra likadana delar så att de innehåller 1, 2, 3 respektive 4 markerade rutor.



9. Kan man alltid dela upp ett 128×128-bräde i trerutiga vinkelhakar om en ruta är borttagen, oavsett vilken?

Extraproblem

10. Dela upp en 1×5-rektangel i fem delar och bygg ihop en kvadrat utav de delarna.

3. Klippgeometri - Godtyckliga figurer

24 juli

När vi säger dela upp menar vi att hela figuren ska delas upp, ingenting ska bli över och delarna får inte ha någon som helst överlapp. Obs: godtycklig betyder vilken som helst (någon annan väljer vilken).

Kongruent betyder exakt likadan (möjligen speglad eller roterad).

1. Kan en godtycklig triangel delas upp
(i) i fyra rätvinkliga trianglar?
(ii) i tre parallelltrapetser?
(iii) i fyra likbenta trianglar?

2. En triangel delades upp i två mindre kongruenta trianglar. Visa att den ursprungliga triangeln var likbent.

3. Dela upp en godtycklig triangel i två delar som sedan kan byggas ihop till ett parallellogram.

4. Rita en figur som kan delas upp i tre kongruenta trianglar men också kan delas upp i fyra kongruenta fyrhörningar.

5. Går det att dela upp en kvadrat i
(i) en åttahörning och 4 trianglar?
(ii) en 16-hörning och 4 trianglar?
(iii) en 34-hörning och 3 tiohörningar?
(iv) en 33-hörning och 3 tiohörningar?
(v) en 35-hörning och 3 tiohörningar?

6. Går det att dela upp en kvadrat i
(i) 4
(ii) 7
(iii) 6
(iv) 8
(v) 3
(vi) 5
mindre kvadrater? Visa hur man gör om det går och bevisa varför det inte går om det inte gör det.

7. Kan en kvadrat delas upp i trianglar på så sätt att varje triangel gränsar till exakt fyra andra?

3. Klippgeometri - Bolyai-Gerwiens sats

25 juli

Definition. Två månghörningar kallas för likbildade om man kan dela en av dem i ett antal delar och bygga ihop den andra av samtliga delarna. Med andra ord kallas det att man kan bygga om en månghörning till en annan. Så klart har likbildade månghörningar lika stora areor.

Sats 3.1. (Wallace-Bolyai-Gerwien). Om två månghörningar har lika stor area så är de likbildade.

1. Visa att om månghörningarna P och Q är likbildade och månghörningarna Q och R är likbildade så är även månghörningarna P och R likbildade.

2. En konvex månghörning är en månghörning där alla vinklarna är under $180°$. Visa att en godtycklig månghörning (även icke-konvex) kan delas in i ett antal konvexa månghörningar.

3. Visa att en godtycklig konvex månghörning kan delas in i ett antal trianglar.

4. Visa att en godtycklig triangel kan byggas om till en rektangel.

5. Visa att en godtycklig rektangel kan byggas om till en rektangel där ena sidans längd är mellan 1 cm och 2 cm.

6. Låt $a \leq b \leq 2a$. Visa att en godtycklig rektangel där en sida har längden b kan delas in i högst tre delar och byggas om till en rektangel där en sida har längden a utav de delarna.

7. Visa att en godtycklig månghörning kan byggas om till en rektangel där en sida har längden 1 cm.

8. Visa att om två månghörningar har lika stor area så är de likbildade.

9. Hitta på ett sätt att bygga om en rektangel av storleken 5Œ1 till en kvadrat.

10. Dela rektangeln av storleken 3 Œ 1 i högst sex delar och bygg en kvadrat av de delarna.

11. Dela rektangeln av storleken 3 Œ 4 i tre delar och bygg en kvadrat av de delarna!

12. Bygg om en kvadrat till tre lika stora kvadrater. Du får skära den ursprungliga kvadraten i högst
(i) 10 delar.
(ii) 7 delar.

Matematik - Medel

1. Satelliter: Kinematik

18 juli

I denna lektionsserie kommer vi att studera matematiken och fysiken bakom satelliter och rymdfärd. Det kommer att börja med några klassiska kinematikberäkningar för att bekanta oss med relevanta ekvationer, och i slutet av kursen kommer ni ha designat er egen satellitkonstellation.

Nedan följer några användbara fysikaliska lagar. I uppgifterna som följer kan ni i allmänhet försumma luftmotstånd. Ni kan även räkna med att gravitationsaccelerationen $g = 10\ \mathrm{m/s^2}$.

Sats 1.1. **Energins bevarande**
Kinetisk energi:

$$E_K = \frac{mv^2}{2} \tag{1.1}$$

där m är objektets massa och v dess hastighet.
Potentiell energi:

$$E_P = mgh \tag{1.2}$$

där g är gravitationsaccelerationen och h är höjden (relativt någon referenspunkt).
För ett slutet system gäller att

$$\sum E = \text{konstant} \tag{1.3}$$

det vill säga att energin bevaras.

Sats 1.2. **Newton II:a lag**

$$F = ma$$

med F kraft, m massa, a acceleration.

Sats 1.3. **Rörelsemängdens bevarande**
Rörelsemängden p för en kropp definieras som

$$p = mv$$

med m massa, v hastighet.

För en stöt av två kroppar är den totala rörelsemängden konstant

$$m_1 v_{1i} + m_2 v_{2i} = m_1 v_{1f} + m_2 v_{2f}$$

i står för *initial* och f för *final*.

Sats 1.4. Kraft över en sträcka (arbete)
Kraft som utförs över en sträcka kräver en viss mängd energi för att utföras. Detta kan beskrivas som

$$E = Fs$$

där F är kraften och s är sträckan. I fallet då F inte är konstant under sträckan så kan man istället integrera kraften över sträckan. Man får då

$$E = \int_a^b F(s)\mathrm{d}s \tag{1.4}$$

där a och b är sträckans start respektive slutpunkt.

Uppgifter

* = *svårare, spara om du vill*

1. Från 10 meters höjd släpps en bowlingkula och en fjäder. Vilken når marken först om de släpps i vakuum (utan luft)?

2. Från 1 meter ovanför marken skjuter Erik en gevärskula horisontellt med hastigheten 300 meter per sekund. Hur långt kommer kulan?

3. Julia är hemlig agent och springer mot maffian Maffiosos flyktbil. Hon behöver sätta fast en GPS-spårare på den, men bilen börjar köra iväg när hon är 9 meter bort. Hon springer med konstant hastighet för att hinna ikapp. Bilen accelererar med konstant acceleration 2 $\mathrm{m/s^2}$. Hur snabbt måste Julia minst springa för att kunna sätta fast spåraren? Hon kan inte kasta eftersom spåraren är ömtålig.

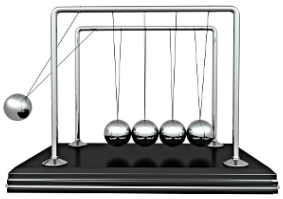
4. * Erik vill nu skjuta på en kokosnöt uppe i ett träd, men han vet att så fort han skjuter så kommer kokosnöten att falla mot marken. Var ska han sikta för att pricka kokosnöten i luften?

5. Valentina spelar biljard och skjuter den vita bollen rakt på 8an med hastigheten 8 m/s. Vilken hastighet får 8an efter stöten? Energin bevaras i stöten.

6. Tobias, som alltid vill vinna, fuskar och spelar biljard med en specialboll. Bollen han skjuter med väger dubbelt så mycket som de vanliga kulorna, eftersom han tänker att han då skjuter med dubbelt så mycket energi för samma hastighet. Stämmer det? Vilken hastighet har fuskbollen efter en rak kollision med en stillastående vanlig boll om den börjar med hastigheten v? Vad kommer hända om han skjuter på en vanlig boll rakt mot ett hål? ***Extra:*** *vad händer om Tobias försöker sabotera för Valentina och istället gör fuskbollen hälften så tung som en vanlig boll?*

7. Benjamin står på ett tågspår och kastar en tennisboll mot ett tåg. Tennisbollen har en hastighet $v_{1i} = 10$ m/s när Benjamin kastar den *(gör inte det)*. Tåget rör sig mot Benjamin med 50 m/s. Hur fort rör sig bollen mot Benjamin efter krocken med tåget?

8. *Newtons vagga* består av ett antal stålkulor upphängda i tunna trådar. När en kula dras ut på änden så studsar en likadan kula ut på andra sidan med samma hastighet, och så studsar de fram och tillbaka så. Men om man drar ut två kulor så studsar det alltid ut två kulor, och inte t.ex. en kula med högre hastighet. Hur ”vet” kulorna att det ska studsa ut lika många kulor varje gång?

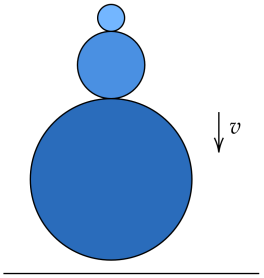


Sats 1.5. Studskoefficient
En boll som släpps mot marken från en höjd h kommer att nå upp till höjd kh efter studsen om $e = \sqrt{k}$ är studskoefficienten. $0 \leq e \leq 1$ gäller alltid.

9. Vad är studskoefficienten k om energin i studsen bevaras?

10. En boll med studskoefficient $k = 0.9$ släpps från höjden 1 meter. Efter hur lång *tid* ligger den helt stilla på marken?

11. Erik har hört att om man staplar bollar på varandra och sedan släpper dem så kan man få den översta bollen att studsa mycket högre än höjden bollarna släpptes ifrån.



(a) Betrakta två bollar: en lätt ovanpå en mycket tyngre. De släpps från en höjd h. Hur högt kommer den lätta bollen att nå efter fallet?

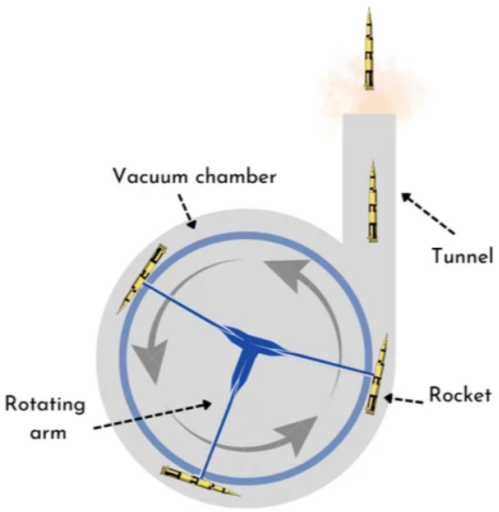
(b)* Kan man ta någon konfiguration av bollar så att den översta lämnar jordens atmosfär för alltid? I teorin? Vad sätter käppar i hjulet i praktiken?

Sats 1.6. Kriterium för cirkelrörelse (centripetalkraft)

$$F_c = \frac{mv^2}{r}$$

med F_c centripetalkraft, m massa, v hastighet (engelska: *velocity*) och r cirkelbanans radie. Kraften riktas vinkelrätt mot rörelseriktningen.

12. Ett amerikanskt företag, *spin launch*, försöker kasta upp en projektil i omloppsbana enbart genom att snurra den väldigt snabbt på marken. Vi antar att armen är 10 meter lång.



(a) Om vi låtsas att luftmotståndet inte finns, hur fort skulle armen behöva snurra för att projektilen ska nå upp till 1 km höjd?
(b) Vad blir kraften på den roterande armen?

Benjamin Verbeek

2. Satelliter: Omloppsbanors kinematik

19 juli

Idag diskuterar vi lagarna som styr planeters rörelser, samt mer av strategierna vi diskuterade igår fast i ett mer rymdigt sammanhang. Fortsätt gärna även lösa problemen från pass 1.

Sats 2.1. **Keplers 1:a lag**
Planeterna rör sig i elliptiska banor med solen i ena brännpunkten

Uppgifter

1. Utan att slå upp värdena på gravitationskonstanten G och M_{jord}, bestäm faktorn GM/R^2 i Newtons gravitationslag $F_g = \dfrac{GMm}{r^2}$.

2. Beräkna flykthastigheten (*escape velocity* på engelska) från jorden. Jordens radie är $R_{jord} = 6371$ km.

3. Vad är den kinetsika engergin $T = \frac{1}{2}mv^2$ som satelliten har i en viss (cirkulär) omloppsbana, uttryckt i G, M, m och r? Vad är den potentiella energin U? Den totala energin E?

Det visar sig att resultatet ovan går att generalisera till elliptiska omloppsbanor, genom att byta ut r mot a: ellipsens halva storaxel (*semi-major axis* på engelska).

4. Vad händer om energin i ditt svar på problem 3 blir positiv?

5. Med vilken hastighet rör sig en satellit runt jorden i en cirkulär omloppsbana på en given höjd h?

6. Beräkna omloppstiden T för en satellit i cirkulär omloppsbana vid höjden $h = 600$ km.

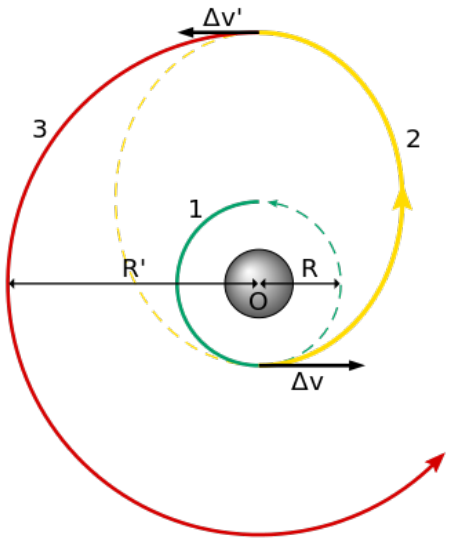
Benjamin Verbeek

3. Satelliter: Avancerade manövrar

20 juli

Uppgifter

1. **Hohmannövergång.** Denna övergång beskriver den minsta mängd Δv som en farkost behöver för att byta omloppsbana från en cirkulär bana till en annan. Beräkna mängden Δv (vilket är hur energiskillnad beskrivs vid rymdfärd) som behövs för att byta omloppsbana från 200 km höjd till 400 km höjd över jorden.



2. **Gravitationsslunga.** Hur fungerar en gravitationsslunga? Beräkna Δv som en farkost får givet att den inkommer vinkelrätt mot en planets rörelse i planetens referenssystem. Planeten har hastigheten v_p relativt solen, och farkosten har initialt hastigheten v_0 i planetens referenssystem.

Benjamin Verbeek

4. Satelliter: Designa ett rymduppdrag

21 juli

Under detta pass ska vi designa en satellitkonstellation av CubeSats runt jorden. Satelliterna har som mål att detektera skogsbränder på jorden, vilket de gör genom att ta infraröda bilder som sedan analyseras. Du tillhör *Mission Analysis*-gruppen; det är ni som ska bestämma omloppsbanan och planera eventuella manövreringar i rymden. Nedan följer ett utdrag av uppdragskriterierna som presenterats från projektledaren:

- **MIS-MA-010** Satelliterna skall fasas ut i konstellationen inom 3 månader.
- **MIS-MA-020** Konstellationen skall ta bild på hela jorden var 12:e timme.
- **MIS-MA-030** Satelliterna skall falla tillbaka in i atmosfären inom 25 år.
- **MIS-MA-040** Omloppsbanorna skall befinna sig i ett och samma plan.

För att tillfredsställa **MIS-MA-030** kan du anta att omloppsbanans altitud är som högst 600 km.

Uppgifter

1. Enligt ovan så får altituden inte överstiga 600 km. Vi vill dock placera oss så högt som vi får; kan du förklara varför? Det kan finnas flera anledningar.
2. Välj en lämplig omloppsbana för konstellationen. Varför är just denna omloppsbana bra? *T.ex. GEO, LEO, LEO/SSO, polär omloppsbana...*
3. Med vilken hastighet rör sig satelliterna runt jorden i den valda omloppsbanan?
4. Beräkna omloppstiden T för satelliterna.
5. Om vi antar att satelliterna kan se ända till horisonten, vad blir deras *field of view, FOV* (hur långt de kan se på jordens yta)?
6. Hur många satelliter behövs som minst i konstellationen för att uppfylla **MIS-MA-020** (och övriga krav)? Antag vi följer en SSO. Kameran ombord kan dessvärre enbart se $\frac{1}{10}$ av FOV.
7. Vid uppskjuttning så placerar raketen samtliga satelliter i ungefär en och samma punkt. För att fördela satelliterna i konstellationens positioner behöver de fasas ut. Om vi antar att vi kan flytta oss till en annan altitud och tillbaka på försumbart kort tid, vad är den minsta höjdförändringen som behöver göras för att tillfredställa **MIS-MA-010**?

8. * När det kommer till att tillfredsställa rymdskrotsriktlinjerna (som sätter kravet **MIS-MA-030**) så är 600 km mest en uppskattning. En viktig faktor som påverkar luftmotståndet är hur stor area satelliten möter atmosfären med. Om vi antar att vi har en rätblocksformad satellit med total ytarea S (från engelskans *surface area*) som roterar slumpmässigt (i värsta fall är satelliten *"dead on arrival"*); vad är den genomsnittliga arean A_{avr} som möter atmosfären?

9. **Extra: be om extra förklaring om du kommer hit och behöver det.* Beräkna den största mängden Δv som behövs för att genomföra utfasningen.

Benjamin Verbeek

5. Konstruktioner

Juli 2023

De gamla grekerna utforskade vilka geometriska konstruktioner som var möjliga med passare och ograderad linjal. Nu är det er tur att gå i deras fotspår. Nedan följer "regler" som gäller för geometriska konstruktioner.

- En linje ritas genom 2 punkter
- Cirklar ritas med att välja mittpunkt A, och en punkt B så att radien på cirkeln blir AB
- Egenskaperna måste vara matematiskt exakta. Exempelvis kan du inte själv gissa mittpunkten på en sträcka, utan det måste i teorin vara exakt.
- Du kan markera punkter godtyckligt, t.ex. på en linje eller cirkel, men inte med mer specifika egenskaper
- Få fram nya punkter genom skärningspunkter av linjer och cirklar.

5.1 Uppvärmning

1. Konstruera en liksidig triangel

5.2 Mittpunktsnormaler

2. Konstruera mittpunktsnormalen av sträckan AB
3. Hitta mittpunkten mellan A och B
4. Rita en godtycklig triangel. Konstruera skärningspunkten av två av sidornas mittpunktsnormaler.
5. En median i en triangel är en linje från ett hörn till motstående sidas mittpunkt. Konstruera skärningen av två av triangelns medianer

5.3 Vinkelräta linjer

6. Givet en punkt A på en linje l, skapa linjen genom A vinkelrät mot l
7. Givet punkt B utanför linje l, skapa linjen genom B vinkelrät mot l
8. Rita en godtycklig triangel. Konstruera skärningspunkten av två av sidornas höjder

5.4 Vinklar och Bisektriser

9. Skapa bisektrisen till en godtycklig vinkel.
10. Skapa en vinkel dubbelt så stor som en given vinkel.
11. Skapa en vinkel på 60 grader.
12. Skapa vinkel på 30 grader.
13. Skapa en vinkel på 45 grader.
14. Skapa en vinkel på 75 grader.

5.5 Parallella linjer

15. Givet en punkt A utanför linje l, skapa linjen genom A parallell med l.
16. Konstruera en parallellogram (med ickeräta vinklar)

6. Konstruktioner 2

Juli 2023

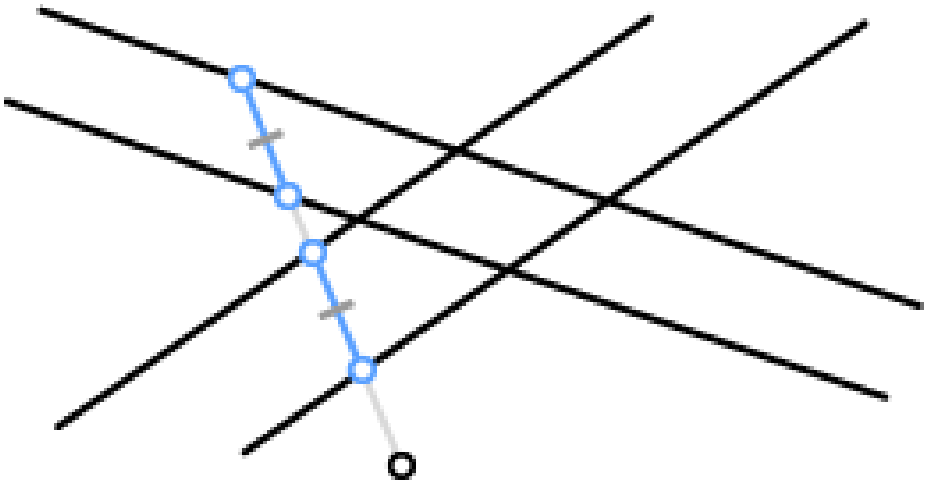
1. Du har en godtycklig triangel. Konstruera en linje genom ett av hörnen så att triangeln delas in i två mindre trianglar med lika area.

2. Samma situation som ovan men de mindre trianglarna ska istället ha lika omkrets.

3. Du har en cirkel. Rita en cirkel, koncentrisk med den första, så att den första cirkeln delas in i två delar med lika area.

4. Två kvadrater av olika storlek delar ett av sina hörn. Konstruera en tredje kvadrat, med samma gemensamma hörn, så att dessa area är lika med summan av de två mindre kvadraternas areor.

5. Fyra linjer, l_1, l_2, l_3 och l_4. l_1 och l_2 är parallella, l_3 och l_4 är parallella. Du har en punkt A utanför alla linjer. Rita en linje genom A så att den skär l_1, l_2, l_3 och l_4 i P_1, P_2, P_3 respektive P_4 och segmenten $P_1P_2 = P_3P_4$.

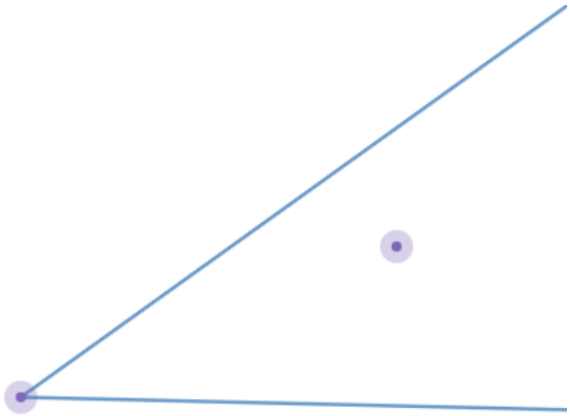


6. Givet en vinkel på 54 grader, dela den i 3 lika stora vinklar.

7. Givet en kvadrat, konstruera en regelbunden oktagon som delar fyra av sina sidor med kvadraten.

8. Välj ut fyra slumpmässiga punkter. Rita en cirkel, med två av punkterna innanför och två utanför cirkeln, så att det kortaste avståndet från varje punkt till cirkeln är lika långt för alla fyra punkter.

Följande problem kommer alla utgå från följande figur:



9. Du ska rita en linje genom punkten så att:

(a) Den triangel som bildas är likbent

(b) Punkten är mittpunkten på linjen

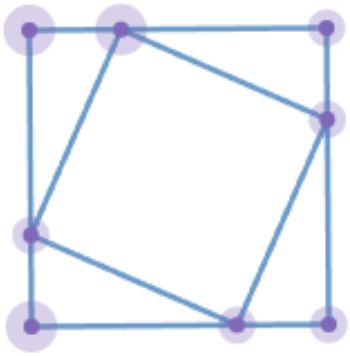
10. Rita en linje så punkten är skärningspunkten av höjderna till triangeln som bildas av linjen och de två redan givna linjerna.

7. Konstruktionsgolf

Juli 2023

Varje given konstruktion kan göras på många olika sätt. Vissa mer effektiva än andra. Till de här problemen är er uppgift att utföra konstruktionerna på så få drag som möjligt! Ett drag är att rita en cirkel eller att rita en linje mellan två punkter. Markering av punkter kostar inget drag.

1. Konstruera en kvadrat.
2. Konstruera en regelbunden Hexagon.
3. Konstruera en regelbunden Oktagon.
4. Givet en sträcka AB och en punkt C. Rita cirkeln med centrum C och radie AB (definitionen av en stel/modern kompass).
5. Givet en kvadrat och en punkt A på kvadraten, konstruera en kvadrat med A och alla dess punkter på den större kvadraten



6. Dela en sträcka i 3 lika stora delar
7. Givet punkt A utanför linje l, skapa linjen genom A vinkelrät mot l. (3)
8. Givet en punkt B på en linje l, skapa linjen genom B vinkelrät mot l. (3*)
9. Givet två parallella linjer, konstruera linjen mittemellan dessa som också är parallell. (5*)

8. Konstruktioner - Konstruerbara tal

Juli 2023

Om man börjar med en given enhetslängd, vilka längder är möjliga att konstruera med passare och ograderad linjal? Du får använda de verktyg vi använt: mittpunktsnormaler, vinkelräta linjer, parallella linjer, bisektriser och återanvändning av längder (stel kompass).

8.1 $\mathbb{N}$

1. **(a)** Konstruera längden 2

   (b) Konstruera längden 3

   (c) Givet n, kan du konstruera $n+1$? Vad innebär det isåfall?

8.2 +

2. **(a)** Givet en sträcka med längd 3 och ett med 5, konstruera längden 8

   (b) Givet sträckor med längderna a och b, hur konstruerar man $a+b$?

8.3 -

3. **(a)** Givet sträckor med längd 10 respektive 4, konstruera längden 6

   (b) Givet sträckor med längderna a respektive b $(a > b)$, konstruera längden $a-b$

8.4 x

4. Kan du givet sträckor med längderna a och b, skapa längden $a \times b$?

8.5 ÷

5. Givet sträckor med längderna 3 och 5:

   (a) Konstruera $3/5$

   (b) Konstruera $5/3$

6. Givet sträckor med längderna 0.3 och 0.5:

   (a) Konstruera $0.3/0.5$

   (b) Konstruera $0.5/0.3$

   Om vi kan konstruera $\frac{a}{b}$, vilken mängd är då konstruerbar?

8.6 $\sqrt{}$

7. **(a)** Konstruera $\sqrt{2}$

   (b) Konstruera $\sqrt{3}$

   (c) Givet $\sqrt{n}$, konstruera $\sqrt{n+1}$

   (d) Givet x (inte nödvändigtvis heltal) konstruera $\sqrt{x}$ (*)

8.7 Övningstal

8. Givet en kvadrat med sidlängd 1, konstruera följande:

   (a) $\sqrt{2}+1$

   (b) $\frac{\sqrt{2}-1}{2}$

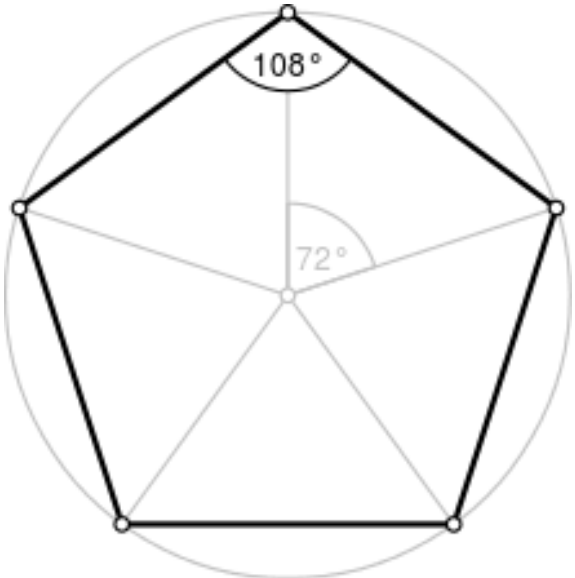
   (c) $\frac{1}{3}$

   (d) $\sqrt{5}$

   (e) $\frac{\sqrt{5}-1}{4}$

8.8 Pentagon

Genom att veta att $\cos 72 = \frac{\sqrt{5}-1}{4}$ går det att konstruera en regelbunden pentagon inskriven i en cirkel. Försök att komma på en egen lösning



9. Kryptografi I: Kongruensräkning

23 juli

9.1 Heltalsräkning

Kongruensräkning bygger på heltalsräkning, så för att förstå kongruensräkning behöver vi börja med heltal (både positiva och negativa). Heltal kan adderas, subtraheras och multipliceras med varandra. Heltal kan i allmänhet inte divideras då vissa kvoter inte är heltal.

Definition. Delbarhet. Låt a, b och c vara heltal. Vi säger att a delas av b, och att b är en delare till a, om $a = bc$. Att b delar a skriver vi som $b|a$.

■ **Exempel 7** 7 delar 35, vilket kan skrivas som $7|35$. ■

Om $a = bc$ är $a = (-b)(-c)$. Negationen av en delare är också en delare med negerad kvot. Detta gör negativa tal tråkiga i sammahanget. Vi kan i princip strunta i negativa tal när vi betraktar delbarhet, för de beter sig likadant som positiva.

Anmärkning 1 1 och a är alltid delare till a.

1. Kan ett heltal ha oändligt många delare?

Definition. Om $a = a_1 a_2 \dots a_n$ med $a_i > 1$ säger vi att a_1, a_2,$\dots a_n$ är en faktorisering av det ickenegativa heltalet a.

Om ingen av a_i kan faktoriseras ytterligare (utöver den triviala faktoriseringen $a_i = a_i$) kallar vi faktoriseringen en primtalsfaktorisering.

2. Primtalsfaktorisera talet 70. Är detta det enda sättet?
3. Vilka ickenegativa heltal saknar faktorisering?

9.2 Kongruensräkning

Om a och b har samma rest vid division med något tal n säger vi att a och b är *kongruenta* mod n. Vi skriver detta som att $a \equiv b \pmod{n}$ om och endast om $n|a-b$.

■ **Exempel 8** Timmar under ett dygn är kongruensräkning modulo 12: $22 + 4 \equiv 26 \equiv 2 \pmod{24}$. ■

4. En mattekollo-elev hade suttit uppe hela natten och räknat matte. Hen vaknade dagen efter och tittade på klockan på väggen och såg att den visade klockan 5. Vad kan klockan egentligen ha varit?
5. Vad är 13 mod 12?
6. Vad är det minsta ickenegativa heltalet x som uppfyller $x \equiv 13 \cdot 13 \pmod{12}$?

Vi kommer nu utifrån definitionen $n|a-b$ bevisa påståenden som gradvis låter oss behandla $\equiv$ mer och mer som ett likhetstecken.

7. Bevisa att om $x \equiv y$ så är $x+a \equiv y+a \pmod{n}$.

8. Bevisa att om $x \equiv y$ så är $x-a \equiv y-a \pmod{n}$.

9. Bevisa att om $x \equiv y$ så är $xa \equiv ya \pmod{n}$.

10. Hitta alla lösningar till $x+6 \equiv 0 \pmod{10}$

11. Hitta alla lösningar till $2x+6 \equiv 0 \pmod{10}$

12. Hitta alla lösningar till $2x+5 \equiv 0 \pmod{10}$

13. Ett pärlhalsband har ett antal pärlor i något av färgerna röd, vit eller grön. Första pärlan är röd, andra är vit, tredje är grön, fjärde är röd osv. Vilken färg har den 43:e pärlan?

9.3 Division

Vi har sett att linjära ekvationer inte alltid har heltalslösningar. Vi kan åtgärda detta genom att endast arbete modulo primtal, så låt oss härmed arbeta modulo primtalet p. Givet att $ax = b \pmod{n}$ så finns det ett k sådant att $b = ax+kn$.

$$ax = b \pmod{n} \implies \exists k : b = ax+kn \implies (p|ax+kn \implies p|b) \tag{9.1}$$

vilket ej går om b saknar någon delare som finns i både a och n.

14. Jämför moduloräkning med primtal jämfört med icke-primtal för t.ex. $10, 20, ..., 100$ och se om du kan hitta ett mönster. Vad hör modulo ett primtal ”bättre”?

15. Bevisa att om $a \not\equiv 0$ och $ax \equiv ay$ så är $x \equiv y \pmod{p}$. *Tips: primtalsfaktorisering*

Extra

Vi kan använda notation där vi låter variabeln x i $x \equiv 5 \pmod{7}$ benämna inte ett godtyckligt heltal som uppfyller kongruensrelationen, utan själva *ekvivalensklassen* av sådana heltal. Addition, subtraktion och multiplikation på dessa ekvivalensklasser ger den ekvivalensklass som innehåller heltalen med motsvarande operation applicerad på sig (Om $x = \{5, 12, 19 \dots\}$ är $x+1 = \{6, 13, 20 \dots\}$).

I uppgift 7-9 bevisade vi att addition, subtraktion och multiplikation är definerad på dessa ekvivalensklasser. I och med uppgift 15 är även division definerad. Kvoten mellan a och b kan skrivas a/b eller ab^{-1}. Vanliga potensregler gäller och till exempel är $a^2 = aa$ och $a^{-2} = a^{-1}a^{-1} = (aa)^{-1}$.

Anmärkning 2 Trots kvottecknet i $5/2 \equiv 6 \pmod{7}$ innehåller vänsterledet INTE ett rationellt tal. I kongruensräkning har vi endast heltal och modulo-ekvivalensklasserna.

16. Låt $\frac{x-1}{2} \equiv \frac{5}{3} \pmod{13}$. Hitta representanten för ekvivalensklassen x, alltså det minsta ickenegativa heltalet som ekvivalensklassen innehåller.

10. Kryptografi II: Fermats lilla sats

24 juli

1. Bestäm 1002 mod 100.
2. Bestäm 9^{2023} mod 8. Vad blir det om du tar mod 10 istället?

10.1 Fermats lilla sats

n är ett heltal för vilket $a^n \equiv 1 \pmod{p}$, där p är ett primtal och $0 < a < p$.

3. Finns a och p för vilka flera n är möjliga?
4. Finns a och p för vilka inga n är möjliga?

Låt p vara ett primtal och a vara ett heltal $1 \leq a < p$. Låt $f(x) = ax \pmod{p}$. Vi kan lite slarvigt tänka på denna funktionen som en funktion från $\{1, 2 \dots p-1\}$ till sig själv. (Egentligen är det en funktion från ekvivalensklassen av rester till sig själv).

5. Låt nu n vara den minsta positiva lösningen. Kan du hitta en övre begränsning på n? *Du kan hoppa tillbaka till denna uppgift senare.*
6. Låt $p = 7$ och $a = 2$. Rita den riktade grafen av noder $1, 2 \dots 6$ med kanter från x till $y = f(x) = 2x$. Till exempel går en pil från nod 2 till nod 4. Hur ser grafen ut? Vad i grafen motsvarar minsta positiva n?

Tips: testa olika (p, a) för att få intuition.

7. Föreställ dig motsvarande graf för ett godtyckligt par (p, a). Kan en nod ha 0, 1 eller fler utgående kanter? Kan en nod ha 0, 1 eller fler ingående kanter?
8. Antag att en nod 1 ingår i en cykel av längd n, alltså att n är det minsta heltalet så att man efter n steg med start i nod 1 kan nå nod 1 igen. Kan någon annan nod ingå i en cykel av annan längd?
9. Återgå till grafen med $p = 7$ och $a = 2$. Hur förhåller sig cykellängderna till antalet noder, 6?
10. För en sådan här graf för godtyckligt (p, a), hur förhåller sig cykellängderna till antalet noder, $p - 1$?

Detta ger oss en sats som lyder

$$a^{???} = 1 \pmod{p} \tag{10.1}$$

och kallas **Fermats lilla sats**.

11. Vad borde stå istället för ??? ?
12. Använd Fermats lilla sats för att bestämma $3^{100\,000}$ mod 53.

10.2 Euler $\phi(n)$, generalisering till ickeprimtal

Vi ska nu göra om tidigare uppgifter modulo ett icke-primtal n (ej samma som exponenten tidigare).

13. (variant av uppgift 10) Bevisa att om a är relativt primt till n (alltså ej har gemensamma delare) och $ax \equiv ay$ så är $x \equiv y \pmod{n}$.

Detta innebär at vi får dividera så länge vi bara arbetar med tal relativt prima n.

14. (variant av uppgift 16) Föreställ dig motsvarande graf nu för (n,a). Hur många ingående respektive utgående kanter kan en nod ha?

15. (variant av uppgift 17) Kan någon nod ingå i en cykel av annan längd än cykeln som innehåller nod 1?

16. (variant av uppgift 19) Hur förhåller sig cykellängderna till antalet noder i grafen?

Låt $\phi(n)$ beteckna antalet noder i grafen, alltså antalet tal relativt prima till n.

17. Hur lyder den generaliserade Fermats lilla sats?

(denna kallas för Eulers sats, men Euler har många satser, så det är ett dåligt namn)

18. Hitta algebraiska uttryck för $\phi(p)$, $\phi(p^2)$, $\phi(p^3)$ samt $\phi(pq)$ där p och q är primtal ($p \neq q$).

11. Kryptografi III: RSA

25 juli

11.1 Vilka tal har en invers modulo n?

1. Vilka tal har en invers modulo 10?
2. I detta problem visar vi följande sats:

> **Sats 11.1. — Bézouts identitet.** Låt a, b vara heltal som inte båda är 0 och låt $D = \text{sgd}(a,b)$. Då finns heltalslösningar (x,y) till ekvationen $ax+by = D$.

Låt $S = \{ax+by \mid x,y \in \mathbb{Z}\} \cap \mathbb{Z}_{>0}$. Låt D var det minsta elementet i S.

a) Visa att om $d|a$ och $d|b$ så gäller att $d|D$.

b) Antag att D inte delar a. Visa att du kan skriva $a = Dq+r$ för något $0 < r < D$.

c) Visa att om D inte delar a så måste det finnas ett element i S mindre än D.

d) Dra slutsatsen att $D = \text{sgd}(a,b)$.

3. När har a en invers modulo n?

11.2 RSA

Låt $pq = n$. Funktionen $\varphi(n)$ är svår att beräkna om man inte vet p och q eftersom det är ekvivalent med att faktorisera n (och primtalsfaktorisering är beräkningstungt!).

Låt n, e och $c \equiv m^e \pmod{n}$ vara välkända. Säg att vi även känner till det hemliga talet $\varphi(n)$ (kanske eftersom vi valde p och q från början).

4. Vilket tal d uppfyller $c^d \equiv m \pmod{n}$?
5. Hur kan detta tal d beräknas?
6. Antag att du undvikit det kniviga i att faktorisera n genom att beräkna d på något annat vis. Hur kan du använda d för att beräkna $\varphi(n)$?
7. Hur kan du använda att du vet $\varphi(n)$ för att beräkna p och q?
   Ledtråd:

$$\left(\frac{a+b}{2}\right)^2 - \left(\frac{a-b}{2}\right)^2 = ab \tag{11.1}$$

Detta är grunden i kryptosystemet RSA. Alice väljer stora primtal p och q, och berättar för Bob $n = pq$, $e = 2^{16}+1 = 65537$. Under tiden tjuvlyssnar Eve (”eavesdropper”).

Bob vill skicka det meddelandet m till Alice. Bob beräknar $c \equiv m^e \pmod{n}$ och skickar detta talet till Alice. Eve hör detta också.

Alice beräknar $m \equiv c^d \pmod{n}$ och kan därmed läsa det hemliga meddelandet!
Eve känner till

$$n = pq \tag{11.2}$$
$$e = 2^{16} + 1 = 65537 \tag{11.3}$$
$$c \equiv m^e \pmod{n} \tag{11.4}$$

men att utifrån detta beräkna m är ekvivalent[1] med att faktorisera n. Vi har använt ett algoritmiskt svårt problem för att upprätta säker kommunikation trots avlyssning!

[1]Man kanske kunna hitta m utan att gå via d? (för vi bevisade att att hitta d är ekvialent med faktorisering)? Ja, detta är faktiskt en giltig metod och kallas *diskret logaritm*. Men detta är också ett beräkningssvårt problem.

12. Kryptografi IV: RSA: Attackdags!

26 juli

12.1 Attackdags!

Följande funktioner används i problemformuleringarna och kan användas i lösningar. Funktionerna bygger på tidigare innehåll i kursen. Ni behöver inte förstå implementationerna, utan börja på uppgifterna och gå tillbaka vid behov.

```
from random import randint

def gcd(a: int, b: int) -> int:
  while a != 0,
    a, b = b%a, a
  return b

def extended_gcd(a: int, b: int) -> tuple[int, int, int]:
  # Hitta d, s, t som minimerar nollskilda
  # absolutvärdet av `s*a + t*b = d`
  if a == 0:
    d, s, t = b, 0, 1
  else:
    k, r = b // a, b % a
    # b == k*a + r
    dd, ss, tt = extended_gcd(r, a)
    # ss*r + tt*a = dd
    # ss(b - k*a - r) = 0
    # => (tt - ss*k)*a + ss*b = dd
    d, s, t = dd, tt - ss*k, ss
  assert s*a + t*b == d
  return d, s, t

def pow(base: int, exponent: int, mod: int) -> int:
  if exponent < 0:
    d, s, t = extended_gcd(base, mod)
    assert d == 1
    # s*base + t*mod == 1
    # s*base == 1 (% mod)
    base, exponent = s, -exponent
  ret = 1
  while exponent > 0:
    if exponent % 2 == 1:
      ret = (ret * base) % mod
    base = (base * base) % mod
  return ret

def is_prime(p: int) -> bool:
  for n in range(2, min(p, 20)):
    if pow(n, p-1, p) != 1:
```

```
      return False
  return True

def next_prime(p: int) -> int:
  while not is_prime(p): p += 1
  return p

def random_bits(bits: int) -> int:
  return randint(2**(bits-1), 2**bits - 1)

def random_prime(bits: int) -> int:
  return next_prime(random_bits(bits))
```

1. Kan vi beräkna p och q?

```
p = random_prime(32)
q = random_prime(32)
n = p*q
print(f"{n = }")
```

2. Kan vi beräkna p, q_1 och q_2?

```
p = random_prime(512)
q1 = random_prime(512)
q2 = random_prime(512)
n1 = p*q1
n2 = p*q2
print(f"{n1 = }")
print(f"{n2 = }")
```

3. Kan vi hitta m?

```
p = random_prime(512)
q = random_prime(512)
n = p*q
e1 = 2**16 + 1
e2 = 2**18 + 1
m = SECRET
c1 = pow(m, e1, n)
c2 = pow(m, e2, n)
print(f"{n = }")
print(f"{c1 = }")
print(f"{c2 = }")
```

4. Kan vi hitta m?

```
def smooth():
  ret = 1
  for t in range(2,1000):
    if is_prime(t) and randint(0,1):
      ret *= t
  return ret

def unsafe_prime():
    while True:
      # Automatically relatively prime to a bunch
      # of small primes, which is good right?
      p = smooth() + 1
      if is_prime(p):
        return p

p = unsafe_prime()
q = unsafe_prime()
n = p*q
e = 2**16 + 1
m = SECRET
c = pow(m, e, n)
print(f"{n = }")
print(f"{c = }")
```

13. Linjär Algebra, del 1

23 juli

Vektorer

1. Vad blir $\vec{u} + \vec{v}$ om $\vec{u} = (1,2,3)$ och $\vec{v} = (0,-4,2)$?
2. Vad blir $\vec{u} - \vec{v}$ om $\vec{u} = (1,2,3,4,5)$ och $\vec{v} = (0,-4,2,3,-3)$?

Introduktion Matriser

Definition. En matris som består av n-rader och m-kolonner kallas för en $n \times m$-matris (n *gånger* m*-matris*). Storleken för en matris är alltså n och m.

$$A = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,m} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,m} \end{pmatrix}$$

Ett element på det i:te raden och j:te kolonnen betecknas som $a_{i,j}$.

3. Hur ska matris B och C se ut för att följande alltid ska gälla?
   (a) $A + B = A$?
   (b) $A + (-A) = C$?

Fundera på detta! Varför gäller följande axiom för matriser A, B, C och konstanter λ och μ ?

(a) $A + (B + C) = (A + B) + C$ (Associativa lagen)

(b) $A + B = B + A$ (Kommutativa lagen)

(c) $\lambda(A + B) = \lambda A + \lambda B$ (Distributiva lagen)

(d) $(\lambda + \mu) \cdot A = \lambda A + \mu A$

(e) $\lambda \cdot (\mu \cdot A) = (\lambda \cdot \mu) \cdot A$

4. Om $A = \begin{pmatrix} 2 & 0 \\ 2 & 3 \end{pmatrix}$ och $B = \begin{pmatrix} 2 & 3 \\ 0 & 7 \end{pmatrix}$, och vi har matrisekvationen $5A + 2X = 4B$.
   (a) Vilken storlek har matris X?
   (b) Lös matrisekvationen och hitta matris X.

Matrismultiplikation

Definition. Matrismultiplikation med en matris A med storlek $n \times m$ och en matris B med storlek $m \times p$ skapar produkten

$$AB = C$$

där C är en matris med storleken $n \times p$. Varje element $c_{i,j}$ i C är definierad som:

$$c_{i,j} = a_{i,1}b_{1,j} + a_{i,2}b_{2,j} + \cdots + a_{i,m}b_{m,j} = \sum_{k=1}^{m} a_{i,k}b_{k,j}.$$

Matrismultiplikation är endast definierad när det är samma antal *kolonner* i A som antal *rader* i B.

5. Hur ska matris E se ut för att $AE = EA = A$ alltid ska gälla?

Fundera på detta! Varför gäller följande axiom på matrismultiplikation? (Skippa att bevisa!)

(a) $A \cdot (B \cdot C) = (A \cdot B) \cdot C$ (Associativa lagen)

(b) $(A + B) \cdot C = AC + BC$ (Distributiva lagen)

(c) $C \cdot (A + B) = CA + CB$ (Distributiva lagen åt andra hållet)

6. Gäller den kommutativa lagen $A \cdot B = B \cdot A$ för matriser?

7. Varför frågade vi om $AE = EA$ i fråga 5? Borde det inte alltid gälla att $A \cdot B = B \cdot A$?

8. Vad är x om

$$\begin{pmatrix} 2 & 5 \end{pmatrix} \begin{pmatrix} 2 & x & 4 \\ 3 & 5 & 2 \end{pmatrix} \begin{pmatrix} x \\ 3 \\ 0 \end{pmatrix} = \mathbf{0}?$$

9. Om $A = \begin{pmatrix} 1 & 2 & 2 \\ 2 & 1 & 2 \\ 2 & 2 & 1 \end{pmatrix}$, uttryck $A^2 - 4A$ med hjälp av enhetsmatriser (**SPOILER:** Svaret till fråga 5 är en enhetsmatris och brukar betecknas som E).

10. Ett polynom $p(x) = c_0 + c_1 x + c_2 x^2$ kan representeras som en vektor $\overrightarrow{p} = \begin{pmatrix} c_0 \\ c_1 \\ c_2 \end{pmatrix}$. Hitta derivata-matrisen D så att $D \cdot \overrightarrow{p} = \overrightarrow{q}$ där $\overrightarrow{q} = \begin{pmatrix} c_1 \\ 2c_2 \\ 0 \end{pmatrix}$ och $p'(x) = q(x) = c_1 + 2c_2 x$.

11. Rotationsmatrisen $R = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$ kan multipliceras $R \cdot \vec{v}$ för att vrida på en vektor $\vec{v}$ moturs med en vinkel θ. Vad blir vektorn $\vec{u} = \begin{pmatrix} 0 \\ \sqrt{2} \end{pmatrix}$ efter att man roterar den 45° moturs?

12. Visa att ifall man transformerar en vektor med en rotationsmatris med en vinkel α och sedan en annan rotationsmatris med vinkel β, så är det ekvivalent som att enbart transformera en vektor en gång med en rotationsmatris med vinkeln $\alpha + \beta$.

13. Matriserna A och B konstrueras på så sätt att produkten $C = AB$ är definierad. Vilka dimensioner kan matriserna A och B ha för att följande krav ska stämma:

(a) Alla element i A, B och C är udda heltal.

(b) Alla element i A, B och C är jämna heltal.

14. A och B är två matriser sådana att $AB = B$ och $BA = A$. Visa att $A^2 + B^2 = A + B$.

15. A och B är två matriser sådana att $AB = BA$. Visa att följande alltid stämmer för alla $n \in \mathbb{N}$:

(a) $A^n B = BA^n$.

(b) $(AB)^n = A^n B^n$.

(c) $A^{2n} - B^{2n} = (A^n - B^n)(A^n + B^n)$.

16. **— Extra.** Hitta två matriser A och B där $A \neq B$, men $AB = BA$.

Extra: Transponat

Definition. Transponatet till en matris A betecknas som A^T. A^T är en reflektion av A genom matrisen A : s huvuddiagonal. Raderna i A blir på så sätt kolonnerna i A^T, och kolonnerna i A blir raderna i A^T.

Exempel:

$$\begin{pmatrix} 3 & 1 & 4 \\ 3 & 7 & 2 \end{pmatrix}^T = \begin{pmatrix} 3 & 3 \\ 1 & 7 \\ 4 & 2 \end{pmatrix}.$$

17. Givet $A = \begin{pmatrix} 2 \\ -5 \\ 3 \end{pmatrix}$ beräkna AA^T och A^TA.

18. Givet $A = \begin{pmatrix} 3 & -1 \\ 2 & 0 \end{pmatrix}$ beräkna AA^T och A^TA.

19. Vilka krav behöver uppfyllas för att $A = A^T$?

20. Givet $A = \begin{pmatrix} 1 & 0 & 9 \\ 5 & 8 & 6 \end{pmatrix}$, $B = \begin{pmatrix} 3 & 6 & 1 \\ 4 & 2 & 5 \end{pmatrix}$ och $C = \begin{pmatrix} 3 & 4 \\ 6 & 2 \\ 1 & 5 \end{pmatrix}$, verifiera det följande:

(a) $(A^T)^T = A$.

(b) $(\lambda A)^T = \lambda A^T$, $\lambda \in \mathbb{R}$.

(c) $(A+B)^T = A^T + B^T$.

(d) $(AC)^T = C^T A^T$.

21. Gäller 20 **(a)-(d)** generellt?

14. Linjär Algebra, del 2

24 juli

Determinanter

Definition. Determinanten till en 2×2-matris $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ är:

$$\det(A) = |A| = \begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc.$$

1. Vad är arean av ett parallellogram med basen 5 och höjden 4?

2. Vad är arean av ett parallellogram med sidorna $\sqrt{2}$ och 2, och två av vinklarna i parallellogrammet är $45°$?

3. Vad är arean av ett parallellogram som spänns upp av vektorerna $\vec{u} = \begin{pmatrix} 3 \\ 2 \end{pmatrix}$ och $\vec{v} = \begin{pmatrix} 3 \\ 4 \end{pmatrix}$?

4. Vad är x om $\begin{vmatrix} 3 & x \\ 4 & x-1 \end{vmatrix} = -5$?

5. **— Extra.** Vad är arean av den konvexa polygonen som har sina hörn i koordinaterna $(0,0)$, $(-2,2)$, $(2,5)$, $(5,1)$ och $(7,3)$?

Inversa Matriser

Sats 14.1. En matris A är inverterbar om och endast om $\det A \neq 0$.

6. Vad är inversen till matrisen $A = \begin{pmatrix} 3 & 1 \\ 2 & 1 \end{pmatrix}$?

7. Rotationsmatrisen multiplicerat med en vektor från vänster innebär att vektorn roteras motsols i en vinkel θ. (Notera att $\sin^2 x + \cos^2 x = 1$)

   (a) Vad är inversen till rotationsmatrisen $R = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$?

   (b) Vilken funktion kan inversen ha?

8. Bestäm k så att matrisen $\begin{pmatrix} 3 & 2 \\ 6 & k \end{pmatrix}$ så matrisen inte är inverterbar.

9. Lös matrisekvationen $\begin{pmatrix} 5 & 6 \\ 4 & 5 \end{pmatrix} X = \begin{pmatrix} 3 & 1 \\ 4 & 1 \end{pmatrix}$.

10. **— Extra.** Visa att en inversen till en matris är unik.

11. **— Extra.** Låt A och B vara två $n \times n$-matriser sådana att $A + B = AB$. Visa att $AB = BA$.

3. Linjär Algebra, del 3

25 juli

3 x 3 Determinanter

Definition. Determinanten till en 3×3-matris $A = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$ är:

$$\det(A) = \begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix} = aei + bfg + cdh - ceg - bdi - afh = a\begin{vmatrix} e & f \\ h & i \end{vmatrix} - b\begin{vmatrix} d & f \\ g & i \end{vmatrix} + c\begin{vmatrix} d & e \\ g & h \end{vmatrix}$$

1. Vad är determinanten av matrisen $\begin{pmatrix} 3 & 2 & 1 \\ 4 & 2 & -1 \\ 5 & 2 & 3 \end{pmatrix}$?

2. Visa att $\begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix} = a\begin{vmatrix} e & f \\ h & i \end{vmatrix} - d\begin{vmatrix} b & c \\ h & i \end{vmatrix} + g\begin{vmatrix} b & c \\ e & f \end{vmatrix}$.

3. På vilka andra sätt kan man dela upp en 3×3-determinant in i 3 st 2×2-determinanter?

Kryssprodukt

Definition. Längden av en vektor $\overrightarrow{u} = (u_x, u_y, u_z)$ kan beräknas som

$$|\overrightarrow{u}| = \sqrt{u_x^2 + u_y^2 + u_z^2}.$$

4. Vad är längden av vektorn $\overrightarrow{v} = (3, -4, 0)$?

Sats 3.1. Kryssprodukten av 2 vektorer

$$\overrightarrow{u} = (a_x, a_y, a_z) = a_x x + a_y y + a_z z$$

$$\overrightarrow{v} = (b_x, b_y, b_z) = b_x x + b_y y + b_z z$$

kommer att bilda en vektor som är ortogonal mot både $\overrightarrow{u}$ och $\overrightarrow{v}$, och den kan beräknas som:

$$\overrightarrow{u} \times \overrightarrow{v} = \begin{vmatrix} x & y & z \\ a_x & a_y & a_z \\ b_x & b_y & b_z \end{vmatrix}.$$

5. Hitta vektorn som går från punkten $(3,-2,5)$ till $(5,1,0)$.

6. Hitta en vektor som är ortogonal mot både vektorn $\vec{u} = (1,0,0)$ och $\vec{v} = (0,1,0)$.

7. Hitta en vektor som är ortogonal mot både vektorn $\vec{u} = (1,2,3)$ och $\vec{v} = (-2,1,5)$.

Definition. Arean av ett parallellogram uppspänt av vektorerna $\vec{u}$ och $\vec{v}$ kan beräknas som längden av kryssprodukten mellan $\vec{u}$ och $\vec{v}$, men även:

$$|\vec{u} \times \vec{v}| = |\vec{u}||\vec{v}| \sin\theta$$

där θ är vinkeln mellan $\vec{u}$ och $\vec{v}$.

8. Vad är arean av parallellogrammet uppspänt av vektorerna $\vec{u} = (1,2,3)$ och $\vec{v} = (-2,1,5)$?

9. Vad är vinkeln mellan $\vec{u} = (1,2,3)$ och $\vec{v} = (-2,1,5)$?

10. Givet är att $\vec{u} = (3,2,-1), \vec{v} = (1,4,5)$ och $\vec{w} = (7,-8,5)$. Kan $\vec{w}$ vara en vektor som är ortogonal mot $\vec{u}$, och $\vec{v}$?

11. **— Extra.** Hitta 2 vektorer $\vec{u}$, och $\vec{v}$ så att vektorerna $\vec{u}$, $\vec{v}$ och $\vec{w}$ är alla parvis ortogonala mot varandra, om $\vec{w} = (3,1,4)$.

II Matematik - Avancerad

1. Kombinatorik 1 - Mängdsystem

18 juli

Definition. Ett mängdsystem är en familj av delmängder av en mängd. Till exempel kan vi ha ett mängdsystem $F = \{A_1, A_2, A_3\}$ på mängden $X = \{1,2,3,4\}$ där $A_1 = \{1,2,3\}$, $A_2 = \{1,2,4\}$ och $A_3 = \{2,3\}$. Vi kallar X för en grundmängd (inte officiell terminologi). Man kan tänka det som att X är en grupp personer och varje element i F är en klubb med medlemmar. Man måste bara anta att inga två klubbar har exakt samma medlemmar då element i mängder inte kan upprepas. Så $\{1,2,3\} = \{1,1,2,3\}$ till exempel.

Definition. Vi betecknar $\{1,2,\ldots,n-1,n\}$ med $[n]$.

När är vår grundmängd onödigt stor?

Vi säger att grundmängden X är för stor om vi kan ta bort något element $x \in X$ från den så att alla element i vårt mängdsystem fortfarande är distinkta när vi bortser från x. Dvs om A och B är i mängdsystemet så måste $A \setminus \{x\} \neq B \setminus \{x\}$.

1. Hitta ett mängdsystem på $[n]$ av storlek $n+1$ så att grundmängden inte är för stor.

2. Visa att alla mängdsystem på $[n]$ av storlek n har en för stor grundmängd.

Definition. En transversal av ett mängdsystem $F = \{A_1, A_2, \ldots, A_m\}$ är en mängd $\{x_1, x_2, \ldots, x_m\}$ av element i X så att $x_i \in A_i$ för alla i och så att alla x_i är distinkta.

Vilka mängdsystem har en transversal?

3. Visa att om mängdsystemet $F = \{A_1, A_2, \ldots, A_m\}$ har en transversal så kommer

$$\left|\bigcup_{i \in I} A_i\right| \geq |I|$$

gälla för alla $I \subset [m]$.

4. (Halls giftermålssats) Det finns n män och n kvinnor. Varje man känner några av kvinnorna. För varje k från 1 till n så känner k män alltid åtminstone k kvinnor.

Låt oss kalla en mängd med m män för *kritisk* om de känner exakt m kvinnor. Visa att:
a) Skärningen och unionen av kritiska mängder också är kritiska mängder.
b) Om man tar bort en kritisk mängd män tillsammans med alla kvinnor som de känner, så kommer villkoren för satsen fortfarande vara uppfyllda.

c) Visa att det går att bilda n gifta par så att man kände varandra sedan innan i paret.

5. Visa att kravet i problem 3 är tillräckligt genom att tolka om män och kvinnor i Halls giftermålssats som mängder.

6. Låt varje man känna åtminstone m kvinnor och varje kvinna som mest m män. Visa att i så fall så är villkoren i Halls giftermålssats uppfyllda.

Tillämpning av Halls giftermålssats

7. Elever löser problem på en Matteklubbsträff. Det visade sig att var och en hade löst 4 problem och varje problem hade lösts av 4 elever. Visa att man kan organisera vem som går fram till tavlan på varje problem så att varje problem blir redovisat på tavlan och varje elev redovisar exakt ett problem.

8. Man tog bort 7 rutor från ett schackbräde. Visa att det går att sätta ut 8 torn på brädet så att de inte hotar varandra.

9. I varje rad och varje kolumn på ett 8×8-bräde står exakt 3 pjäser. Visa att man kan välja 8 av pjäserna - en i varje rad och en i varje kolumn.

10. På ett schackbräde är 16 av 64 rutor markerade, på så sätt att varje rad och varje kolumn har exakt två rutor markerade. Visa att man kan sätta ut 8 svarta och 8 vita pjäser på de markerade rutorna så att varje rad och varje kolumn har pjäser av olika färg.

11. (En latinsk rektangel kompletteras till en latinsk kvadrat.) I rutorna på en $m \times n$-rektangel ($m < n$) står talen 1 till n på så sätt att varje rad och varje kolumn innehåller olika tal. Visa att rektangeln kan kompletteras upp till en $n \times n$-kvadrat med talen från 1 till n så att det fortfarande står olika siffror inom varje rad och varje kolumn.

12. En trollkarl tillsammans med en assistent visar följande trick. En åskådare skriver upp en följd med 101 siffror. Assistenten täcker över två grannsiffror med svarta cirklar. Därefter kollar trollkarlen på sifferföljden. Hans jobb är att lista ut vilka siffror som täcktes över och i vilken ordning. Visa att det går att garantera att tricket lyckas.

Svåra problem

13. Låt F vara ett mängdsystem av en oändlig mängd X. Varje element i F är ändligt. Det visar sig att för alla ändliga delmängder A av X finns det disjunkta $B, C \in F$ så att $A = B \cup C$. Visa att för alla positiva heltal k finns det en ändlig delmängd av X som kan representeras som unionen av två disjunkta mängder i F på minst k sätt.

2. Kombinatorik 2- Spernersystem

19 juli

Definition. Ett mängdsystem $F = \{A_1, A_2, ..., A_m\}$ kallas för ett Spernersystem om för alla distinkta index $i, j \in [m]$ så är $A_i \not\subseteq A_j$.

Definition. Given en mängd X så är en kedja en lista delmängder $A_1, A_2, ..., A_k$ av X så att $A_1 \subseteq A_2 \subseteq ... \subseteq A_k$. Vi säger att den är maximal om $k = n+1$.

Definition. Vi betecknar $1 \cdot 2 \cdot ... \cdot (n-1) \cdot n$ med $n!$ och med $\binom{n}{k}$ antalet sätt att välja ut k saker från n saker. Notera att $\binom{n}{k} = \frac{n!}{k!(n-k)!}$.

Sperners sats

Vi börjar med ett resultat som bestämmer hur stort ett Spernersystem kan vara.

1. Visa att det finns $n!$ olika maximala kedjor av en mängd X med n element.
2. Visa att givet en delmängd $A \subseteq X$ så finns det minst $\frac{n!}{\binom{n}{\lfloor \frac{n}{2} \rfloor}}$ maximala kedjor som innehåller A.
3. Visa att två element i ett Spernersystem inte kan vara i samma kedja.
4. Visa att ett Spernersystem med grundmängd $[n]$ har max $\binom{n}{\lfloor \frac{n}{2} \rfloor}$ element.
5. Kan du hitta ett Spernersystem av storleken i problem 4?

LYM-olikheten

Vi förfinar Sperners sats lite.

6. Låt F vara ett Spernersystem på $X = [n]$ och sätt $F_k = F \cap X^{(k)}$. Visa att
$$\sum_{k=0}^{n} \frac{|F_k|}{\binom{n}{k}} \leq 1.$$
7. Kan du härleda Sperners sats från LYM-olikheten?

Ännu mer generalisering

Vi ska nu lämna delmängder helt och tänka oss följande upplägg. Vi har $n+1$ mängder numrerade från 0 till n som kallas nivåer. Elementen i en nivå får relatera till vissa element i nivån under och över utifrån någon regel. Vi bestämmer också att två element i niver som inte är grannar relaterar om vi kan hoppa från elementet i den lägre nivån till den högre genom att bara hoppa uppåt via relationer som bestämts mellan grannnivåer. Till exempel kan vi ha att nivåer är generationer av personer och att två personer relaterar om de är barn och förälder. Vi

låtsas här att nivåerna är fina. I verkligheten är det inte så. Då skulle också barnbarn relatera till sina mor- och farföräldrar eftersom vi kan gå från barnbarnet till dess föräldrar och sedan till dess mor- eller farföräldrar. Ett sånt här system kallas ett poset (partially ordered set).

8. Kan du beskriva hur delmängder bildar ett poset?

Definition. Låt F vara ett poset med nivåer $F_0, F_1, ..., F_n$. En antikedja är en delmängd av F så att inga två element relaterar med varandra.

9. Beskriv hur ett Spernersystem är en antikedja.

Definition. Vi säger att ett poset F med nivåer $F_0, F_1, ..., F_n$ är regelbundet kopplat om varje element i någon viss nivå relaterar till lika många element i nivån ovan och varje element i någon viss nivå relaterar till lika många element i nivån under. Ett element kan alltså relatera till olika antal under och ovanför sig och hur många element som relateras till kan skiljas åt mellan nivåer. Ett exempel skulle kunna vara att alla i milleniegenerationen skulle ha exakt 5 föräldrar och 3 barn. Alla i generation Z har 2 föräldrar och 7 barn o.s.v.

10. Låt F vara ett poset med nivåer $F_0, F_1, ..., F_n$ och låt $A \subseteq F$ vara en antikedja. Visa olikheten

$$\sum_{i=0}^{n} \frac{|A \cap F_i|}{F_i} \leq 1.$$

Kan du använda denna satsen för att lösa problem 11 som kallas Krafts olikhet. Den är tyvärr inte döpt efter William.

11. Vi har en ändlig lista med ändliga binära talföljder. Det visar säg att ingen talföljd är ett prefix till någon annan i listan. Säg att antalet talföljder i listan av längd k är n_k. Visa att

$$\sum_{k=0}^{\infty} \frac{n_k}{2^k} \leq 1.$$

Svåra problem

12. Vi har en ändlig lista med ändliga binära talföljder. Det visar säg att alla binära talföljder som kan byggas av dessa kan göras så på ett unikt sätt av dem. Säg att antalet talföljder av längd n är n_k. Visa att

$$\sum_{k=0}^{\infty} \frac{n_k}{2^k} \leq 1.$$

13. Vi har n komplexa tal $z_1, z_2, ..., z_n$, alla med absolutbelopp större än 1. Vi vill göra så många summor av dessa som möjligt, av 2^n möjliga totalt, så att inga två summor skiljer sig från varandra med mer än 1. Visa att vi max kan ta ut $\binom{n}{\lfloor \frac{n}{2} \rfloor}$.

3. Kombinatorik 3- Skärande system

20 juli

Definition. Ett skärande system är ett mängdsystem så att varje par av mängder i systemet delar minst ett element.

Maximal storlek

1. Visa att varje skärande system F av $[n]$ har maximalt 2^{n-1} mängder i sig. Kan du hitta ett sådant system?

Erds-Ko-Rado

Låt n och r vara två positiva heltal så att $n \geq 2r$. Vi ska bestämma hur stort ett skärande system av $[n]^{(r)}$ kan vara. Kalla det skärande systemet A.

2. Betrakta en cykel av talen 1 till n. En sån cykel ger n stycken r-element delmängder genom att välja på varandra följande element i cykeln. Till exempel kan vi ta cykeln (314625) om $n = 6$ vilket ger mängderna $\{3,1,4\},\{1,4,6\},\{4,6,2\},\{6,2,5\},\{2,5,3\}$ och $\{5,3,1\}$ om $r = 3$. Visa att max r stycken av de delmängder bildade från en given cykel kan vara i A.

3. Visa att antalet par (S,C) där S är en mängd i A och C är en cykel som ger S är $|A|r!(n-r)!$.

4. Visa nu att samma sorts par är max $r(n-1)!$ i antal.

5. Dra slutsatsen att $|A| \leq \binom{n-1}{r-1}$. Kan du hitta ett sådant system?

Svåra problem

6. Låt n,k vara positiva heltal så att $k \leq n/2$. Vi har ett $n \times n$ bräde och väljer ut några $k \times k$ delbräden. Det visar sig att alla par av delbräden har minst 1 ruta gemensamt. Visa att antalet delbräden som valts ut är max

$$\binom{n-1}{k-1}^2.$$

7. Låt F vara ett mängdsystem som har följande egenskaper:
(i) Om $X \in F$ så finns det $Y \in F$ och $Z \in F$ så att $Y \cap Z = \emptyset$ och $X = Y \cup Z$.
(ii) Om $X \in F$ och $Y \cup Z = X$, $Y \cap Z = \emptyset$ så är $Y \in F$ eller $Z \in F$.
Visa att det finns en minskande kedja $X_0 \supseteq X_1 \supseteq X_2 \supseteq \ldots$ av mängder $X_n \in F$ så att

$$\bigcap_{i=0}^{\infty} X_i = \emptyset.$$

4. Kombinatorik 4 - Kombinatorisk talteori

21 juli

Delbarhet

1. Visa att bland $n+1$ positiva heltal mindre eller lika med $2n$ finns det någon som delar en annan.

2. Visa att bland $n+1$ positiva heltal mindre eller lika med $2n$ finns det två som är relativt prima.

Delsummor

Uncle Ben, eller Benjamin som han brukar kallas, har lagt en förbannelse över idrotten på Mattekollo. Den ökar chansen att föremål går sönder. Därför bestämmer sig Kevin och Erik för att anteckna hur många saker som går sönder varje dag. De får en följd av $2n-1$ positiva heltal. De undrar ifall det finns n stycken tal i följden vars summa är delbar med n. Kan du hjälpa dem med detta?

Sats 4.1. — Erds-Ginzburg-Ziv. Utav $2n-1$ heltal finns det n stycken vars summa är delbar med n.

För att bevisa satsen ska vi bygga upp lite resultat på vägen. Vi börjar med ett uppvärmningsproblem.

3. Låt $a_1, a_2, \ldots, a_n$ vara heltal. Visa att det finns en icke-tom delsumma som är delbar med n.

Definition. Låt A, B vara två delmängder av $\mathbb{Z}$ eller $\mathbb{Z}_n$ (restklasser modulo n). Vi definierar mängden $A+B = \{a+b : a \in A, b \in B\}$.

4. Låt A och B vara ändliga delmängder av $\mathbb{Z}$. Visa att $|A+B| \geq |A|+|B|-1$.

För $\mathbb{Z}_p$ (p är primtal) har vi ett motsvarande resultat som kallas Cauchy-Davenports sats.

5. Låt A och B vara icke-tomma delmängder av $\mathbb{Z}_p$ där p är ett primtal så att $|A|+|B| \leq p+1$. Visa med hjälp av induktion på $|A|$ eller på annat sätt att $|A+B| \geq |A|+|B|-1$. Du kan anta utan inskränkning att $1 \leq |A| \leq |B|$.

6. Härled att om $A_1, A_2, \ldots, A_k$ är icke-tomma delmängder av $\mathbb{Z}_p$ så att

$$\sum_{i=1}^{k} |A_i| \leq p-k+1$$

så är

$$|A_1 + \ldots + A_k| \geq \sum_{i=1}^{k} |A_i| - k + 1.$$

Vi kommer nu visa sats 4.1 med hjälp av induktion. Vi börjar med induktionssteget.

7. Visa att om sats 4.1 gäller för $n = a$ och $n = b$ så gäller den för $n = ab$ där a, b är positiva heltal.

Vi avslutar nu beviset genom att visa det för primtal. Vi kan sedan multiplicera ihop vilka primtal vi vill och få att satsen gäller för alla positiva heltal n.

8. Visa sats 4.1, om n är ett primtal, genom att välja lämpliga mängder A_i i problem 6.

Svåra problem

9. Låt a_n vara en talföljd av positiva heltal så att för alla primtal p kommer $(a_1, a_2, ..., a_p)$ bilda ett fullständigt restsystem modulo p. Visa att

$$\lim_{n\to\infty} \frac{a_n}{n} = 1.$$

10. Låt A vara en ändlig mängd reella tal. Visa att

$$|A + A| \cdot |A \cdot A| \geq \frac{1}{32}|A|^{5/2}.$$

5. Konvexitet 1. Hellys sats.

18 juli

Definition. En figur kallas *konvex* om den tillsammans med dess två valfria punkter också innehåller hela sträckan som förbinder dem.

1. Visa att skärningen mellan ett valfritt antal konvexa figurer är en konvex figur.

2. Visa att om alla triangelns hörn hör till en konvex figur så hör även varje inre punkt utav triangeln till den figuren.

3. På en linje finns några sträckor givna sådana att varje par av dem har en gemensam punkt. Visa att alla sträckorna har en gemensam punkt.

4. På planet finns några rektanglar givna vars sidor är parallella med koordinataxlarna. Visa att om varje par av rektanglar har en punkt gemensamt så har även alla rektanglarna en punkt gemensamt.

5. Fyra konvexa figurer på planet är givna. Vilka tre av dem som helst innehåller en gemensam punkt. Visa att det finns en punkt som tillhör alla fyra figurerna.

6. Visa att i vilken som helst konvex månghörning utom ett parallellogram så kan man välja och förlänga tre sidor och bilda på så sätt en triangel som omfattar månghörningen.

Definition. Ett *konvext hölje* av en figur är den minsta (genom inklusion) konvexa figuren som innehåller den.

7. Visa att varje figur **(a)** har ett konvext hölje **(b)** har ett unikt konvext hölje.

8. **(Hellys sats).** Det finns n givna konvexa figurer i planet. Om vilka tre av dem som helst har en gemensam punkt så har alla figurer minst en gemensam punkt.

9. Ge ett exempel på när Hellys sats inte uppfylls för icke-konvexa mängder.

Extrauppgifter

10. Betrakta en valfri konvex sjuhörning och alla fyrhörningar som bildas av fyra på varandra följande hörn i den. Visa att det finns en punkt inuti sjuhörningen som inte tillhör någon av fyrhörningarna.

11. Några punkter är givna på planet. Vilka som helst tre av dem kan täckas med en cirkel med radie 1. Visa att alla punkterna kan täckas med en cirkel med radie 1.

12. **(Jungs sats.)** Några punkter är givna i planet, avståndet mellan varje par av punkter är inte längre än 1. Visa att alla punkterna kan täckas med en cirkel med radie $\frac{1}{\sqrt{3}}$.

6. Konvexitet 2. Centralpunktssatsen.

19 juli

Definition. Låt M vara en mängd med n punkter i planet. En punkt X kallas för en *centralpunkt* för denna mängd om varje linje som passerar genom X delar planet i två delar, där varje del (tillsammans med den dragna linjen) innehåller minst $\frac{n}{3}$ punkter från M.

1. Hitta åtminstone en centralpunkt för fyra punkter i planet som bildar en *konvex* respektive *icke-konvex* fyrhörning.

Sats 6.1. *Centralpunktssatsen.* För varje punktmängd i planet så finns det någon centralpunkt.

Låt oss lära oss att konstruera den.
Låt M vara en mängd med n punkter i planet. Betrakta mängden $\mathscr{C}$ av alla konvexa figurer som innehåller mer än $\frac{2n}{3}$ punkter från M.

2. Visa att om alla figurer i $\mathscr{C}$ har en gemensam punkt så är den punkten en centralpunkt.

3. **(a)** Visa att varje par av figurer i $\mathscr{C}$ har en icke-tom konvex figur som snitt.

(b) Visa att det finns en punkt X i skärningen mellan två figurer i $\mathscr{C}$ så att vilken annan figur som helst i $\mathscr{C}$, som inte innehåller X, innehåller punkter som ligger ovanför X.

(c) Visa att alla figurer i $\mathscr{C}$ har en gemensam punkt.

Anmärkning. På så sätt har vi bevisat centralpunktssatsen för planet.

4. Ge ett exempel som visar att uppskattningen $1/3$ i centralpunktssatsen inte kan förbättras.

5. Visa att för varje mängd M av n punkter i planet finns det punkter P och Q i planet så att alla konvexa figurer som innehåller mer än $\frac{4n}{7}$ punkter från M också innehåller P eller Q.

6. Kom på en mindre uppskattning än $\frac{4n}{7}$ i föregående uppgift, men för tre punkter.

7. Det finns 36 punkter på planet. Visa att det finns minst 660 trianglar (bland vilka det kan finnas degenererade trianglar) med hörn i de givna punkterna som innehåller en viss centralpunkt X.

7. Konvexitet 3: Radons sats

20 juli

1. *Carathéodorys sats.* Givna är n punkter i $\mathbb{R}^2$. Visa att om en punkt A tillhör deras konvexa hölje så tillhör det något konvext hölje utav tre av punkterna.

Sats 7.1. *Radons sats.* Givet en mängd P med $d+2$ punkter i $\mathbb{R}^d$ så kan man dela upp punkterna i P i två mängder P_1 och P_2 så att $conv(P_1) \cap conv(P_2) \neq \emptyset$.

2. Bevisa Radons sats:

   (a) för $d = 1$.

   (b) för $d = 2$.

   (c) för $d = 3$.

   (d) generellt.

3. Formulera och bevisa Carathéodorys sats för $\mathbb{R}^d$.

8. Tverbergs sats eller ”flytta lite grann”

21 juli

1. På ett plan finns n punkter givna. Visa att på gränsen av den minsta cirkeln som innehåller alla dessa punkter

(a) finns minst två punkter från mängden.

(b) finns antingen minst tre punkter från mängden eller två punkter från mängden som är diametralt motsatta.

Sats 8.1. *Tverbergs sats.* Antag att vi har $(r-1)(d+1)+1$ punkter i $\mathbb{R}^d$. Då kan vi dela upp dem i disjunkta delmängder $P_1, P_2, \ldots, P_r$ så att de konvexa höljen av dessa delmängder har åtminstone en gemensam punkt.

2. Bevisa följande resultat utifrån påståendet i Tverbergs sats:

(a) Radons sats för $\mathbb{R}^d$.

(b) Centralpunktssatsen för $\mathbb{R}^2$.

(c) Centralpunktssatsen för $\mathbb{R}^d$.

3. Bevisa Tverbergs sats för $\mathbb{R}^2$: En mängd P med $3r-2$ punkter i planet kan alltid delas upp i delmängder $P_1, P_2, \ldots, P_r$ så att de konvexa höljen har en gemensam punkt. Bevisa följande påståenden:

(a) Det finns någon uppdelning av P och en cirkel B som skär alla $conv(P_i)$.

(b) Om det finns en uppdelning av P och en cirkel B från föregående delpunkt så kan punkterna omfördelas mellan mängderna P_i så att varje mängd innehåller högst tre punkter och B fortfarande skär alla $conv(P_i)$. (Från och med nu antar vi att alla mängder P_i innehåller högst tre punkter.)

(c) Om B innehåller inre punkter av alla $conv(P_i)$ så B:s radie minskas (utan att förlora egenskapen att skära alla konvexa höljen).

(d) Antag att cirkeln B endast skär gränspunkterna för vissa $conv(P_{i_1}), \ldots, conv(P_{i_k})$. Om B:s mittpunkt inte ligger inuti konvexa höljet av skärningspunkterna (beröringspunkterna) med dessa mängder så kan B flyttas lite och radien minskas utan att man förlorar huvudegenskapen.

(e) Antag att cirkeln B endast skär gränspunkterna för vissa $conv(P_{i_1}), \ldots, conv(P_{i_k})$. Om B:s mittpunkt ligger inuti konvexa höljet av skärningspunkterna med dessa mängder så kommer åtminstone en av dessa mängder att innehålla tre (eller fler) punkter. (Det finns några särskilda fall där detta inte är så, i så fall omfördela punkterna i P så att cirkeln kan minskas.)

(f) Om B:s mittpunkt ligger inuti konvexa höljet för skärningspunkterna, som i föregående delpunkt, så ligger det inuti konvexa höljet för åtminstone tre av skärningspunkterna. Då kan punkterna omfördelas mellan mängderna $P_{i_1}, \ldots, P_{i_k}$ så att åtminstone ett av de konvexa höljen skär B i inre punkter.

(g) Om radien för B är positiv kan den minskas samtidigt som den fortfarande skär alla $conv(P_i)$. Då finns det en uppdelning av P där B är bara en punkt.

4. **Kirszbrauns sats in $\mathbb{R}^2$.** Låt $P_1, \ldots, P_r$ vara mängder med punkter i planet så att skärningen av deras konvexa höljen är icke-tom. Då finns det delmängder $P_1' \subset P_1, \ldots, P_r' \subset P_r$ med totalt $3r-2$ punkter så att alla $conv(P_i')$ har åtminstone en gemensam punkt.

5. Generalisera beviset av Tverbergs sats till $\mathbb{R}^d$.

6. **Carathéodorys sats med flera färger.** Antag att A_1 är en mängd blå punkter, A_2 är en mängd röda punkter och A_3 är en mängd gröna punkter i planet. Dessa mängder har minst en gemensam punkt X i sina konvexa höljen. Visa att X tillhör minst en triangel med olikafärgade hörn.

9. Mängdlära 1 - Välordningar

23 juli

Definition. En *välordning* är en total ordningsrelation där varje delmängd har ett minsta element.

Definition. Två ordnade mängder $(X, <_1), (Y, <_2)$ sägs vara *ordningsisomorfa* (eller bara *isomorfa*) om det finns en bijektiv funktion $f : X \to Y$ så att

$$x <_1 y \iff f(x) <_2 f(y)$$

1. Visa att en ordning är en välordning om och endast om det inte finns några oändliga följder $x_1 > x_2 > x_3 > \dots$

2. Givet två välordnade mängder $(A, <), (B, <)$, hur kan man göra en välordning av $A \times B$?

3. Är mängden av alla ändliga strängar (av bokstäver) välordnad av den vanliga ordboksordningen?

4. a) Visa att $\mathbb{R}$ är ordningsisomorf med (0,1).

   b) Visa att (0,1) inte är isomorf med [0,1).

   c) Är $\mathbb{Q}$ isomorf med $\mathbb{Q} \setminus \{0\}$?

5. (*) Visa att en välordning är uppräknelig om och endast om den är isomorf med en delmängd av $\mathbb{R}$

Definition. Ett *initialsegment* av en ordnad mängd $(X, <)$ är en mängd på formen $\{x \in X : x < a\}$ för något element $a \in x$. Notera att mängden X själv inte är ett initialsegment.

Det finns flera standarder för hur initialsegment är betecknade, t.ex $a^< = \{x \in X : x < a\}$

6. Visa att ingen välordnad mängd kan vara isomorf med ett initialsegment av sig själv.

7. (*) Visa att givet två välordnade mängder så är de antingen isomorfa eller så är den ena isomorf med ett initalsegment av det andra

Axiom **Urvalsaxiomet.** Alla mängder av icke-tomma mängder har en urvalsfunktion. Dvs för alla A som bara innehåller icke-tomma mängder så existerar en funktion $f : A \to \cup_{x \in A} x$ så att $f(x) \in x$ för alla $x \in A$.

8. (*) Visa att påståendet “Alla mängder har en välordning.“ är ekvivalent med urvalsaxiomet.

9. Det går att göra stark induktion på godtyckliga välordnade mängder. Hur?

10. (*) Visa att varje ordnad mängd kan färgas i två färger så att mellan varje par av punkter i samma färg finns en punkt i den andra färgen.

11. Oändligt många personer har blivit tillfångatagna! De måste klara följande utmaning för att bli frigivna: Varje person kommer få en hatt på sig i en av k färger. De kan se vilka färger alla andra har på sina hattar, men inte färgen av sin egen hatt. De ska samtidigt gissa vilken färg de har på hatten, och de blir endast frigivna om bara ändligt många av dem har fel. Vad ska deras strategi vara för att lyckas?

10. Mängdlära 2 - Ordinaltal

24 juli

Ordinaltal är ett sätt att klassificera på vilka sätt välordningar kan se ut. Betrakta följande sätt att ordna $\mathbb{N}$

1. $A = \{1,2,3\ldots\}$
2. $B = \{2,3,4\ldots\} \cup \{1\}$
3. $C = \{3,4,5\ldots\} \cup \{1,2\}$

Där vi tänker oss att ordningen går från vänster till höger. Mängden som ordnas är samma varje gång, men ordningarna är inte isomorfa med varandra.

1. Hur många olika (inte isomorfa) sätt kan du ordna $\mathbb{N}$ på? Hitta så många som du kan!

Definition. Ett ordinaltal representerar ett sätt att välordna mängder på.
Ordningen med ett element brukar representeras med siffran 1. Ordningen med två element med siffran 2 osv. Den vanliga ordningen på de naturliga talen (mängd A ovan) betecknas ω.

2. Hur tycker du man borde beteckna mängderna B och C ovan? Skriv en lämplig ordinal för de välordningar du gjorde i problem 1.

Sats 10.1. Givet två välordningar X, Y så är de antingen isomorfa eller så är den ena isomorf med ett initialsegment av den andra.

Med hjälp av detta kan vi jämföra ordinaler

Definition. Givet två ordinaler α och β säger vi att $\alpha < \beta$ om α är ett initialsegment av β. (Upp till isomorfism)

3. Bekräfta med sats 2.1 att för alla ordinaler α och β så gäller ett av påståenden $\alpha < \beta, \beta < \alpha$ eller $\alpha = \beta$.
4. Givet en mängd med ordinaler, visa att det finns en ordinal som är större än alla dem, och beskriv hur man hittar denna.
5. Visa att ordinalerna är välordnade. Det vill säga, givet en mängd med ordinaler, beskriv hur man kan hitta den minsta av dem.
6. Visa att det inte kan finnas en oändlig minskande följd av ordinaltal.
7. Givet ordinaler α och β, definiera

a) $\alpha + \beta$

b) $\alpha \cdot \beta$ (Ledning: Problem 2 från igår)

c) α^{β} (Svårare)

8. Är $\omega + 1 = 1 + \omega$? Är $2 \cdot \omega = \omega \cdot 2$? Skriv ner några intressanta ordinaluttryck och se om du kan förstå hur de ser ut.

OBS Förhoppningsvis kom du fram till att $2 \cdot \omega \neq \omega \cdot 2$, och att den ena är större än den andra. I standardbeteckning så är faktiskt $\omega \cdot 2$ den större av de två (vilket i alla fall jag inte tyckte var intuitivt). Vi har även att $\omega + 1 > 1 + \omega$

Problemlösning

9. Betrakta mängden $\mathbb{Q}^2$ av rationella punkter i planet. Visa att det finns en delmängd av dem så att alla linjer i $\mathbb{Q}^2$ skär denna mängd exakt 2 gånger.

10. Går det att färga punkterna i $\mathbb{Q}^2$ i n olika färger så att alla linjesegment med rationella ändpunkter har punkter med alla n färger på sig?

11. Visa sats 10.1 genom att använda följande bevismall:

a) Betrakta mängden $f = \{(x,y) \in X \times Y : x^{<} \text{ är isomorf med } y^{<}\}$. Visa att för alla x existerar högst ett y så att $(x,y) \in f$, och symmetriskt att det för varje y existerar högst ett x

b) Visa att f kan tolkas som en bijektiv funktion från en delmängd av x till en delmängd av y.

c) Visa att denna funktion bevarar ordning och därmed är en isomorfism.

d) Visa att definitionsmängden och värdemängden av f inte båda kan vara initialsegment.

e) Dra slutsatsen att sats 10.1 stämmer

12. Visa att urvalsaxiomet är ekvivalent med påståendet "Alla mängder kan välordnas" genom använda följande bevismall:

a) Visa att om alla mängder kan välordnas så gäller urvalsaxiomet genom att explicit konstruera en urvalsfunktion. Konstatera att det nu räcker att visa att godtyckliga mängder kan välordnas givet urvalsaxiomet.

b) Låt X vara en godtycklig mängd och f vara en urvalsfunktion på mängden $\mathscr{P}(X)$ av alla delmängder av X. Kalla en delmängd $A \subseteq X$ trevlig om det finns en välordning av A så att

$$\forall x \in A, \quad x = f(X \setminus x^{<})$$

Försök få en intuition för vad denna definition betyder, och varför det kanske kommer hjälpa dig att visa satsen.

c) Visa att om A är trevlig så är $A \cup \{f(X \setminus A)\}$ också trevlig.

d) Visa att om två trevliga mängder är isomorfa så är de lika.

e) Betrakta unionen av alla trevliga mängder och visa att denna är X.

13. Fråga Sebastian om fler problem, eller lös problem du inte hann med igår.

11. Mängdlära 3 - Transfinit induktion

25 juli

Idag ska vi undersöka hur man kan göra induktion på mängder som är större än $\mathbb{N}$

Sats 11.1. Varje mängd kan välordnas.

1. Visa att för alla ordinaler α, så gäller ett av följande två fall
   a) $\alpha = \beta + 1$ för någon ordinal β. (α kallas följdordinal)
   b) $\alpha = \bigcup_{\beta<\alpha} \beta$ (α kallas gränsvärdesordinal)

Sats 11.2. **Transfinit induktion** Låt $P(\omega)$ vara ett påstående som beror på ett ordinaltal. Om du har visat att

1. $P(0)$ gäller
2. om $P(\beta)$ gäller för alla $\beta < \alpha$ så gäller även $P(\alpha)$

Då gäller $P(\alpha)$ för alla ordinaler α

Ofta behöver man skilja på följdordinaler och gränsvärdesordinaler i problem 1 när man gör transfinit induktion.

2. Visa sats 11.2

Definition. Låt A vara en mängd. *As kardinal* är den minsta ordinalen α så att det finns en bijektion från A till α.

3. Visa att det går att definiera en ordning på A så att A är isomorf med sin kardinal.

Problemlösning

4. Visa att alla ordinaler kan skrivas som summan av en ändlig ordinal och en gränsvärdesordinal på ett unikt sätt.

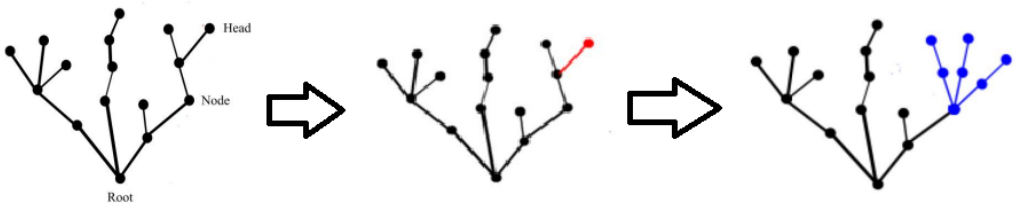
5. Visa att det finns en delmängd $S \subseteq \mathbb{Q}^2$ så att alla linjer i $\mathbb{Q}^2$ skär S i exakt två punkter.

6. Visa att det finns en delmängd $S \subseteq \mathbb{R}^2$ av planet så att alla linjer i $\mathbb{R}^2$ skär S i exakt två punkter.

7. Visa att det finns en delmängd $V \subseteq \mathbb{R}$ så att för alla $x \in \mathbb{R}$ finns ett **unikt** $v \in V$ så att $x - v$ är rationellt.

8. Visa att alla funktioner $f : \mathbb{R} \to \mathbb{R}$ kan skrivas som en summa av två injektiva funktioner.

9. Herkules slåss mot en hydra, som består av ett ändligt träd med en rot R. Varje drag väljer herkules ett av hydrans huvud (ett av trädets löv) och hugger av det. Då kommer huvudets farfarsnod att växa ut n kopior av subträdet som blev attackerat. Visa att Herkules vinner stiden mot hydran på ändligt antal drag, oavsett hur han gör.



Figur 11.1: Ett exempel på ett drag Herkules kan göra. Här är $n = 2$

10. Givet ett heltal $n \geq 2$ genererar vi $h(n)$ på följande vis: Låt r vara entalssiffran i n.

1. Om $r = 0$ så ges $h(n)$ av att ta bort denna siffra 0. T.ex $h(120) = 12$.
2. Om $1 \leq r \leq 9$. Delar vi in decimalrepresentationen av n i en maximal högra del R som bara innehåller siffror $\geq r$, och en vänstra del L som antingen är tom eller slutar med en siffra $< r$. Då består decimalrepresentationen av $h(n)$ av L följt av två kopior av $(R-1)$. Till exempel, för talet 171543 så har vi $L = 171$, $R = 543$ och $h(n) = 171344344$

Visa att för $n \geq 2$ så kommer talföljden $n, h(n), h(h(n))...$ att nå talet 1 efter ändligt många applikationer av h.

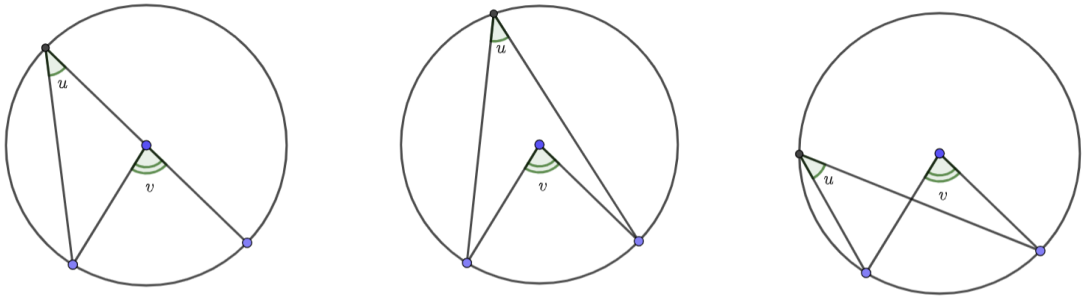
12. Vid gymnasiemattens slut

23 juli

Matematik 2c, fast ordentligt.

Tävlingstips 1 När ett nytt geometriproblem ser ut lite som ett gammalt, men bara lite, lägg till det som behövs att den gamla bilden ska synas tydligt i den nya.

1. **— Medelpunktsvinkelsatsen.** Bevisa att $v = 2u$ i alla fall i Figur 12.1.



(a) (b) (c)

Figur 12.1: Olika fall av medelpunktsvinkelsatsen.

2. **— Randvinkelsatsen.** Bevisa randvinkelsatsen (Figur 12.2a).

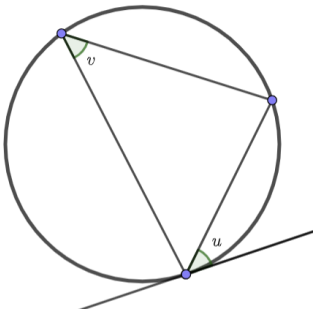
(a) Randvinkelsatsen säger att $u = v$.

(b) Omvändningen säger att punkterna ligger på en cirkel.

Figur 12.2: Randvinkelsatsen och dess omvändning.

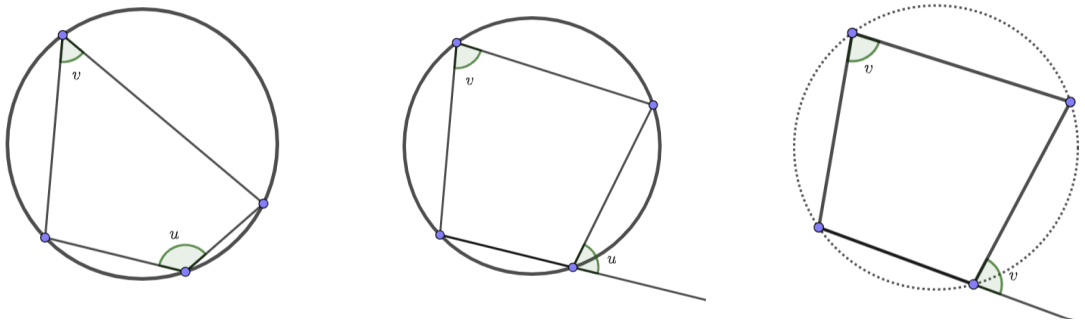
3. **— Omvändningen till randvinkelsatsen.** Bevisa omvändningen till randvinkelsatsen (Figur 12.2b).

4. **— Korda-tangentsatsen.** Bevisa korda-tangentsatsen (Figur 12.3).



Figur 12.3: Korda-tangentsatsen säger att $u = v$.

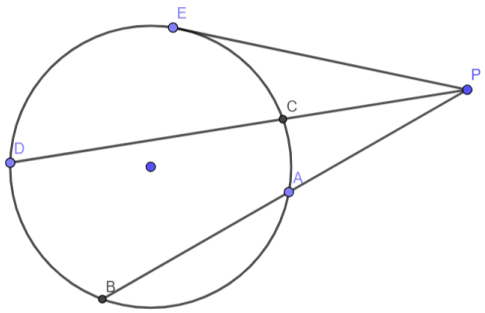
5. — Satsen om cykliska fyrhörningar. Bevisa satsen om cykliska fyrhörningar och dess omvändning (Figur 12.4).



(a) $u + v = 180°$ (b) $u = v$ (c) Fyrhörningen är cyklisk.

Figur 12.4: Satsen om cykliska fyrhöringar och dess omvändning.

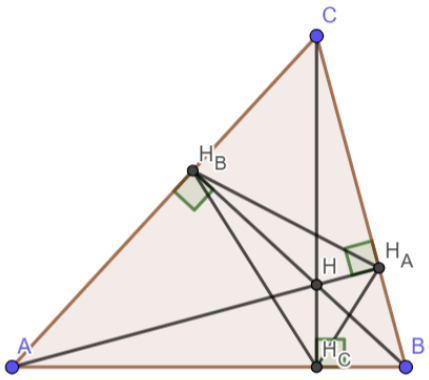
6. — Kordasatsen och dess omvändning. Bevisa kordasatsen, vars ena fall visas i Figur 12.5. Vad säger den när P ligger inuti eller på cirkeln? Kan du formulera och bevisa dess omvändning?



Figur 12.5: Kordasatsen säger att $PA \cdot PB = PC \cdot PD = PE^2$.

På jakt efter cykliska fyrhörningar

Tävlingstips 2 När du ser ett geometriproblem, särskilt ett som handlar om vinklar, börja alltid med att leta efter cykliska fyrhörningar.



Figur 12.6: I en triangel ABC, låt H_X vara skärningen mellan höjden från X och förlängningen av den motstående sidan. Höjderna skär varandra i en punkt (se Problem 9) som kallas ortocentrum.

7. Hitta sex cykliska fyrhörningar i Figur 12.6.

8. **— Pedaltriangeln.** Uttryck vinklarna i triangel $H_AH_BH_C$ med hjälp av vinklarna i triangel ABC. Triangel $H_AH_BH_C$ kallas pedaltriangeln. Vilken roll har H i pedaltriangeln?

9. **— Ortocentrum.** Bevisa att höjderna i en triangel skär varandra i en punkt.

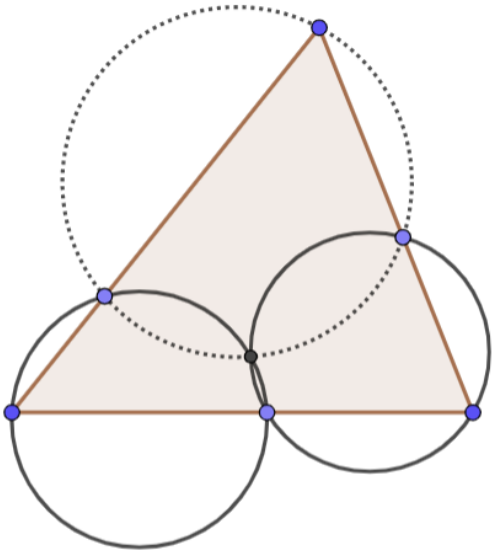
10. **— Miquels sats.** Bevisa satsen i Figur 12.7.

11. (SMT-kval 2011, problem 4) I fyrhörningen $ABCD$ är $AC = 2BC$. Vidare gäller $\angle ABD = \angle DBC = \angle DAC$. Man vet att $\angle ADC = 90°$. Bestäm fyrhörningens övriga vinklar.

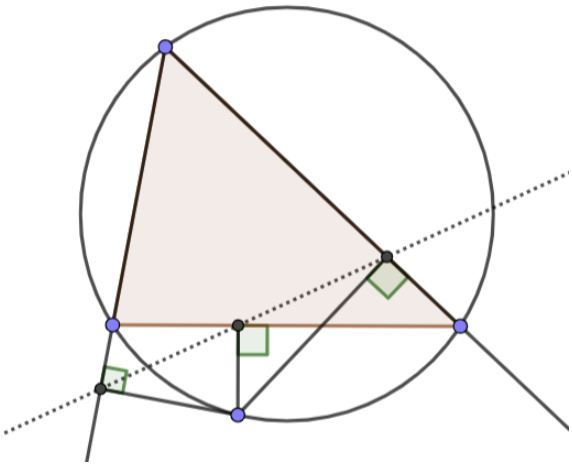
12. I triangeln ABC dras medianerna AD och BE. Visa att om $\angle CAD = \angle CBE$, så är $AC = BC$.

13. Låt $ABCD$ vara en cyklisk fyrhörning vars diagonaler är vinkelräta mot varandra. Låt E vara diagonalernas skärningspunkt och M vara mittpunkten på AB. Visa att linje ME är vinkelrät mot CD.

14. **— Simsons linje.** Bevisa satsen i Figur 12.8. Gäller omvändningen?



Figur 12.7: I en triangel ABC, låt D, E och F vara punkter på sidorna BC, AC, respektive AB. Låt de omskrivna cirklarna till trianglarna AEF och BDF skära varandra i punkten P. Bevisa att den omskrivna cirkeln till triangel CDE också går genom P.

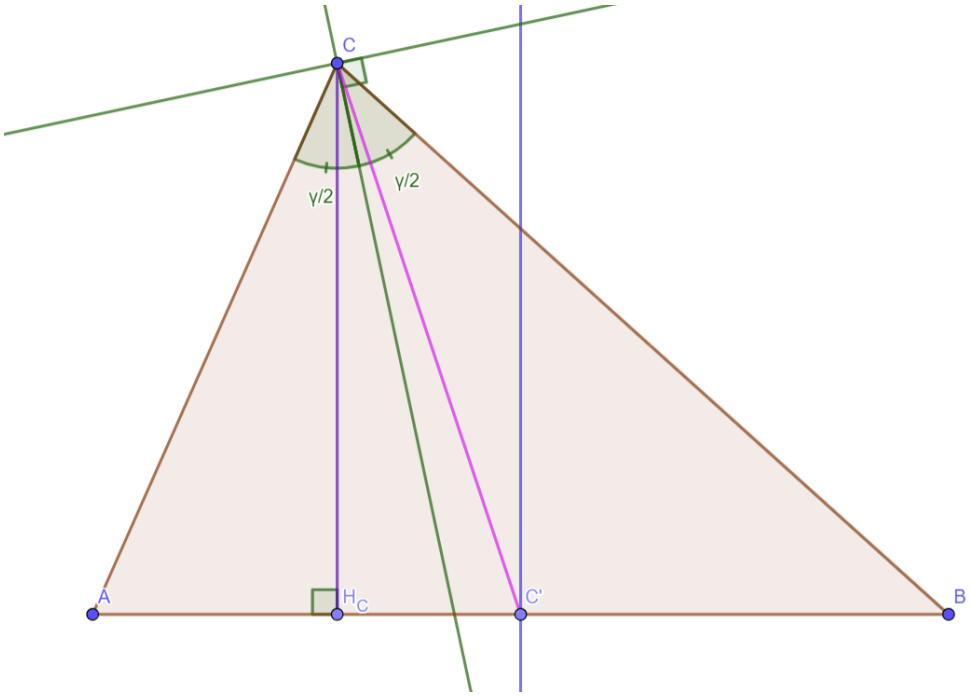


Figur 12.8: Om P ligger på triangel ABCs omskrivna cirkel så ligger Ps ortogonalprojektioner på sidornas förlängningar på en rät linje.

13. Triangelns anatomi

24 juli

Triangelns kroppsdelar



Figur 13.1: Några olika linjer i en triangel. I lila har vi höjden, i skrikrosa medianen, i blått har vi mittpunktsnormalen, och i grönt har vi den inre och den yttre bisektrisen.

Definition. — **Mittpunktsnormal.** Låt AB vara en sträcka. Dess mittpunktsnormal är orten för de punkter som har lika långt avstånd till A som till B.

1. Bevisa att mittpunktsnormalen till AB är den vinkelräta linjen till linje AB som går genom ABs mittpunkt.

2. Givet två linjer l_1 och l_2 som inte är parallella, hitta orten för de punkter som ligger lika långt ifrån l_1 som l_2.

Definition. — **Inre och yttre bisektriser.** Låt ABC vara en vinkel. Den inre bisektrisen till ABC är en linje som delar vinkeln mitt itu. Den yttre bisektrisen är linjen genom B som är vinkelrät mot bisektrisen.

Definition. — **Höjd.** I en triangel är en höjd en linje som går genom ett hörn och

är vinkelrät mot förlängningen av hörnets motstående sida.

3. Rita en triangel ABC där $\angle ABC$ är trubbvinklig. Rita höjden från A.

Definition. — **Median.** I en triangel är en median sträckan mellan ett hörn och mittpunkten på den motstående sidan.

4. Låt ABC vara en triangel. Visa att om två av följande linjer sammanfaller så är $AB = AC$.

a) Höjden från A

b) Medianen från A

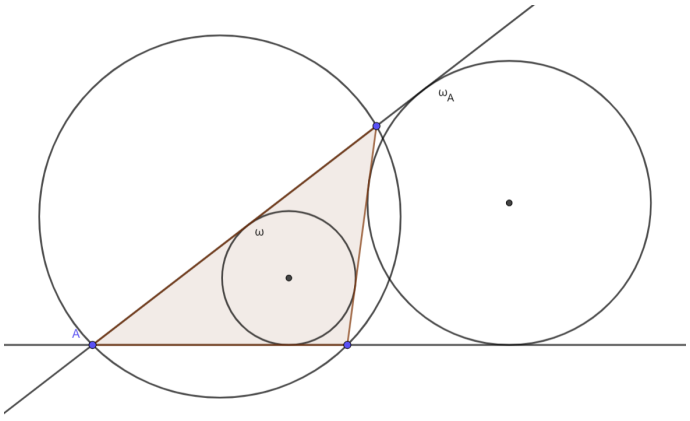
c) Bisektrisen till $\angle CAB$

5. — **Extra.** Bevisa att bisektrisen alltid ligger mellan höjden och medianen. (Ledtråd: Vad var det bisektrissatsen sa, nu igen?)

Definition. En omskriven cirkel till en triangel är en cirkel som går igenom alla triangelns hörn.

Definition. En inskriven cirkel till en triangel är en cirkel som tangerar alla triangelns sidor.

Definition. En vidskriven cirkel till en triangel är en cirkel som tangerar en sida och förlängningarna av de andra två sidorna.



Figur 13.2: En triangel och dess omskrivna cirkel, dess inskrivna cirkel och en av dess vidskrivna cirklar.

6. Givet en triangel ABC, konstruera dess omskrivna cirkel. (Ledtråd: Vad är en cirkel, nu igen?)

7. Givet en triangel ABC, konstruera dess inskrivna cirkel.

8. Givet en triangel ABC, konstruera dess vidskrivna cirklar.

Triangelns kroppsdelar kommer i trior

För nästintill varje speciell typ av linje i en triangel som det finns tre av, så skär dessa tre varandra i en punkt. I detta kapitel ska vi visa det.

Tävlingstips 3 För att visa att tre linjer går genom en punkt, börja med att markera skärningspunkten P mellan två av linjerna. Vi vill visa att den tredje linjen l går genom den punkten också. Det finns flera olika tekniker för det.

a) Om l kan beskrivas som mängden/orten av de punkter som har egenskap X, så kan vi försöka visa att P har egenskap X.

b) Om l definieras av någon egenskap kan man dra en ”liknande” linje l' genom P och visa att l' har samma egenskap som l och alltså är samma linje.

Sats 13.1. I en triangel ABC skär de tre sidornas mittpunktsnormaler varandra i en punkt. Denna punkt är medelpunkten för dess enda omskrivna cirkel, betecknas oftas med O och kallas ofta ”omcentrum”.

Sats 13.2. I en triangel ABC skär de tre vinklarnas inre bisektriser varandra i en punkt. Denna punkt är medelpunkten för dess enda inskrivna cirkel, betecknas oftast med I och kallas oftast ”incentrum”.

Sats 13.3. I en triangel ABC skär $\angle A$s inre bisektris och de två andra vinklarnas yttre bisektriser varandra i en punkt. Denna punkt är medelpunkten för en av dess vidskrivna cirklar och betecknas oftast med I_A.

9. Bevisa ovanstående satser.

Sats 13.4. Höjderna i en triangel skär varandra i en punkt. Denna punkt kallas ortocentrum och betecknas oftast med H. Jag har hört goda argument för att den borde kallas ”höjdpunkten”.

10. Bevisa Sats 13.4. (Ledtråd: Gör uppgift 7 och kanske 8 från igår först.)

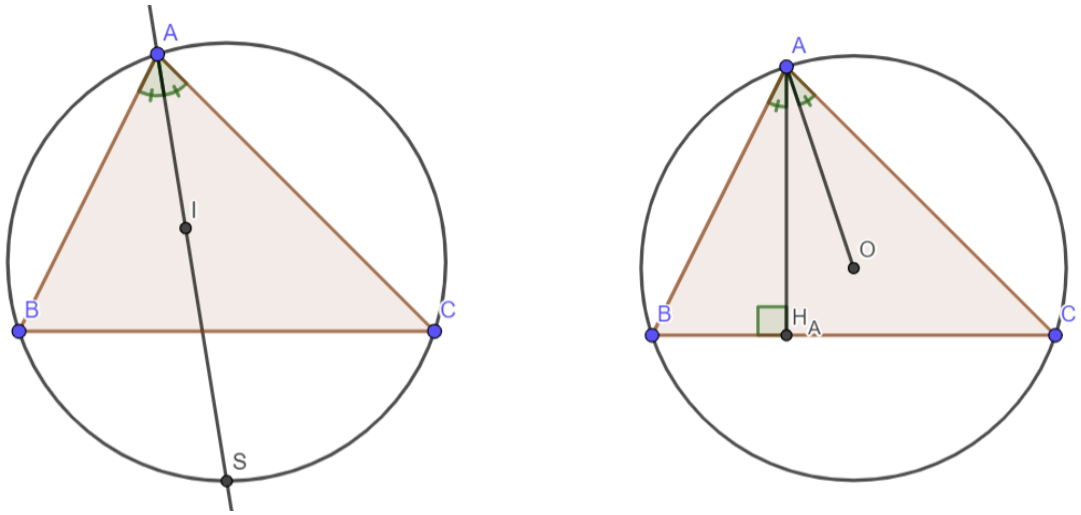
Sats 13.5. Medianerna i en triangel skär varandra i en punkt. Denna punkt kallas masscentrum/tyngdpunkten och betecknas oftast med M. Dessutom delar M varje median i förhållandet $2:1$.

11. **— Extra.** Bevisa Sats 13.5. (Ledtråd: Det speciella med medianer är att de delar

triangeln i två delar med lika stor area.)

Hur olika kroppsdelar samverkar

12. — **Superlemma.** Lös problemet i Figur 13.3a.

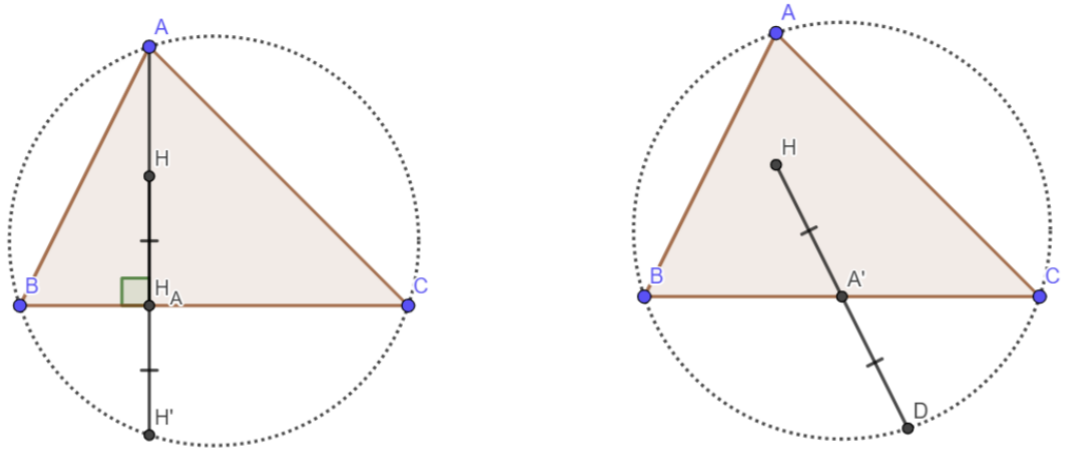


(a) Visa att $BS = IS = CS$.

(b) Visa att $\angle BAH_A = \angle OAC$.

Figur 13.3

13. Lös problemet i Figur 13.3b.



(a) Visa att speglingarna av H i varje sida ligger på triangelns omskrivna cirkel.

(b) Visa att speglingarna av H i varje sidas mittpunkt ligger på triangelns omskrivna cirkel.

Figur 13.4

14. Lös problemet i Figur 13.4a.
15. Lös problemet i Figur 13.4b.

16. **— Det ryska problemet, svårt.** Låt ABC vara en spetsvinklig triangel med omskriven cirkel ω och omcentrum O. Beteckna med H triangelns ortocentrum och med H_B och H_C fötterna till höjderna från B respektive C till AC respektive AB. Låt D vara skärningspunkten mellan sträckorna AH och H_BH_C. Låt M vara sträcka BCs mittpunkt. Låt N vara skärningspunkten mellan linje AO och BC. Visa att $DN||HM$.

IV Programmering - Ada

1. Python Intro 1

18 juli

1.1 Datatyper

Det här är några sätt man kan deklarera variabler med olika datatyper:

```
x = 5                                       # Integer
y = 3.1                                     # Float
z = "Text"                                  # String
bl = True                                   # Bool
lst = [9, False, "mango", ["?", 0]]         # List
dct = {"key1":"value1", "key2":"value2"}    # Dictionary
st = {"banan", 18, "kiwi"}                  # Set
tpl = (5, "tre", 9)                         # Tuple
```

1.2 Symbolhanterande

Testa att lagra följande värden i olika variabler, och kör koden. Vad händer?

1. Integer + Integer?

```
a = 3 + 5
print(a)
b = 2
print(a+b)
```

2. String + String?

```
a = "ost" + "macka"
print(a)
```

3. Integer + String?

```
a = 9 + "smörgåsar"
print(a)
```

4. String * Integer?

```
a = "Echo" * 3
print(a)
```

5. Integer * Bool? (Testa även Integer + Bool!)

```
a = 9 * False
print(a)
```

6. List * Integer

```
a = [1, 2, 3] * 4
print(a)
```

7. List + List

```
a = [3, 1, 4] + [1, 5, 9]
print(a)
```

1.3 Indexering

8. Vad händer när man kör koden nedan? Gör några egna gissningar först innan ni kör koden. Reflektera sedan varför det blev som det blev.

```
x = "Hello world"
print(x[7])
print(x[1:3])
print(x[-2])
print(x[3:])
print(x[:]) # När vill man använda detta?
print(x[:1])
print(x[:-3])
print(x[-5:])
```

9. Vad händer när man kör koden nedan med $print(a)$ efter varje rad? Gör några egna gissningar först innan ni kör koden. Reflektera sedan varför det blev som det blev.

```
a = [2,3]
a[1] = 5
a[1] = [3,4,5]
a[2] = [7]
a[2:] = [7]
a[2:] = [7,8,9,10]
a[2:] = [3,4]
a[:-3] = [0]
```

10. Vad innebär indexen i och j när man skriver $[i, j]$?

1.4 Indata/Input

För att ta in input från en användare kan man använda funktionen:

```
input()
```

Ifall man vill att programmet skriver ett meddelande innan den frågar om input, kan man göra så här:

```
namn = input("Skriv ditt namn här: ")
print("Hej " + namn + "!")
```

11. Skriv ett program som ber användaren om 2 tal, utför ett par beräkningar och sedan skriver ut svaret.

1.5 Funktioner

Ibland nöjer vi oss inte med de inbyggda funktionerna i programmeringsspråket man använder, och då kan vi definiera våra egna funktioner:

```
def add(a, b):
    return a + b
```

12. Implementera någon funktion som tar ett eller flera argument och returnerar ett lämpligt svar. Några förslag:

1. En funktion som kan räkna ut antal bananer du kan köpa ifall du har X kronor, och varje banan kostar 5 kr.
2. En funktion som konverterar antal grader i Fahrenheit till Celsius, eller tvärtom. (Googla gärna hur man räknar ut Celsius med hjälp av Fahrenheit!)
3. En funktion som konverterar längder i fot och tum till centimeter.

1.5.1 Rekursion

Rekursion innebär att en funktion kan kalla på sig själv.

```
def rec(count)
    if count > 0:
        print('There are ' + count + 'function calls left!')
        return rec(count - 1)
```

Skapa en egen rekursiv funktion. Exempel:

1. n! (n-fakultet)
2. fibonacci-talen
3. Ackermann-funktionen (Googla!)

Extra: Lägg till en variabel som räknar hur många rekursiva anrop som görs till funktionen. Finns det smartare sätt att implementera din funktion som inte kräver lika många anrop?

1.6 Omvandling mellan datatyper

13. Hur kan man konvertera en variabel mellan string, integer och float? Vilka begränsningar finns?

1.7 Extra: Klasser

Python används ofta för att skriva objektorienterad kod. Detta innebär att man kan skriva egna klasser, med egna medlemsfunktioner. Implementera en egen klass! Ett exempel finns nedanför:

```
class Bus:
    def __init__(self, ID, color, passengers):
        self.ID = ID
        self.color = color
        self.passengers = passengers

    def add_passengers(self, n):
        self.passengers += n

    def remove_passengers(self, n):
        self.passengers -= n

    def info(self):
        print('Bus ID: ' + self.ID + '\n' + 'Color: ' + self.color + '\n' +
            'Current number of passengers: ' + str(self.passengers))

bus1 = Bus('ABC123', 'green', 5)
bus1.info()
bus1.add_passengers(3)
bus1.info()
```

1.8 Fler instruktioner och exempel

`https://docs.python.org/3/tutorial/introduction.html`

2. Python Intro 2

19 juli

2.1 For-loopar

När man vill repetera en liknande process i kod ett antal gånger, så brukar man använda sig av loopar. Andra anledningar till att man använder loopar är för att spara tid och för att minska misstag.
En vanlig loop är en For-loop.

1. Testa runt med följande kod, och svara på frågorna.

```
for number in range(5):
    print(number)
```

(a) Hur många gånger loopas print-funktionen? Alltså, hur många gånger kommer variablen *number* att printas?

(b) Vilka tal printas?

(c) Vad är $range(5)$?

(d) Skapa en for-loop som printar de 5 första udda talen.

2. Testa runt med följande kod, och svara på frågorna.

```
a = ["Harry", "Elias", "Tobias", "Fredrik"]
for namn in a:
    print(namn)
```

(a) Hur många gånger loopas print-funktionen här? Vad beror det på?

(b) I vilken ordning printas namnen?

(c) Hur kan man printa samma namn i samma ordning genom att använda

```
for i in range(len(a)):
```

?

2.2 While-loopar, Break och Continue

3. Testa runt med följande kod, och svara på frågorna.

```
a = 0
while a<10:
    print(a)
    a += 1
```

(a) Hur många gånger loopas print-funktionen?

(b) Vilka tal printas?

(c) Vad innebär olikheten efter *while*?

4. Testa runt med följande kod, och svara på frågorna.

```
a = 0
while 1:
    if a == 10:
        continue
    elif a == 20:
        break
    else:
        a += 1
    print(a)
```

(a) Hur många gånger loopas print-funktionen?

(b) Vilka tal printas?

(c) Testa runt och ändra på siffrorna för att lista ut vad *continue* och *break* gör.

2.3 Lägga till element i listor och sets

Att lägga till och ta bort element i listor och sets använder sig av olika sorters funktioner i Python. Till exempel:

```
frukter = ["banan", "kiwi"]
frukter.append("druvor")
frukter.pop()   #tar bort det sista elementet
frukter.pop(1)  #vad innebär siffran här?

harrys_favoritsiffror = {1,2,0}
harrys_favoritsiffror.add(7)
harrys_favoritsiffror.remove(1)
```

5. Givet en lista av positiva heltal där det kan finnas dubbletter, skriv en kod som kan printa ut alla tal som uppstår i listan (alltså inga dubbletter).

6. Givet en lista av positiva heltal, skriv en kod som räknar ut antalet möjliga summor man kan skapa genom att addera 2 tal från listan.

Python har flera olika sätt att med kort och koncis kod skapa de objekt man vill. Till exempel finns list comprehension, set comprehension och dictionary comprehension. Kolla upp vad dessa innebär. Ser du varför den här notationen är användbar?

2.4 Övningar listor

Använd list comprehension (listbyggare) för att lösa följande uppgifter:

1. Skapa en lista med alla kvadrattal mellan 1 och 100.
2. Skapa en lista med alla tal mellan 1 och 100 som innehåller siffran 6.
3. Utgå från listan ['Björn', 'Hund', 'Katt', 'Häst']. Lägg till ett utropstecken till alla strängar och filtrera bort strängar som inte innehåller bokstaven 'n'.

2.5 Övningar ordböcker

Använd dictionary comprehension (ordboksbyggare) för att lösa följande uppgifter:

1. Skapa en dictionary vars nycklar är talen från 1 till 15 (båda inkluderas) och värdena är kvadraten av nycklarna. Testa först med en for-loop och sen med en ordboksbyggare.
2. Skapa en dictionary med alla orden i denna delfråga där orden är nyckeln och ordens längd är dess värden. Använd ordboksbyggare.
3. Använd nästlade list / dictionary comprehensions för att hitta största ensiffriga delaren för varje tal från 1 till 1000. Talen är nyckel och största ensiffriga delaren är värdet.

2.6 Problemlösning! (Whoaaa!)

7. Lös följande uppgift: https://projecteuler.net/problem=9
8. https://projecteuler.net/problem=14
9. https://projecteuler.net/problem=25

   Fler problem finns här: https://projecteuler.net/archives

3. Spelprogrammering 1

20 juli

Idag ska vi börja med spelprogrammering. Vi kommer att använda python-biblioteket `pygame` för att rita saker på skärmen och ta input från tangentbordet och musen. `pygame` kan man använda för allt från att göra allt från spel till simuleringar.

Idag kommer vi fokusera på grunderna, hur spel är uppbyggda, hur man använder `pygame` och sist återskapa det klassiska spelet Pong.

Spel generellt

1. Gameloop

Många spel är uppbyggda på väldigt liknande sätt, det är en initieringsdel i början av koden, här sker allt som bara ska ske en gång när programmet startar. Alla varibler och objekt kan skapas, bilder och ljudfiler kan laddas in och vi skapar ett fönster.

Sedan kommer gameloopen, en `while`-loop som kör om och om igen tills vi stänger programmet. Den kör en gång för varje uppdatering spelet gör. Vi läser in input från användaren, uppdaterar påverkade objekt och ritar allt.

Sist rensar vi upp efter oss, tar bort objekt och stänger fönstret.

```
# --- Initiering / setup ---
#  skapa alla variabler och objekt man behöver senare under spelets gång,
    ladda in bilder, osv

running = True
while running:
    # --- Events ---
    #  ta input från användaren, har man klickat med musen, tryckt ner eller
        släppt på en tangent
    # --- Updates ---
    #  uppdatera alla variabler och objekt. Flytta till exempel spelaren
        frammåt om den håller ner 'W'.
    # --- Render ---
    #  rita allt på skärmen
    ...

# --- Cleanup ---
#  gör allt man vill göra precis innan spelet stängs, radera objekt i minnet,
    spara
```

Hur man använder pygame

Vi vill nu använda `pygame` för att skapa ett fönster, ta input från användaren och börja rita saker.

2. Öppna ett fönster

```
# --- Initiering / setup ---
import pygame # Importera pygame biblioteket

pygame.init() # Starta allt internt i pygame

# Välj storlek på fönstret med bredd (800 pixlar) och höjd (600 pixlar).
screen = pygame.display.set_mode([800, 600])
pygame.display.set_caption("Namn på fönstret")

running = True
while running:
    # --- Events ---
    # --- Updates ---
    # --- Render ---
    screen.fill((255, 255, 255)) # Fyll skärmen med vit

    # Uppdatera det som visas
    pygame.display.flip()

# --- Cleanup ---
# Rensa allt som skapats vid init()
pygame.quit()
```

3. Fönstret går i nuläget bara att stänga genom att skicka *CTRL+C* i terminalen. För att programmet ska stoppa när man trycker på stängknappen måste det kodas. Allt användaren gör, tex trycker på tangenter eller klickar på stängknappen läggs till som ett event som koden kan reagera på.

```
    # --- Events ---
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False
```

4. Rita former

För att göra klassiska spel räcker det med att kunna rita rektanglar och cirklar. `pygame` har inbyggt att rita massa olika former och linjer.

```
    # --- Render ---
    screen.fill((255, 255, 255))

    pygame.draw.rect(screen, (255, 0, 255), (100, 200, 500, 300))
    # Rita en lila rektangel (röd = 255, grön = 0, blå = 255), men övre hörnet
        i punkten (100, 200) (100 från vänster, 200 uppifrån) som är 500x300
        pixlar stor
```

```
pygame.draw.circle(screen, (255, 255, 0), (550, 300), 80)
# Rita en gul cirkel (röd = 255, grön = 255, blå = 0), med centrum i
    punkten (550, 300) och radie 80.

pygame.draw.line(screen, (255, 0, 0), (0, 0), (800, 400), width = 20)
# Rite en röd linje (röd = 255, grön = 0, blå = 0) från punkten (0, 0) till
    punkten (800, 600).

pygame.display.flip()
```

`pygame` kan rita många fler former och polygoner, alla olika `draw`-funktioner finns här: `https://www.pygame.org/docs/ref/draw.html`.

Notera också vilka av rektangeln, cirkeln och linjen som hamnar ovanpå varandra. Testa byta plats på två rader och se vad som händer.

5. Interaktivitet

(a) Testa om en tangent blivit nedtryckt eller uppsläppt:

```
# --- Event ---
for event in pygame.event.get():
    if event.type == pygame.QUIT:
        running = False
    elif event.type == pygame.KEYDOWN:
        if event.key == pygame.K_a:
            print("A was pressed this frame")
    elif event.type == pygame.KEYUP:
        if event.key == pygame.K_a:
            print("A was released this frame")
```

Alla olika `K_` värden finns en bit ner på denna sida:
`https://www.pygame.org/docs/ref/key.html`.

Testa att köra och se vad som händer när du trycker på 'A'. Hur många gånger skriver den ut att du tryckt respektive släppt på tangenten?

(b) Ibland är man inte intresserad av om en tangent just trycktes ner, utan endast om den är nedtryckt eller inte. Det kan göras med `get_pressed()` som returnerar en lista med booleans (True/False), en för varje tangent. Denna lista kan indexeras med `K_` för att få om en viss tangent är nere eller inte.

```
if pygame.key.get_pressed()[pygame.K_a]:
    print("A is held this frame")
```

6. Rörelse

Vi vill nu låta användaren flytta på rektangeln med tangentbordet. Introducera globala variabler som bestämmer rektangelns x- och y- position, till exempel

`pos_x` och `pos_y`. Rektangeln kan då ritas i punkten `(pos_x, pos_y)` och när upp/ner/höger/vänster piltagent är nedtryckt kan dessa variabler uppdateras.

Piltangenterna heter: `K_UP`, `K_DOWN`, `K_LEFT` och `K_RIGHT`.

7. Delta tid och FPS

Om du testar att jämföra med en kompis som har en annan dator kanske ni märker att rektanglarna rör sig olika snabbt på skärmen. Detta är då datorn nu försöker köra gameloopen så ofta vi kan. En dator kanske kör den 80 gånger i sekunden medan en annan dator kör den 200 gånger i sekunden. Varje uppdatering flyttar sig rektangeln lika långt, vilket gör att hastigheten blir väldigt olika. De flesta skärmar kan dessutom bara uppdatera vad de visar 60 gånger i sekunden. Vi kan alltså begränsa så att spelet bara kör 60 gånger i sekunden och på så sätt spara datakraft utan att det påverkar vad man ser.

Vi introducerar även en delta-tid, tiden som har gått sedan förra gången spelet uppdaterades. Den kan vi använda för att göra så att all rörelse går lika fort. Målet är att datorn ska uppdatera spelet 60 gånger i sekunden, och då kommer deltatiden att bli 0.0166 s, men en lite svagare dator kanske bara klarar av att uppdatera spelet 40 gånger i sekunden, då blir deltatiden 0.025 s. Om vi multiplicerar all rörelse med deltatiden blir alla hastigheter i pixlar/sekund oavsett dator.

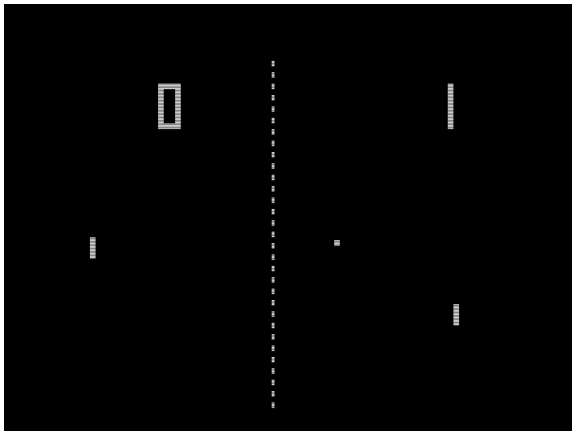
I `pygame` kan vi använda `Clock` där `clock.tick(FPS)` både begränsar till att endast köra `FPS`-gånger i sekunden, samt ger oss tiden sedan förra uppdateringen i milisekunder. Vi delar därmed på 1000.0 för att få till sekunder.

```
# --- Initiering / setup ---
clock = pygame.time.Clock()
dt = 0
```

```
    # --- Updates ---
    pos_x += 10 * dt # Notera att du kanske har en annan variabel för att
        bestämma rektangeln position från Upg 6.
    ...
    # Sist i gameloopen
    dt = clock.tick(60) / 1000.0
```

Efter detta borde rektangeln flytta sig lika fort oavsett vilken dator den är på.

Det gamla arcade-spelet Space invaders blev snabbare ju färre fiender som var kvar, idag är det som ett vanligt sätt att göra spel svårare, men från början var det en bug. Ju färre fiender det var på skärmen desto mindre hade processorn att göra och desto snabbare kunde den uppdatera spelet. Många äldre spel som är skrivna för långsamma processorer använder att processorn endast kan göra en begränsad mängd saker och att spelet får en bra hastighet genom detta. Men med moderna datorer blir det alldeles för snabbt.



Pong

Nu är vi redo att börja skapa Pong.

8. Lägg in två paddlar på varsin sida om skärmen som två spelare kan kontrolera. Till exempel med piltangenterna och 'W' och 'S'. Dessa kan göras som vita rektanglar. Låt två variabler avgöra dess höjder.

9. Lägg in en boll (antingen cirkel eller liten kvadrat som det var i orginalspelet) som har en hastighet. Bollen ska börja i mitten av skärmen och få en slumpmässig riktning och fart. Skapa fyra variabler som anger positionen och hastigheten. Dessa kan förslagsvis vara `pos_x`, `pos_y`, `vel_x` och `vel_y`.

   Varje uppdatering kan då bollen flyttas enligt

```
pos_x += vel_x * dt
pos_y += vel_y * dt
```

10. Kollision

    Vid kollisioner mellan två rektanglar eller en rektangel och en linje/kant på skärmen jämför vi deras koordinater för att se om de är inuti varandra. Kom ihåg att koordinaten är övre vänstra hörnet, så för att testa om nedre delen av rektangeln är under en linje, måste man jämföra `pos_y + rectangle_height` med linjens y-koordinat.

(a) Lägg till kollision med taket och golvet. Om bollen är i botten eller toppen av skärmen, spegla dess hastighet i y-riktningen.

```
if pos_y <= 0 and vel_y < 0 or pos_y + ball_size >= 600 and vel_y > 0:
    vel_y = -vel_y
```

   Varför behöver vi kolla `vel_x/y`?

(b) Lägg till kollision med paddlarna. Vi kan börja med att approximera paddlarna som en linje och när bollen träffar paddeln kan man spegla dess hastighet i x-riktningen.

```
if pos_x < paddle_left_x + paddle_width and pos_x > paddle_left_x and \
   vel_x < 0 and pos_y < paddle_left_y + paddle_height and \
   pos_y > paddle_left_y:
   vel_x = -vel_x
```

Implementera detta för både höger och vänster paddeln. Notera att vissa variabler som `paddle_left_x`, `paddle_width`, `...`, är gissningar på vad variablerna kan heta.

11. Räkna poäng

Om bollen försvinner ut från skärmen till vänster eller höger, räkna upp poäng för rätt spelare och återställ bollen i mitten av skärmen. Vi kommer se nästa lektion hur man kan rita text på skärmen för att visa poängen, nu kan man temporärt skriva ut poängen i terminalen med `print()`.

12. Förbättra spelet ytterligare

Några idéer på saker man kan implementera är:

- Låt bollen vara stilla i mitten av skärmen från början, och börja röra på sig när man trycker ned första tangenten.
- Låt bollen påverkas av paddelns hastighet, om paddeln åker snabbt uppåt, kan bollen få lite av den hastigheten uppåt.
- Gör en bot som kontrollerar den andra paddeln så man kan köra ensam. Hur gör man så att den är så rolig att spela mot som möjligt? Kanske borde botten inte vinna varje gång?

4. Spelprogrammering 2

21 juli

1. Ladda in och rita bilder

Fram tills nu har vi bara ritat former i olika färger, men vill man göra snyggare spel kan man ladda in bilder och använda istället. Bilden laddas först in och sparas i en surface som sedan kan placeras på skärmen.

```
# --- Initiering / setup ---
# Ladda in bilden, om det är en png med transparent bakgrund använd
    .convert_alpha()
name_pic = pygame.image.load("name.png").convert()
# Skala om bilden till rätt storlek.
name_pic = pygame.transform.scale(name_pic, (40, 40))
```

```
    # --- Render ---
    # Placera bilden på skärmen, med övre vänstra hörnet i punkten (10, 10).
    screen.blit(name_pic, (10, 10))
```

2. Musen

Vi har sett hur man tar input från tangentbordet, men många spel använder musen också. Om man vill reagera på ett vänster- eller högerklick kan det göras i event loopen:

```
    # --- Events ---
    for event in pygame.event.get():
        if event.type == pygame.MOUSEBUTTONDOWN: # eller MOUSEBUTTONUP
            if event.button == 1: # 1 = vänsterklick, 3 = högerklick
                print(f"Vänsterklick tryckt på: {event.pos}")
```

Vill man ha muspositionen för att till exempel sikta kan det göras i update med följande kod:

```
    # --- Update ---
    # Få mus-positionen
    # (0, 0) betyder att den är i övre vänstra hörnet.
    # (800, 600) betyder att den är i nedre högra hörnet.
    pos = pygame.mouse.get_pos()
    print(f"Musen är på x: {pos[0]}, y: {pos[1]}.")

    # Få hur långt muspekaren har rört sig sedan förra uppdatering
    rel = pygame.mouse.get_rel()
    print(f"Musen flyttade sig {rel[0]} pixlar i x och {rel[1]} pixlar i y.")
```

3. Text

Tidigare skrev vi ut poängen i terminalen, men den vill vi ju hellre rita på skärmen! För att göra det måste vi initiera `font`-modulen i python. Sedan kan vi ladda in 'helvetica'.

```
pygame.font.init()
my_font = pygame.font.SysFont('helvetica', 20)
```

För att sedan rita texten måste vi rendrera texten för sig till en surface, som vi kan rita till skärmen på samma sätt som en bild. Om vi har en text som aldrig ändras är det en bra idé att endast rendrera texten en gång när vi initierar allt.

```
    # --- Render ---
    # Skapa en bild med texten 'Poäng: 10', med anti-aliasing, som är svart
    text = my_font.render("Poäng: 10", True, (0, 0, 0))

    # Rita texten i punkten (10, 100)
    screen.blit(text, (10, 100))
```

Om en text oftast är samma, till exempel poäng-texten som bara uppdateras då och då, kan man spara rendreringen och rendrera om när poängen uppdateras.

Nu har vi gått igenom de flesta funktioner man behöver för att börja göra spel! Välj om du vill fortsätta jobba på Pong och polera färdigt den, visa poängen med text, eller börja göra ett annat spel, till exempel ett flug-smällar-spel (eller något helt eget du hittar på!).

4. Flug-smällar-spel

Skapa ett spel där man kontrollerar en flugsmälla med musen. Skapa massa flugor på skärmen som man kan smälla genom att klicka.

Skapa en lista med x- och y-positioner för flugorna. Loopa igenom de och rita bilden av flugan på den koordinaten. När användaren klickar, räkna ut avståndet till alla flugor med pytagoras sats, om det är mindre än radien på flugan, ta bort flugan. Om listan är kort kan man ta bort med `pop()`, men då måste datorn flytta alla element som är till höger om *i* ett steg till vänster. Om listan är längre är det smart att först byta plats på det sista elementet och element-*i*, och sedan ta bort det nya sista.

```
flugor_x.pop(i)
flugor_y.pop(i)
```

Förbättra Flug-smällar-spelet. Förslag:

(a) Lägg till maximalt antal flugor, och att det inte skapar flugor när flugsmällan står still.

(b) Räkna antalet smälda flugor, visa poängen i spelet.

(c) Det ska inte kunna skapas flugor på flugsmällan (åtminstone ska de inte ge poäng om de äts upp).

(d) Gör så att flugsmällan växer vid varje mumsbit.

(e) Få flugornas att röra sig. Kanske är de rädda för smällan och flyger ifrån honom?

(f) Lägg till något som flugsmällan ska undvika, till exempel en vas eller dator.

(g) Lägg till Game Over

(h) Lägg till möjligheten att starta om spelet.

(i) Begränsa flugsmällans rörelser med väggar och annat som den inte kan röra sig genom.

5. Tävlingsprogrammering: Simulering

23 juli

1. **Titelkostnad**

Du är chef för en avdelning på FlixNet som ansvarar för att skicka enbart filmtitlarna till kunderna.

Den totala kostnaden för att skicka en viss titel är helt enkelt antalet tecken (inklusive mellanslag) i den titeln, men din internetleverantör har kommit överens om att begränsa den kostnaden till ett angivet värde.

Givet titeln på en film och värdet på den kostnadsbegränsningen, beräkna kostnaden för att skicka den filmtiteln till dina kunder.

Körningsexempel:

```
1. Filmtitel? Radian Hood
Begränsning? 20
Kostnad: 11
2. Filmtitel? The Integer Games
Begränsning? 16
Kostnad: 16
3. Filmtitel? Pie Hard
Begränsning? 3.14159265
Kostnad: 3.14159265
```

2. **Talföljden**

Att fortsätta en påbörjad talföljd är en vanlig sorts uppgift i såväl matteböcker som IQ-tester. Men det smartaste måste väl ändå vara att skriva ett datorprogram som löser problemet en gång för alla.

Betrakta till exempel de så kallade triangeltalen:

1, 3, 6, 10, 15, 21 . . .

Ett mönster framträder om man tittar på skillnaden mellan varje tal och dess föregångare:

2, 3, 4, 5, 6 . . .

Tydligen ökar skillnaden med 1 i varje steg. Men för någon annan talföljd kan skillnaden till exempel öka med 13, minska med 2 eller vara konstant. Det enda som är säkert om de talföljder som förekommer i denna uppgift är att skillnaden mellan två intill varandra stående tal ändras lika mycket i varje steg.

Skriv ett program som frågar efter de tre första talen i talföljden (positiva heltal mindre än 100) och sedan skriver ut de 10 första talen åtskilda med blanksteg.

Körningsexempel:

```
Första talet ? 1
Andra talet ? 3
Tredje talet ? 6
Talföljden: [1, 3, 6, 10, 15, 21, 28, 36, 45, 55]
```

3. Frodiga Fält Från Fredrik (FFFF)

Fredrik satt på sitt kontor och tittade ut genom fönstret, han funderade på livet och vad som egentligen är viktigt. Ska han verkligen spendera resten av sitt liv framför en datorskärm och skriva programmeringskod? Det låter rätt meningslöst.

Det var dags för en förändring och Fredrik bytte han ut sitt tangentbord och mus mot en kratta och spade och startade ett trädgårdsföretag!

Efter flera års hårt arbete utomhus har Fredrik nu biceps som Van Damme och äger det främsta trädgårdsföretaget i hela Sverige. Han har precis haft tur nog att förhandla fram ett stort avtal för att så rektangulära gräsmattor åt godsägare.

Varje avtal specificerar storleken på de gräsmattor som behöver sås, samt kostnaden för frön per kvadratmeter. Hur mycket behöver han spendera på frön?

Körningsexempel:

```
Kostnad för frön per kvadratmeter? 20
Antal gräsmattor? 3
Bredd på grässmatta 1? 2
Längd på gräsmatta 1? 3
Bredd på grässmatta 2? 4
Längd på gräsmatta 2? 5
Bredd på grässmatta 3? 5
Längd på gräsmatta 3? 6
Svar: 1120 kronor

Kostnad för frön per kvadratmeter? 0.75
Antal gräsmattor? 2
Bredd på grässmatta 1? 2
Längd på gräsmatta 1? 3.33
Bredd på grässmatta 2? 3.41
Längd på gräsmatta 2? 4.567
Svar: 16.68 kronor
```

4. A Furious Cocktail

I Minecraft kan drycker (potion) skapas och drickas. När en dryck dricks, appliceras motsvarande dryckseffekt på spelaren under en viss tid. ”A Furious Cocktail” är namnet på bedriften att lyckas få alla dryckeseffekter samtidigt. Detta är en utmanande prestation, eftersom det tar tid att dricka varje dryck, och endast en dryck kan drickas åt gången.

Antag att det finns N unika drycker i din inventory, som var och en har en aktiv tidsperiod på $t_1, ..., t_N$ sekunder. Om det tar T sekunder att dricka en dryck, är det möjligt att ha alla N dryckseffekter applicerade samtidigt? Observera att om en dryckseffekt tar slut samtidigt som du avslutar att dricka en annan dryck, räknas inte dryckerna som applicerade samtidigt.

Körningsexempel:

```
Antal drycker? 5
Antal sekunder det tar att dricka en dryck? 1
Tid dryck 1 är aktiv? 1
Tid dryck 2 är aktiv? 2
Tid dryck 3 är aktiv? 3
Tid dryck 4 är aktiv? 4
Tid dryck 5 är aktiv? 5
Svar: Ja, det går!

Antal drycker? 3
Antal sekunder det tar att dricka en dryck? 2
Tid dryck 1 är aktiv? 4
Tid dryck 2 är aktiv? 3
Tid dryck 3 är aktiv? 2
Svar: Nej, det går inte.

Antal drycker? 3
Antal sekunder det tar att dricka en dryck? 2
Tid dryck 1 är aktiv? 2
Tid dryck 2 är aktiv? 3
Tid dryck 3 är aktiv? 5
Svar: Ja, det går!
```

5. Snöskottning

Scott har som många andra haft problem under vintertiden med allt snö. Scotts hus måste skottas ofta, men han tycker själv att skotta varje dag är för ofta. Du ska hjälpa Scott att sätta upp ett skott-schema för de kommande N dagarna. Till din hjälp har du en detaljerad väderleksprognos, så du vet exakt hur många cm det snöar eller smälter varje dag (innan kvällen). När det ligger minst H cm snö en kväll så är det dags att skotta, och när man skottar så försvinner all snö. Från början är det ingen snö alls.

Indata

Första raden innehåller antalet dagar $1 \leq N \leq 100$ och andra raden snötröskeln $1 \leq H \leq 1000$. Därefter följer N heltal mellan -100 och 100 (inklusive). Dessa tal beskriver hur många cm snö det kommer under var och en av de kommande N dagarna. Ett negativt värde betyder att denna mängd smälter istället (men snötäcket kan naturligtvis aldrig bli mindre än 0).

Utdata

Utdatat ska bestå av ett heltal, antalet kvällar Scott måste skotta.

Körningsexempel:

```
Antal dagar? 10
Snögräns? 27
Snöförändringar? 5 -7 8 19 -20 22 8 26 -15 14
Scott behöver skotta minst 2 gånger
```

6. Tävlingsprogrammering: Permutationer

24 juli

1. **Permutationer**

   a) Skriv ett program som skriver ut alla möjliga strängar, av längd 3, som man kan bilda med bokstäverna ’a’, ’b’ och ’c’. Det ska bli 27st strängar, varför?

   b) Modifiera programmet i a) så att varje bokstav bara får förekomma en gång. Det ska bli 6st strängar, varför?

   c) Skriv ut alla permutationer av strängen ”EIR”.

2. **Send more money**

   Ersätt varje bokstav med en unik siffra mellan 0 och 9 så att uträkningen i figur 6.1 stämmer. Notera att talen inte kan börja med 0, för i så fall skulle man kunna strunta i att skriva ut första bokstaven.

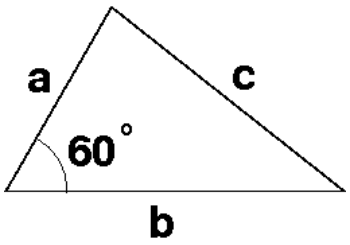
Figur 6.1

3. Lagomvinklade trianglar

En lagomvinklad triangel är vad vi i denna uppgift kallar en triangel där minst en av vinklarna är exakt 60 grader. De lagomvinklade trianglarna känner sig ofta förbisedda jämfört med de mycket mer kända rätvinkliga trianglarna (så kallat mindervinkelkomplex), trots att de lagomvinklade också har en snygg formel för sina sidlängder:

$c^2 = a^2 + b^2 - ab$

Skriv ett program som skipar lite rättvisa i detta triangeldrama genom att fråga efter ett tal N (mellan 1 och 100) och sedan skriva ut hur många lagomvinklade trianglar det finns vars sidor är heltal i intervallet 1 till N.



Figur 6.2: Ett exempel på en lagomvinklad triangel med sidlängderna $a = 5, b = 8$ och $c = 7$.

Körningsexempel

```
Talet N ? 25
Antal trianglar: 35

Talet N ? 70
Antal trianglar: 112
```

4. Klockan

Figur 6.3

Om någon frågar hur mycket klockan är, svarar de flesta ”kvart över fem”, 15:29 eller något liknande. Vill man göra det lite svårare så kan man annars svara med vinkeln mellan tim- och minutvisaren, eftersom man ur denna information entydigt kan bestämma klockslaget. Dock är det många människor som är ovana vid detta sätt att ange tider, så det vore bra att ha ett datorprogram som översätter till ett mer normalt format. Du ska skriva ett sådant program.

Vi förutsätter att vår klocka saknar sekundvisare och endast visar ett helt antal minuter (det vill säga: båda visarna hoppar framåt bara på hel minut). Vinkeln avläses genom att utgå från timvisaren och sedan mäta hur många grader medurs minutvisaren ligger (se figur 6.3). För att undvika decimaler anges vinkeln i tiondels grader (så att 85.5 grader skrivs som 855). Detta tal är alltid ett heltal mellan 0 och 3595 (inklusive) och är, som en följd av att endast hela minuter visas, alltid delbart med 5.

Programmet ska fråga efter en vinkel och sedan skriva ut tiden i vanligt digitalformat, alltså h:mm eller hh:mm, beroende på antalet timmar. Vi förutsätter att det är morgon, så alla tider ska ligga mellan 0:00 och 11:59 (inklusive).

Två körningsexempel

```
Vinkel ? 855
Klockan är 1:21

Vinkel ? 3140
Klockan är 3:08
```

V Programmering - Babbage

1. Introduktion

18 juli

Uppvärmningsuppgifter

1. Skriv ett program som skriver ut alla jämna tal i intervallet $[10, 80]$.

2. Skriv ett program som beräknar vilken tuple (2 tal i varje) i en lista som har störst produkt. T.ex. tuplen $(4, 8)$ har en produkt som är 32.

```
pairs = [(1, 3), (5, 7), (-5, -1), (-9, -10), (-5, 4), (5, 100)]
...
```

3. Skriv ett program som först läser in 5 heltal som användaren skriver in, sen beräknar vilket tal som är näst störst och skriver ut det.

   Tips på användbara funktioner: input(), split()

Sets och dictionaries

4. Skriv ett program som med hjälp av ett set beräknar hur många unika element det finns i en lista.

```
x = [1, 2, 3, 1, 2, 1, 1, "Hello", "Hello again", "Hello"]
...
print(number_of_unique_elements)
```

5. Skriv ett program som med hjälp av en dictionary beräknar hur många gånger varje element i en lista förekommer. Skriv ut varje element i listan samt hur många gånger det förekommer.

```
x = [1, 2, 3, 1, 2, 1, 1, "Hello", "Hello again", "Hello"]
...
```

6. Skriv ett program lagrar namnen samt åldrar på personer i en dictionary, och sen skriver ut åldern på varje person som användaren skriver in.

```
names = ["Alice", "Bob", "Charlie", "David", "Erik", "Fredrik"]
ages = [14, 20, 25, 30, 84, 12]
... # Skapa en dictionary här

for i in range(5): # Exempel på att köra programmet för flera namn.
    name = input()
    ... # Skriv ut åldern på personen som man fick från input
```

Rekursiva funktioner

7. Skriv en rekursiv funktion som beräknar fakulteter. Vi utgår ifrån definitionen:

```
0! = 1
x! = x * (x-1)! # För heltal där x > 0
```

8. Skriv en rekurisv funktion som beräknar fibonacci-talen. Lägg även in en variabel som håller koll på hur många rekursiva anrop som gjordes. Skriv ut både fibonaccitalet och antalet rekursiva anrop som gjordes. Finns det nått intressant mönster i antalet anrop som behövs?

Lite inbyggda funktioner

Det finns en del bra funktioner att känna till i python, som gör våra liv mycket lättare ibland. Här kommer ett par exempel. Ifall nånting är nytt så får ni gärna testa koden för att se vad som händer. Det rekommenderas att printa värdena på variabler för att se vad som händer i varje steg ifall man är förvirrad.

Det finns så klart många fler inbyggda funktioner än de vi tar upp här, men det sparar vi till ett annat tillfälle ;)

```
numbers = [5, 9, 10, 2]
contains_2 = 2 in numbers # detta blir False
contains_5 = 5 in numbers # detta blir True

# Byt från t.ex. list till set på enkelt sätt
x = [3, 4, 5]
y = set(x)
# Detta funkar även för andra klasser!
# Tuple till lista:
x = (3, 10, 20)
y = list(x)
# range till lista:
x = range(10)
y = list(x)
# str till lista
x = "This is a string"
y = list(x)

# Sätt ihop alla strängar i en lista:
words = ["This", "is", "a", "sentence"]
join_string = " " # Testa med lite olika värden här och se vad som händer.
sentence = join_string.join(words)
print(sentence)


numbers = [10, 30, 5, 2, 3, 100]
max_number = max(numbers) # Hittar största talet i numbers
max_index = numbers.index(max_number) # Vilket index talet ligger på
mean_number = sum(numbers) / len(numbers)
# Fråga: kan man använda dessa funktioner på saker som inte är heltal? Ifall
    ni fyller numbers med till exempel stränger, vad händer då?
```

9. Som ni såg ovan så finns det många klasser som man kan konvertera direkt mellan. Använd dessa för att skriva ut alla bokstäver som förekommer minst en gång i en sträng.

```
word = input()
...
unique_letters = ...
print(unique_letters)
```

10. Testa följande kod. Som ni lär inse så har Tobias skrivit en bugg i den här koden som gör att programmet kan krascha. Hitta buggen och fixa den så att programmet fungerar, helst utan att ändra på massor i koden. Programmet ska först läsa in rad text från input som vi antar består enbart av siffror och mellanslag, slå ihop alla tal och sen räkna ut vilken den största tvåpotensen är som delar talet. T.ex. för 48 så är 16 den största tvåpotensen som delar 48, så då skriver vi ut 4. Ifall ni kör fast så är återigen ett tips att printa variabler för att se att ni har koll på exakt vad som händer.

```
x = input()
s = set()
for c in x:
    s.add(c)
if " " in s:
    s.remove(" ")
x = int("".join(s))
ans = 0
while x % 2 == 0:
    ans += 1
    x = x // 2
print(ans)
```

Flyttal är onda (eller?)

11. Testa att köra följande kod:

```
x = 0.1
print(x)
y = 0.3
print(x * 3 - y)
```

Vad fick ni för resultat? Varför hände det?

12. Ett exempel på när flyttal kan ställa till med problem: följande program är tänkt att skriva ut alla multiplar av 0.3 som är mindre än eller lika med 12. Men om ni testkör det så märker ni att nånting blir fel. Hur kan ni fixa programmet så att rätt saker skrivs ut.

```
x = 0
while x <= 12:
```

```
    print(x)
    x += 0.3
```

13. Som vi har sett så kan det bli lite fel ibland om man har decimaltal. Tänk er nu följande problem: ni vet redan att ett tal är delbart med 7 och vill skriva ut vad talet blir ifall ni dividerar det med 7. Ni tänker kanske då att ni borde undvika problem eftersom allting är heltal. Men vad är skillnaden mellan de två följande printsatserna.

```
x = int(input())
print(x // 7)
print(round(x / 7)) # Använder round för att slippa .0 i slutet.
```

Som ni kanske förstår efter det här så kan flyttal ge lite oväntade resultat ibland. Därför är det viktigt att man är försiktig i vissa sammanhang, och det kan ibland löna sig att hålla sig till heltal.

2. Kompakt pythonkod

19 juli

Att läsa in input

```
# Läsa in en rad input:
line = input() # Allt blir till en sträng, inklusive mellandslag osv.

# Läsa in ett heltal som står självt på en rad:
number = int(input())

# Läs in en lista med ord:
words = input().split()

# Läs in en lista med heltal:
numbers = list(map(int, input().split()))
```

Att printa saker

```
# För att printa ut en sak (kan vara antingen en sträng, int, flyttal osv):
variable = ...
print(variable)


# För att printa ut en lista med saker (på en rad):
# Antingen:
for elem in list:
    print(elem, end = " ") # Det blir ett mellanrum mellan alla saker i listan.
print() # Gör så att framtida saker skrivs ut på en ny rad.

# Eller:
print(*list)
```

Inbyggda saker i python

Här kommer vi att gå igenom en del funktioner som finns inbyggda i python. Dessa funktioner är väldigt vanliga i programmaring, då det förkortar koden med mycket, men så länge man kan syntaxen, så är det fortfarande läsbart.

Lambdafunktioner

Lambdafunktioner är en typ av pythonfunktion, men som inte skrivs med ordet *def* för att definiera den. Det gör att man kan använda den lokalt, och göra koden lite mer kompakt och läsbar.

```
# Exempel på hur man kan spara en lambdafunktion och sedan använda den
foo = lambda x: x ** 2
five_squared = foo(5)

# Samma med en vanlig funktion:
def foo(x):
    return x ** 2
five_squared = foo(5)

# Lambda funktioner kan även ta in flera argument:
division = lambda numerator, denominator: numerator / denominator
```

Det är ofta användbart att använda lambdafunktioner tillsammans med de funktioner som vi kommer gå igenom nu. Försök att använda lambdafunktioner om ni kan. Men ifall det känns svårt så kan ni hålla er till vanliga funktioner.

map

Nu ska vi använda funktionen map. map tar in 2 parametrar: först en funktion och sen en iterable. Dvs någonting som går att iterera genom. Det kan vara t.ex. en lista, tuple, dictionary och liknande. Ett exempel såg ni tidigare:

```
numbers = list(map(int, input().split()))
```

Det som skickas in här är funktionen *int*, och det kommer att appliceras på varje sträng som kommer som del av *input().split()*. Som returvärde får vi ett mapobjekt. Därför måste vi göra om det till den datastruktur som vi vill använda istället. I det här fallet så gör vi därför om det till en lista.

1. Skriv kod som använder map för att skapa en lista som innehåller kvadraterna av alla talen i en annan lista.

2. Skriv kod som använder map som applicerar följande på varje tal i listan: ifall talet är jämnt, kvadrera det. Annars, ersätt det med None.

List comprehensions

List comprehensions är väldigt användbara i python. Det är på vissa sätt väldigt likt att använda lambda, men syntaxen är lite annorlunda. Ibland kan det vara lättare att använda lambda, och ibland tvärt om. Därför är det bra att känna till båda metoderna.

```
# Exemepel på en list comprehension som kvadrerar alla tal i en lista:
x = [1, 2, 3, 4]
y = [elem ** 2 for elem in x] # Blir: [1, 4, 9, 16]

# Exempel på en list comprehension som bara behåller jämna tal:
x = [1, 2, 3, 4]
y = [elem for elem in x if elem % 2 == 0] # Blir [2, 4]

# Exempel där man gör båda saker samtidigt.
```

```
x = [1, 2, 3, 4]
y = [elem ** 2 for elem in x if x % 2 == 0] # Blir [2, 16]
```

3. Skriv en list comprehension som lägger till bokstaven *s* i slutet av varje sträng i en lista.

4. Skriv en list comprehension som skapar en lista av tuples av längd 26, där tuple på index *i* i den resulterande listan är lika med $(i, bokstavi)$, där bokstav i är den bokstav som kommer på plats *i* i alfabetet.

5. Översätt föregående lösning till en dictionary comprehension, som mappar varje bokstav i alfabetet till vilken plats den har i alfabetet.

```
# Här är ett annat exempel på en dictionary comprehension:
x = {i : i ** 2 for i in range(10)} # Skapar en dictionary som innehåller de
    10 första kvadrattalen.
```

Kodgolf

Kodgolf går ut på att skriva så kort kod som möjligt, (för det mesta räknat som antalet tecken koden består utav). Nu ska vi försöka att göra det för lite exempel. Försök gärna att använda funktioner så som map, list comprehensions osv då det ofta hjälper för att göra koden kortare.

6. Skriv ett program som läser in en lista av heltal från input, och sedan kollar hur många av talen som är lika med deras index i listan. Printa svaret så att det syns när man kör programmet.

7. Skriv ett program som för varje rad av input skriver ut den lägsta bokstaven som finns i raden och även den högsta bokstaven som finns.

```
# Ifall man till exempel skriver detta som en rad:
>>> Hejsan det här är lite text.
# Så ska erat program skriva ut:
>>> a ä
```

8. Skriv ett program som för varje rad av input skriver ut andelen ord som innehåller bokstaven x.

```
# Exempel:
>>> Det här är ett par ord. Vissa innehåller x så som xylofon men vissa gör
    inte det.
# Bör skriva ut (2 / 17):
>>> 0.11764705882352941
```

9. Ni ska nu försöka förkorta eran kod i tidigare uppgifter så mycket som möjligt. Ifall ni kör fast eller har frågor så är det alltid ok att fråga ledarna om tips. Det finns nämligen ofta många knep som man kan använda för att förkorta sin kod.

10. Grattis! Du har tagit dig igenom alla upggifter på dagens lektion. Som extra

utmaning nu så finns det ett par problem delade i kodgolf kanalen på discord. Där gäller det att lösa något svårare problem med så få tecken som möjligt. Lycka till!

3. Spelprogrammering 1

20 juli

Idag ska vi börja med spelprogrammering. Vi kommer att använda python-biblioteket `pygame` för att rita saker på skärmen och ta input från tangentbordet och musen. `pygame` kan man använda för allt från att göra allt från spel till simuleringar.

Idag kommer vi fokusera på grunderna, hur spel är uppbyggda, hur man använder `pygame` och sist återskapa det klassiska spelet Snake.

Spel generellt

1. Gameloop

Många spel är uppbyggda på väldigt liknande sätt, det är en initieringsdel i början av koden, här sker allt som bara ska ske en gång när programmet startar. Alla varibler och objekt kan skapas, bilder och ljudfiler kan laddas in och vi skapar ett fönster.

Sedan kommer gameloopen, en `while`-loop som kör om och om igen tills vi stänger programmet. Den kör en gång för varje uppdatering spelet gör. Vi läser in input från användaren, uppdaterar påverkade objekt och ritar allt.

Sist rensar vi upp efter oss, tar bort objekt och stänger fönstret.

```
# --- Initiering / setup ---
#  skapa alla variabler och objekt man behöver senare under spelets gång,
    ladda in bilder, osv

runningrön = True
while running:
    # --- Events ---
    #  ta input från användaren, har man klickat med musen, tryckt ner eller
        släppt på en tangent
    # --- Updates ---
    #  uppdatera alla variabler och objekt. Flytta till exempel spelaren
        frammåt om den håller ner 'W'.
    # --- Render ---
    #  rita allt på skärmen
    ...

# --- Cleanup ---
#  gör allt man vill göra precis innan spelet stängs, radera objekt i minnet,
    spara
```

Hur man använder pygame

Vi vill nu använda `pygame` för att skapa ett fönster, ta input från användaren och börja rita saker.

2. Öppna ett fönster

```
# --- Initiering / setup ---
import pygame # Importera pygame biblioteket

pygame.init() # Starta allt internt i pygame

# Välj storlek på fönstret med bredd (800 pixlar) och höjd (600 pixlar).
screen = pygame.display.set_mode([800, 600])
pygame.display.set_caption("Namn på fönstret")

runningrön = True
while running:
    # --- Events ---
    # --- Updates ---
    # --- Render ---
    screen.fill((255, 255, 255)) # Fyll skärmen med vit

    # Uppdatera det som visas
    pygame.display.flip()

# --- Cleanup ---
# Rensa allt som skapats vid init()
pygame.quit()
```

3. Fönstret går i nuläget bara att stänga genom att skicka *CTRL+C* i terminalen. För att programmet ska stoppa när man trycker på stängknappen måste det kodas. Allt användaren gör, tex trycker på tangenter eller klickar på stängknappen läggs till som ett event som koden kan reagera på.

```
    # --- Events ---
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            runningrön = False
```

4. Rita former

För att göra klassiska spel räcker det med att kunna rita rektanglar och cirklar. `pygame` har inbyggt att rita massa olika former och linjer.

```
    # --- Render ---
    screen.fill((255, 255, 255))

    pygame.draw.rect(screen, (255, 0, 255), (100, 200, 500, 300))
    # Rita en lila rektangel (röd = 255, grön = 0, blå = 255), men övre hörnet
        i punkten (100, 200) (100 från vänster, 200 uppifrån) som är 500x300
        pixlar stor
```

```
pygame.draw.circle(screen, (255, 255, 0), (550, 300), 80)
# Rita en gul cirkel (röd = 255, grön = 255, blå = 0), med centrum i
    punkten (550, 300) och raide 80.

pygame.draw.line(screen, (255, 0, 0), (0, 0), (800, 400), width = 20)
# Rite en röd linje (röd = 255, grön = 0, blå = 0) från punkten (0, 0) till
    punkten (800, 600).

pygame.display.flip()
```

`pygame` kan rita många fler former och polygoner, alla olika `draw`-funktioner finns här: `https://www.pygame.org/docs/ref/draw.html`.

Notera också vilka av rektangeln, cirkeln och linjen som hamnar ovanpå varandra. Testa byta plats på två rader och se vad som händer.

5. Interaktivitet

(a) Testa om en tangent blivit nedtryckt eller uppsläppt:

```
# --- Event ---
for event in pygame.event.get():
    if event.type == pygame.QUIT:
        runningrön = False
    elif event.type == pygame.KEYDOWN:
        if event.key == pygame.K_a:
            print("A was pressed this frame")
    elif event.type == pygame.KEYUP:
        if event.key == pygame.K_a:
            print("A was released this frame")
```

Alla olika `K_` värden finns en bit ner på denna sida:
`https://www.pygame.org/docs/ref/key.html`.

Testa att köra och se vad som händer när du trycker på 'A'. Hur många gånger skriver den ut att du tryckt respektive släppt på tangenten?

(b) Ibland är man inte intresserad av om en tangent just trycktes ner, utan endast om den är nedtryckt eller inte. Det kan göras med `get_pressed()` som returnerar en lista med booleans (True/False), en för varje tangent. Denna lista kan indexeras med `K_` för att få om en viss tangent är nere eller inte.

```
if pygame.key.get_pressed()[pygame.K_a]:
    print("A is held this frame")
```

6. Rörelse

Vi vill nu låta användaren flytta på rektangeln med tangentbordet. Introducera globala variabler som bestämmer rektangelns x- och y- position, till exempel `pos_x` och `pos_y`. Rektangeln kan då ritas i punkten `(pos_x, pos_y)` och när upp/

ner/höger/vänster piltagent är nedtryckt kan dessa variabler uppdateras.

Tips: piltangenterna heter: `K_UP`, `K_DOWN`, `K_LEFT` och `K_RIGHT`.

7. Delta tid och FPS

Om du testar att jämföra med en kompis som har en annan dator kanske ni märker att rektanglarna rör sig olika snabbt på skärmen. Detta är då datorn nu försöker köra gameloopen så ofta vi kan. En dator kanske kör den 80 gånger i sekunden medan en annan dator kör den 200 gånger i sekunden. Varje uppdatering flyttar sig rektangeln lika långt, vilket gör att hastigheten blir väldigt olika. De flesta skärmar kan dessutom bara uppdatera vad de visar 60 gånger i sekunden. Vi kan alltså begränsa så att spelet bara kör 60 gånger i sekunden och på så sätt spara datakraft utan att det påverkar vad man ser.

Vi introducerar även en delta-tid, tiden som har gått sedan förra gången spelet uppdaterades. Den kan vi använda för att göra så att all rörelse går lika fort. Målet är att datorn ska uppdatera spelet 60 gånger i sekunden, och då kommer deltatiden att bli 0.0166 s, men en lite svagare dator kanske bara klarar av att uppdatera spelet 40 gånger i sekunden, då blir deltatiden 0.025 s. Om vi multiplicerar all rörelse med deltatiden blir alla hastigheter i pixlar/sekund oavsett dator.

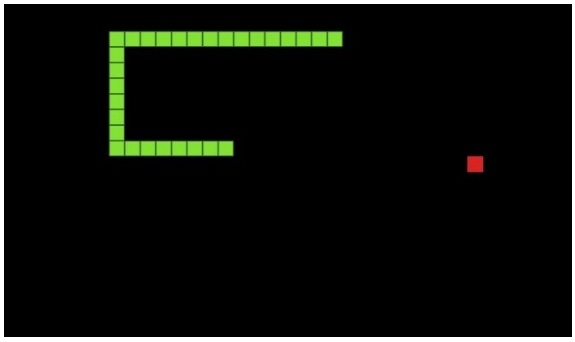
I `pygame` kan vi använda `Clock` där `clock.tick(FPS)` både begränsar till att endast köra `FPS`-gånger i sekunden, samt ger oss tiden sedan förra uppdateringen i milisekunder. Vi delar därmed på 1000.0 för att få till sekunder.

```
# --- Initiering / setup ---
clock = pygame.time.Clock()
dt = 0
```

```
    # --- Updates ---
    pos_x += 10 * dt # Notera att du kanske har en annan variabel för att
        bestämma rektangeln position från Upg 6.
    ...
    # Sist i gameloopen
    dt = clock.tick(60) / 1000.0
```

Efter detta borde rektangeln flytta sig lika fort oavsett vilken dator den är på.

Det gamla arcade-spelet Space invaders blev snabbare ju färre fiender som var kvar, idag är det som ett vanligt sätt att göra spel svårare, men från början var det en bug. Ju färre fiender det var på skärmen desto mindre hade processorn att göra och desto snabbare kunde den uppdatera spelet. Många äldre spel som är skrivna för långsamma processorer använder att processorn endast kan göra en begränsad mängd saker och att spelet får en bra hastighet genom detta. Men med moderna datorer blir det alldeles för snabbt.



Snake

Nu är vi redo att börja skapa Snake. Snake existerar på ett rutnät, det är ofta smidigt att spara rutstorleken i en variabel så man enkelt kan ändra det senare. Till exempel `cell_size = 40`. Detta gör att vi får två koordinatsystem, ett som går $0 \rightarrow 800$ och ett som går $0 \rightarrow 20$. Det ena är pixlarna där vi ritar alla kvadrater, och det andra där all logik sker. Om ett äpple är på punkten `(2, 3)` ska det då ritas på `(2 * cell_size, 3 * cell_size)`.

8. Lägg in ormens huvud som kontinuerligt rör sig framåt, försök få till så att den hoppar framåt en ruta i taget. Gör så att man kan byta riktning med till exempel piltangenterna. Du kan rita huvudet som en kvadrat i valfri färg.

Tips: man kan använda `time`-biblioteket för att vänta en viss tid mellan flyttarna.

```
import time
last_move = time.time()
```

```
    # --- Update ---
    if time.time() - last_move > 0.7:
        # Move
        last_move = time.time()
```

`time.time()` ger hur många sekunder som gått sedan 1 Jan 1970. Vad representerar `time.time()- last_move` i koden ovan?

9. Lägg till ormens svans, spara huvudets senaste 'n' positioner och rita även dessa. Introducera en variabel som bestämmer svanslängden.

10. Kollisioner

(a) Om huvudet åker utanför skärmen får du välja om spelaren ska dö, eller om den ska åka runt till andra sidan. För att loopa runt koordinaten kan man använda `%` operatorn med bredden på skärmen.

```
pos_x = pos_x % (screen_width / cell_size)
```

Testa `print(-1 % 10)` och se vad du får.

(b) Om huvudet krockar in i svansen är spelet över. Jämför huvudets position med alla svanspositioner.

11. Lägg in ett äpple som spelaren kan äta. Öka svansens längd varje gång spelaren äter ett äpple och öka hastigheten lite. Vi kommer se nästa lektion hur man kan ladda in bilder och visa för att få snygga äpplen, men nu kan de temporärt ritas som en röd kvadrat.

12. Förbättra spelet ytterligare
Några idéer på saker man kan implementera är:

- Gör så att spelet går lite snabbare varje gång man äter ett äpple för att öka svårighetsgraden.
- Lägg in en liten buffer tid om spelaren är på väg att dö och krocka in i sig själv. Gör så att spelet väntar
- När spelaren dör, skriv ut poängen (hur många äpplen den lyckades äta) i terminalen.

4. Spelprogrammering 2

21 juli

1. Ladda in och rita bilder

Fram tills nu har vi bara ritat former i olika färger, men vill man göra snyggare spel kan man ladda in bilder och använda istället. Bilden laddas först in och sparas i en surface som sedan kan placeras på skärmen.

```
# --- Initiering / setup ---
# Ladda in bilden, om det är en png med transparent bakgrund använd
    .convert_alpha()
name_pic = pygame.image.load("name.png").convert()
# Skala om bilden till rätt storlek.
name_pic = pygame.transform.scale(name_pic, (40, 40))
```

```
    # --- Render ---
    # Placera bilden på skärmen, med övre vänstra hörnet i punkten (10, 10).
    screen.blit(name_pic, (10, 10))
```

2. Musen

Vi har sett hur man tar input från tangentbordet, men många spel använder musen också. Om man vill reagera på ett vänster- eller högerklick kan det göras i event loopen:

```
    # --- Events ---
    for event in pygame.event.get():
        if event.type == pygame.MOUSEBUTTONDOWN: # eller MOUSEBUTTONUP
            if event.button == 1: # 1 = vänsterklick, 3 = högerklick
                print(f"Vänsterklick tryckt på: {event.pos}")
```

Vill man ha muspositionen för att till exempel sikta kan det göras i update med följande kod:

```
    # --- Update ---
    # Få mus-positionen
    # (0, 0) betyder att den är i övre vänstra hörnet.
    # (800, 600) betyder att den är i nedre högra hörnet.
    pos = pygame.mouse.get_pos()
    print(f"Musen är på x: {pos[0]}, y: {pos[1]}.")

    # Få hur långt muspekaren har rört sig sedan förra uppdatering
    rel = pygame.mouse.get_rel()
    print(f"Musen flyttade sig {rel[0]} pixlar i x och {rel[1]} pixlar i y.")
```

3. Text

Tidigare skrev vi ut poängen i terminalen, men den vill vi ju hellre rita på skärmen! För att göra det måste vi initiera `font`-modulen i python. Sedan kan vi ladda in 'helvetica'.

```
pygame.font.init()
my_font = pygame.font.SysFont('helvetica', 20)
```

För att sedan rita texten måste vi rendrera texten för sig till en surface, som vi kan rita till skärmen på samma sätt som en bild. Om vi har en text som aldrig ändras är det en bra idé att endast rendrera texten en gång när vi initierar allt.

```
    # --- Render ---
    # Skapa en bild med texten 'Poäng: 10', med anti-aliasing, som är svart
    text = my_font.render("Poäng: 10", True, (0, 0, 0))

    # Rita texten i punkten (10, 100)
    screen.blit(text, (10, 100))
```

4. Sprites

När spelet växer och koden blir längre får vi en massa variabler av typen `head_x`, `head_y`, som kan vara svåra att hålla koll på. Därför kan vi vilja dela upp det i flera filer och klasser. Pygames inbyggda Sprite klass är till för just detta. Vi kan skapa våra egna klasser som utgår från Sprite för att få tillgång till en massa inbyggda funktioner.

```
class Snake(pygame.sprite.Sprite):
    def __init__(self):
        super(Snake, self).__init__()
        self.surf = pygame.Surface((40, 40))
        self.surf.fill((0, 255, 0))
        self.rect = self.surf.get_rect()
```

Spriten innehåller en surface och en rect som innehåller vad och var den ska rendreras. Fördelen med rect är att den håller reda på en x,y-koordinat, bredden och höjden i bara en variabel. När vi har ritat en rektangel och get den en tuple med (x, y, width, height) motsvarar det en rect, men vi kan spara det som en rect för att både få mindre att skriva, samt få tillgång till en massa kollisionsfunktioner som kan testa om två sprites överlappar. I snake är kollisionerna väldigt simpla, då de antingen är på samma ruta eller inte, men i spel med friare rörelse kan det bli krångligare, och då kan rect undelätta.

Om vi har många sprites kan vi sätta in de i en sprite group, det fungerar nästan som en lista, men ger oss några extra funktioner som kan vara smidiga.

```
head = Snake()
tail = pygame.sprite.Group()
for i in range(10):
```

```
    tail_part = Snake()
    tail.add(tail_part)
```

Dessa kan sedan rendreras på samma sätt som bilder och text när de också blivit ett pygame surface:

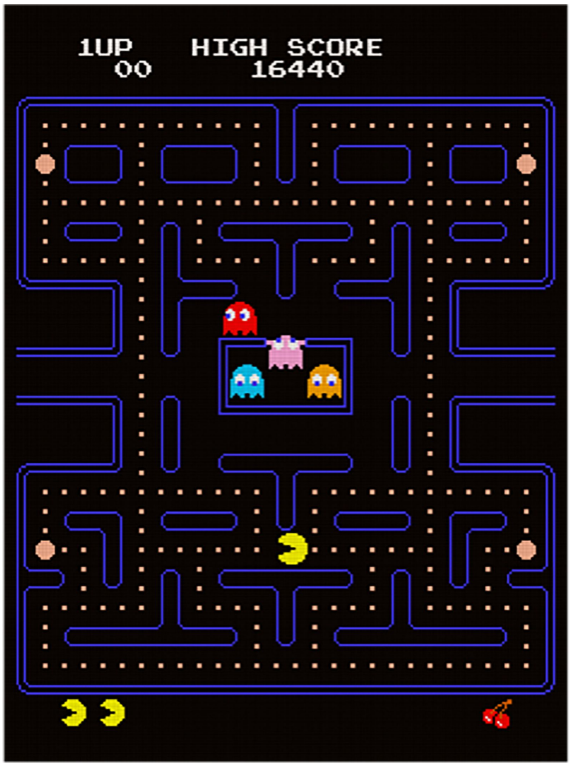
```
    # --- Render ---
    screen.blit(head.surf, head.rect)

    for t in tail:
        screen.blit(t.surf, t.rect)
```

En stor fördel med att använda sprite groups, är att pygame kan testa kollisionen åt oss:

```
if pygame.sprite.spritecollideany(head, tail):
    print("Game over!")
    running = False
```

https://www.pygame.org/docs/ref/sprite.html#pygame.sprite.collide_rect

Nu har vi gått igenom de flesta funktioner man behöver för att börja göra spel! Välj om du vill fortsätta jobba på snake och polera färdigt den, eller börja göra ett annat spel, till exempel pacman eller snake (eller något helt eget du hittar på!).



5. — **Extra.** Pacman

(a) Skapa kartan/världen pacman rör sig i. Man kan börja med att rita kvadrater där det är väggar, senare kan man byta ut det till bilder av väggarna för att få det snyggt som orginalet.

(b) Skapa pacman som kan gå runt och kollidera med väggarna.

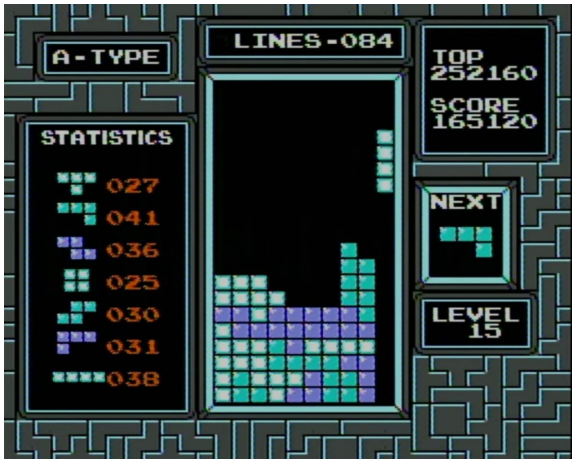
(c) Skapa mat som pacman kan äta upp, samla det i en poäng.

(d) Visa poängen högst upp som en text.

(e) Skapa fyra spöken som följer efter pacman, här man kan ha kul och exprimentera med deras AI. Sök upp hur de beter sig i orginalspelet, kan du återskapa det? Kan du göra de svårare att spela mot?

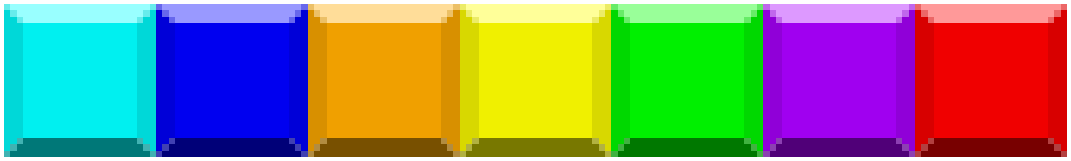
(f) Animera pacman och spökerna genom att rotera genom en sekvens av bilder.

Ett sprite-sheet som innehåller alla bilder från pacman finns här: `https://www.spriters-resource.com/arcade/pacman/sheet/52631/`.

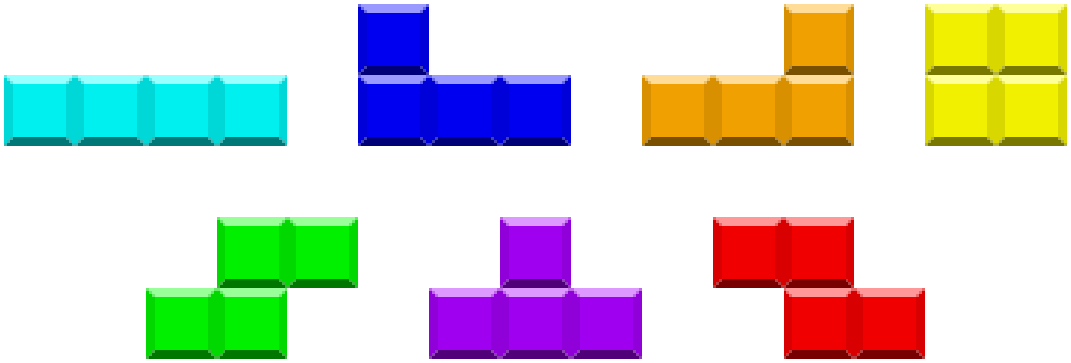


6. — **Extra.** Tetris

(a) Skapa ett rutnät och rita rektanglar. Vill man göra det snyggare kan man rita bilder för rutorna istället.



(b) Skapa ett fallanda block som är ett av de möjliga i tetris.



Tips: Tänk på att när en rad rensas kan ett halvt objekt försvinna och halva vara kvar. Det är därför bra att spara och rita alla block som flera objekt, ett för varje kvadrat.

(c) Låt användaren rotera och flytta (höger/vänster) det fallande blocket.

(d) När blocket landar på botten, se om det fyller ut några rader, ge poäng om den gör det och ta bort raderna. Sök upp hur orginal-tetris delar ut poäng.

(e) Implementera funktionen att användaren kan lägga undan en bit till senare, och byta biten den har med den undanlagda.

(f) Lägg till att den visar vilken bit som kommer härnäst.

5. Grafteori och algoritmer

23 juli

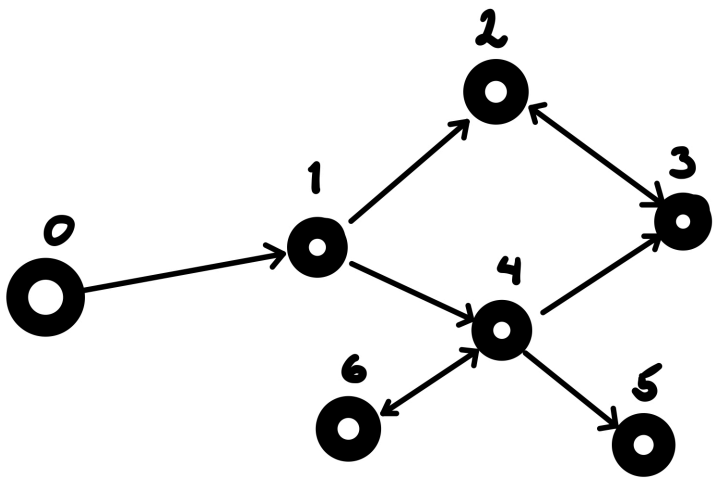
Vad är grafer?

Grafer är när man har en mängd objekt som man visualiserar som noder, och där man kan para ihop vissa par av dessa noder så att de är relaterade. För att visa att ett visst par av noder är relaterade kan man rita kanter som kopplar ihop noder.

Kanter kan vara riktade eller oriktade, och beroende på vilka sorters kanter man har så kan man ha en riktad graf respektive en oriktad graf.

Att spara grafer

Grafer kan se ut på väldigt olika sätt, och för att man ska kunna urskilja en graf från en annan är det bra att man kan på något enkelt sätt representera en graf.



Grannlista (för grafen ovan)

n	grannar
0	1
1	2,4
2	3
3	2
4	3,5,6
5	
6	4

Matris (för grafen ovan om alla vikter är 1)

0	1	0	0	0	0	0
0	0	1	0	1	0	0
0	0	0	1	0	0	0
0	0	1	0	0	0	0
0	0	0	1	0	1	1
0	0	0	0	0	0	0
0	0	0	0	1	0	0

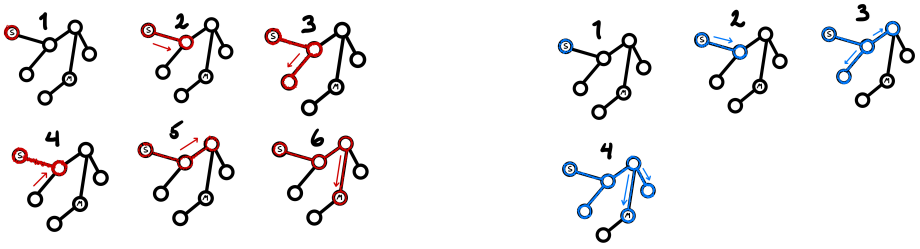
Listorna kan då byggas på följande sätt i kod:

```
grannlista = [[1], [2, 4], [3], [2], [3, 5, 6], [], [4]]

grannMatrix = [[0, 1, 0, 0, 0, 0, 0],
               [0, 0, 1, 0, 1, 0, 0],
               [0, 0, 0, 1, 0, 0, 0],
               [0, 0, 1, 0, 0, 0, 0],
               [0, 0, 0, 1, 0, 1, 1],
               [0, 0, 0, 0, 0, 0, 0],
               [0, 0, 0, 0, 1, 0, 0]]
```

1. Givet en grannlista, skapa den korresponderande grannmatrisen.
2. Givet grannmatrisen, skapa den korresponderande grannlistan.

DFS och BFS

DFS och BFS (Djupet- och Bredden-Först-Sökning) är två algoritmer för att söka igenom grafer. I en DFS går datorn igenom grafen och fortsätter tills den kommer till en återvändsgränd då den går tillbaka till den senaste korsningen med outforskade vägar. I en BFS går datorn igenom alla alternativ i varje korsning. Se Figur 5.1 för exempel på några steg i respektive algoritm i en liten graf.

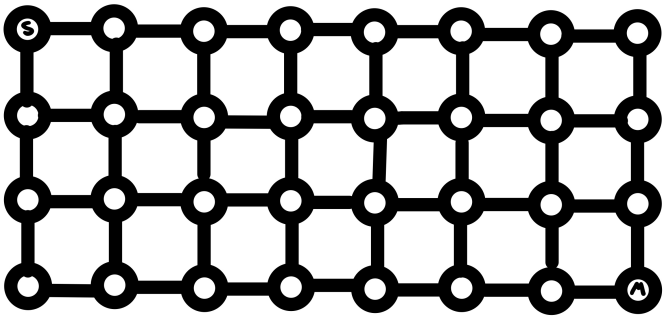


Figur 5.1: Exempel på DFS (vänster sida, rött) respektive BFS (höger sida, blå)

DFS och BFS har olika styrkor och svagheter, ett tydligt exempel är om man vill hitta den kortaste vägen mellan (S)tart och (M)ål. När BFS har hittat en väg till målet är det garanterat den kortaste och man kan avbryta utan att titta igenom resten av grafen. Detta gäller inte för DFS då man måste kolla alla alternativ innan man är säker. Däremot ifall grafen är stor och endast en väg behöver hittas mellan (S)tart och (M)ål är det mycket mer effektivt att ha en DFS. Hitta på grafer där detta blir tydligt om det känns oklart.

3. Implementera en DFS- och en BFS-funktion som söker igenom en graf sparad som en grannlista.

Tänk er följande graf:



En DFS skulle vara väldigt långsam då antalet sätt att ta sig från S till M är enormt! (Observera att det ej finns begränsningar mot att röra sig upp eller till vänster). Men med hjälp av dynamisk programmering kan vi göra även en DFS snabb i detta fall. Om vi på varje nod sparar vilken som är den kortaste vägen dit vi har hittat hittills, behöver programmet inte utforska omvägar om det redan har hittat en kortare.

4. Modifiera er DFS lösning så att den sparar kortaste vägen till alla noder. Alltså ändra på listan `visited` med besökta noder från en boolean array till en array med heltal. Uppdatera överallt där du använder dig av besökt variabeln.

5. Skriv en BFS som tar sig från ordet `majs` till ordet `nice` och som bara får ändra på en bokstav ett steg i taget och endast använder sig av bokstäver från det engelska alfabetet. (Det är okej ifall lösningen tar lite längre tid att lösa problemet, men under 10 sekunder dock!)

6. Använd följande kod för att kontrollera ifall ett naturligt tal är ett primtal för att lösa `https://open.kattis.com/problems/primepath`.

```
def is_prime(n):
    """returns True if n is prime else False"""
    if n < 5 or n & 1 == 0 or n % 3 == 0:
        return 2 <= n <= 3
    s = ((n - 1) & (1 - n)).bit_length() - 1
    d = n >> s
    for a in [2, 325, 9375, 28178, 450775, 9780504, 1795265022]:
        p = pow(a, d, n)
        if p == 1 or p == n - 1 or a % n == 0:
            continue
        for _ in range(s):
            p = (p * p) % n
            if p == n - 1:
                break
        else:
            return False
    return True
```

7. Då grafer kan representeras på många olika sätt förväntas man kunna läsa in dem på många olika sätt. Lös dessa kattisproblem för att både öva på graf-

algoritmer och att ta in data:

- `https://open.kattis.com/problems/birthday` (Notera time-limit!)
- `https://open.kattis.com/problems/flyingsafely`
- `https://open.kattis.com/problems/weakvertices`
- `https://open.kattis.com/problems/grid`
- **Extra:** `https://open.kattis.com/problems/arcticnetwork`

8. Hur kan BFS och DFS skrivas iterativt?

Dijkstras algoritm

Dijkstra algoritmen kan användas om man snabbt vill hitta den garanterat kortaste vägen i en graf med viktade kanter (dvs olika kanter är olika långa). Den går ut på att ha flera aktiva noder som rör sig utåt likt en BFS, men i Dijkstra fortsätter vi på den nod som har kortast väg hittills. Om noden sedan når målet fortsätter algoritmen tills alla andra aktiva noder har färdats längre än den till målet. Det är då garanterat att det är den kortaste vägen.

Hint: För att effektivt hitta nästa nod med minst väg kan `PriorityQueue` användas.

```
from queue import PriorityQueue
pg = PriorityQueue()
pg.put(3) # Lägger till ett element i kön.
pg.put(5)
pg.put(2)
 # Returnerar det minsta elementet som finns i kön och tar bort det.
x = pg.get() # x = 2
y = pg.get() # x = 3
z = pg.get() # x = 5
```

9. `https://open.kattis.com/problems/walkforest`

VI Programmering - Curie

1. Kodgolf

18 juli

Vanliga knep för IO

```
# kort sätt att läsa in en lista med heltal
arr=list(map(int,input().split()))

# kort och enkelt sätt att loopa igenom alla rader input
for line in open(0):
    ...

# kort sätt att printa en hel lista på
print(*arr)
```

Blandade knep

Här kommer lite knep som ni förmodligen kommer tycka är användbara för senare uppgifter. Ifall ni är osäkra på exakt vad en rad gör så rekommenderas det att testa att köra den på er dator för att se vad som händer.

```
# Ifall man vill kolla att nånting är "truthy", till exempel != 0 för heltal
    (eller att en lista inte är tom)
if x:
    ...
# istället för till exempel
if x != 0:
    ...

# Vill man kolla att nånting är "falsy", till exempel 0, eller en tom lista
    kan man använda följande:
if not x:
    ...
# Istället för
if x == 0:
    ...


# användning av '*':
# båda alternativ A och B saker gör samma sak, men använder man '*' så blir
    det kortare
# A)
x=list(range(100))
y=set(x)

# B)
x=[*range(100)]
```

```
y={*x}


# map, med lambda funktioner
a = [2, 3, 5, 7, 11, 13, 1000000007, 1000000009, 1234567807]
b = list(map(lambda x: x**2, a))


# använd:
if 10 <= x <= 20:
    ...
# istället för
if 10 <= x and x <= 20:
   ...


# samma sak fast lite mer extremt:
# använd:
if 10 <= a <= 20 == b // 3 in c is d - e:
    ...
# istället för
if 10 <= a and a <= 20 and 20 == b // 3 and b // 3 in c and c is d - e:
    ...


# filter
odd_numbers = list(filter(lambda x: x%2, range(10)))


# list comprehensions
x = list(range(100))
squares = [elem ** 2 for elem in x]
only_even_numbers = [elem for elem in x if elem % 2 == 0]

# dubbla list comprehensions
[print(i,j) for i in range(10) for j in range(i, i+2)]
```

Korta one-liners

Skriv klart följande program så att de gör det som efterfrågas. För varje uppgift så får ni endast byta ut de 3 punkterna med ett pythonuttryck. Resten av koden skall vara kvar och är tänkt att hjälpa er slippa krångel med att få in IO för på samma rad för tillfället.

1. Skriv ut alla tal i listan som innehåller siffran 5 (i bas 10).

```
numbers = list(map(int, input().split()))
...
```

2. Skriv ut varrannat tal från urpsrungslistan.

```
numbers = list(map(int, input().split()))
...
```

3. Skriv ut alla bokstäver som förekommer i minst ett av orden i listan.

```
words = input().split() # innehåller bara a-z som små bokstäver
...
```

One liners, med inläsning och utskrifter

För följande uppgifter ska ni skriva program som både läser in input från stdin och skriver ut svaret till stdout, samtidigt som ni håller er till en rad kod. Ifall det känns för svårt att skriva allting på en rad kod, börja med att dela upp eran kod på flera rader, och när ni har fått allting att fungera så kan ni försöka slå ihop raderna en efter en.

Tips isåfall: spara en backupfil med eran orginallösning.

4. Titta tillbaks på dina lösningar till de tidigare one-liners som du skrev. Modifiera de lösningarna så att du gör alla steg på en rad.

5. Skriv ett program som läser in en sträng från stdin och sen skriver ut den i antingen stora bokstäver eller i små bokstäver. Programmet skall skriva ut strängen med enbart små bokstäver ifall orginalsträngen innehåller en vokal direkt följt av en konsonant. Annars skall programmet skriva ut strängen med stora bokstäver.

6. Skriv en lösning till problemet FizzBuzz. FizzBuzz är ett väldigt känt problem inom datalogi som går ut på att skriva ut alla heltal, från 1 upp till en viss gräns. Men ifall ett tal är delbart med 3 så ska det ersättas med ordet Fizz, och är det delbart med 5 så ska det ersättas med ordet Buzz. Däremot ifall ett tal är delbart med både 3 och 5 så ska det ersättas med ordet FizzBuzz istället.

Skriv en lösning som skriver ut de första n elementen av FizzBuzz följden, där ni läser in n som ett heltal från stdin.

Lite svårare uppgifter

Nu ska vi fokusera på lite svårare uppgifter. Och trots att det är kul att skriva one-liners, så kan det bli lite krångligt ibland. Därför ska ni nu få skriva program som kan ha hur många rader som helst. Däremot är målet att eran kod ska bli så kort som möjligt. Ni kommer säkert märka att ni kan återanvända samma knep som för one-liners, bara att ni nu kanske har möjligheten till ännu fler kul knep :).

7. Skriv en funktion som tar in ett heltal som parameter och returnerar nånting ”truthy” ifall att heltalet är ett primtal, och nånting ”falsy” annars. Ni kan tänka er att följande kod ska gå att köra för alla positiva heltal som är mindre än 1000:

```
if is_prime(x):
    print(x, "is a prime number")
```

```
else:
    print(x, "is NOT a prime number")
```

Försök att komma under så många som möjligt av följande gränser, där ni räknar antal tecken som är del av eran kod, inklusive def is_prime(x):".

180, 144, 101, 71, 57, 53

8. Skriv en funktion som kollar ifall ett parantesuttryck som består endast utav tecknena "()[]"är balanserat. För att uttryck ska anses vara balanserat så måste varje vänsterparantes kunna matchas med en högerparantes till höger om sig, men det får förekomma andra typer av paranteser däremellan. Dessutom för att en parantes på index *L* i strängen skan kunna matchas med en högerparantes på index *R* i strängen så måste alla andra matchningar antingen vara helt till vänster om A, helt till höger om R, eller helt mellan L och R.

Ett par exempel på parantesuttrykc som är giltiga är ([]) och ([])()[]. Ett par exempel på ogiltiga parantesuttryck är)(, (][) och ([)].

Mål att försöka nå i antal tecken:

297, 256, 212, 174, 155

9. Om du har kommit så här långt, bra jobbat! Grattis ifall du även slog målen i antalet tecken. Nu är det dags att optimera eran koden ännu mer. Hur få tecken kan ni komma ner i? Det går ofta att få mycket kortare kod än vad man först tror. Diskutera gärna med andra deltagare och/eller ledare för att få ett par till idéer om förbättringar.

2. Dataanalys (Curie)

19 juli

Programmering är väldigt användbart ifall vi har stora mängder av data som vi vill analysera. Idag kommer vi undersöka hur vi kan använda oss av Python för att analysera en stor mängd data på ett effektivt sätt!

1. Ladda ner csv-filerna från `https://github.com/hairez/mattekollo-2023`.
2. Spara filerna i en directory du gillar (och som du har lätt tillgång till).
3. Starta en Python fil i samma directory.
4. Ta reda på den exakta directoryn du har filerna. Kör följande kod för att hitta directoryn som filen körs från just nu:

```
import os
print(os.path.dirname(__file__))
```

För att kunna arbeta med datan kan du först öppna filen i Python:

```
file=open("/Users/dirc/svenska-stader.csv")

inp=[*file] #Varje rad från filen blir ett element i arrayen.
file.close()

print(inp[0])
print(inp[1])
```

1. **(a)** Hur många platser (Locality) i Sverige börjar på en vokal?

(b) Hur många av dessa ligger i Skåne (County)?

2. Vilken "County" i Sverige har flest platser (Locality) i datan `svenska-stader.csv`?
3. Vilka 100 platser (Locality) ligger mest i norr i Sverige?
4. Vilka 100 platser ligger närmast Nynäshamn, Stockholm?
5. Hur många städer i US har både militär och är "incorporated"? (Svaret är inte 0. Men ifall du får 0 som svar, tar reda på varför!)
6. Hur många ZIP-nummer har alla städer i Texas?
7. **(a)** Vad är medelvärdet på invånarantal för alla städer i US?

(b) Vilken stad har ett invånarantal som ligger närmast medelvärdet?

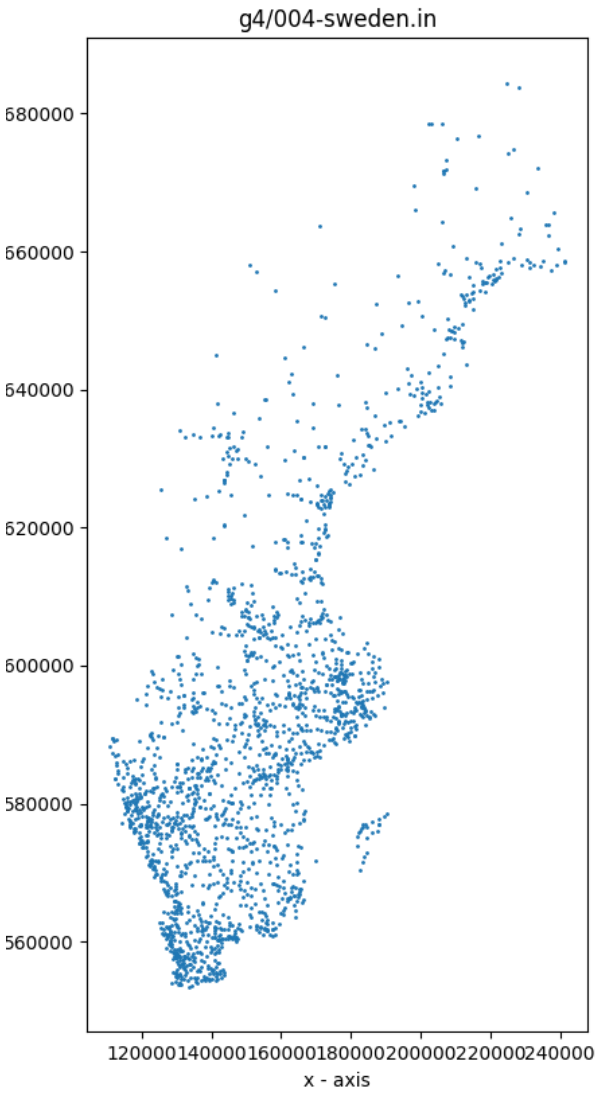
8. Finns det fler invånare i Västra-USA, eller Östra-USA?
9. — **Extra.** Placera ut alla städer i Sverige/US på ett koordinatsystem på ett bra sätt med hjälp av matplotlib.pyplot.

Ledtrådar: `https://github.com/hairez/diploma-project/blob/main/code/tsp/data/manual_testdata/csvDataRetriever.py` och `https://github.com/hairez/diploma-project/blob/main/code/visualizer/plotTestdata.py`

10. — **Extra.** Ändra storleken på alla punkter i koordinatsystemet så att storleken på punkterna representerar invånarantalet.

11. — **Extra.** I `uscities.csv` finns det data av städer i USA som ligger relativt långt borta från alla andra punkter. Lista ut vilken del av USA de tillhör, och filtrera ut dessa ur datan.

12. — **Extra Extra.** Googla upp data på något annat som du vill undersöka. Skapa någon sorts graf eller placera ut datan på ett koordinatsystem.



3. AI för spel med perfekt information

19 juli

Under den här lektionen så kommer vi att skriva en AI som spelar ett brädspel. Den här guiden kommer att utgå från schack som spel, men ni får även göra det för andra spel om ni tycker att det är roligare. Vi kommer att lägga fokus på AI-delen. Därför kommer vi att använda python-paketet chess och att göra allting textbaserat till att börja med.

Representation av spelet

För att kunna testa vårat program så är det väldigt viktigt att skriva kod som sköter reglerna för spelet, och som gärna kan visualisera hur spelet ser ut. Ni får själva välja vilket spel ni vill skriva en AI för. Ett exempel på ett spel ni kan skriva en AI för är schack. Gör ni det i python så kan ni importera paketet *chess*, och bygga koden för AIn på det. Väljer ni något annat spel så är det bra att välja ett spel som det redan finns kod för online eller som ni snabbt kan skriva ihop. Idag ska vi lägga fokus på AI-delen, och försöka undivka att bli fast med buggar i själva spelet.

Här är ett exempelprogram för hur ni kan skriva ut alla möjliga drag från startställningen i schacknotation:

```
import chess
board = chess.Board()
print(board.legal_moves)
```

Det finns även många fler funktioner som stöds av paketet chess. För mer info kan ni ta en titt på `https://python-chess.readthedocs.io/en/latest/`, eller googla på nånting i stil med *python chess*.

Game loop

1. Skriv ihop ett program som visualiserar spelbrädet och tillåter användaren att spela spelet genom att skriva in drag som text. Programmet bör låta användaren skriva in ett drag i taget som en sträng och sen skriva ut ett uppdaterat bräde. Ifall spelet tar slut (någon vinner, eller att det blir oavgjort) så bör erat program skriva ut det.

   Ifall ni vill, så får ni lägga in logik för att kolla att det användaren skrev in är ett giltigt drag så får ni göra det, men det är inte ett krav.

AIn som ena spelaren

2. Uppdatera erat program så att AIn spelar för ena spelaren och gör helt slumpade drag.

3. Implementera en heuristikfunktion för att avgöra ungefär hur bra en ställning

är. Använd sedan denna heuristik för att avgöra vilket drag AIn ska spela. För tillfället kommer vi inte att göra någon rekusion. Istället kommer vi bara att titta på vilka ställningar AIn skulle kunna nå. Utifrån heuristiken så ska AIn sen välja det alternativ som ser bäst ut då. Även med en väldigt enkel heuristik så bör ni märka att AIn kommer att ta pjäser om den får chansen. Testa!

4. Implementera minimax (eller negamax). Ni bör skriva det som en rekursiv funktion som tar in ett state (ett board objekt) samt ett maxdjup som ni får söka till. För första anropet till funktionen så får ni välja ett maxdjup som gör att funktionen tar max en sekund att köra klart. Ju högre djup ni väljer, desto bättre kommer AIn att spela, men den kommer även att ta längre tid.

5. Efter att ni har säkerställt att eran kod fungerar, så är det dags att lägga in alpha-beta pruning till eran kod. Ifall ni gör rätt så kommer ni märka att detta snabbar upp AIn väldigt mycket, vilket tillåter er att öka maxdjupet för rekursionen.

6. Nu är det dags att lägga in ett tidsgräns som AIn får använda per drag istället för att utgå ifrån en maxgräns på antalet rekursioner. Det är ju svårt att veta hur lång tid datorn kommer ta att söka igenom alla möjliga drag för ett visst djup. Därför kommer vi använda en metod som heter iterative deepening. Tanken är att vi först sätter maxdjupet till 1. Ifall vi hinner söka igenom allting till det djupet så ökar vi maxdjupet till 2 och kör om algoritmen igen. Hinner vi klart med det så ökar vi maxdjupet till 3 och kör om allting igen osv.

   Vid något tillfälle så kommer vi att öka maxdjupet för mycket, och då kommer vi kanske behöva vänta länge på att algoritmen blir klar. Därför är det bra att lägga in kod på flera ställen som kollar ifall AIn har använt för mycket tid. Så fort vi märker att vi har använt för mycket tid så kan vi avbryta sökningen. Vi kommer då att slänga bort det drag vi hittar som det bästa, och använda det vi hittade från den tidigare sökningen.

7. Om ni har kommit så här långt har ni förmodligen en AI som spelar hyffsat ok. Det finns dock flera förbättringar vi kan göra. Först och främst kan man alltid jobba på att förbättra heuristiken. Men förrutom det så finns det flera metoder som kan förbättra AIns förmåga att spela spelet. Ett par vanliga metoder som ni kan googla på är:

   - Move ordering.
   - Transposition table.
   - Quiescence search. (Kanske mest applicerbart för schack)

4. GPT-4 och andra externa API:er

20 juli

1. Skriv ett program som frågar användaren om ett grundämne och skriver ut dess kemiska beteckning? Till exempel om man frågar om syre ska programmet svara bokstaven O.

2. Skriv ett program som svarar på regissörer till olika (kända) filmer?

3. Gör en FORK-prompt till GPT för att svara på antingen vilken dag det är idag, eller vem som har namnsdag idag.

4. Skapa en return-prompt till GPT som ger vilken dag (”idag”, ”imorgon”, ”på torsdag”) personen frågar efter (om den inte frågar om en specifik dag så ska den svara ”idag”). Använd sedan detta svar i python för att räkna ut datumet som man ska skriva ut namnsdagen för.

5. Skicka med dagens datum i kontexten så att GPT själv räknar ut vilket datum man vill veta namnsdagen för. Då ska man både kunna fråga om ”Vem har namnsdag imorgon?”, men också ”Vem har namnsdag den 18e juli?” eller ”Vem har namnsdag på julafton?”.

   Bonus, testa med t.ex. 6e januari och uppdatera er `join()` så den fungerar bättre?

6. Skriv ett program som listar namnet på alla badställen i Linköping. Använd följande API:

```
http://waterqualityobserved.linkoping.se/api/v1/WaterTemperature?
api_key=fadae3b1cda840d0b58d498a367dc4ca
```

   För att t.ex. hämta temperaturen för bara Amundebobadet kan du skriva in dess ID (som du får från föregående anrop):

```
http://waterqualityobserved.linkoping.se/api/v1/WaterTemperature/
3590beee-acd8-4e39-aab1-0d5199be1399?api_key=fadae3b1cda840d0b58d498a367dc4ca
```

7. Be användaren om en badplats och skriv ut dess vattentemperatur. Du kan börja med att användaren måste skriva exakt namnet så som det ges av API:et, alltså t.ex. ”Johannelundsbadet Stångån”.

   Men utveckla detta vidare så att man med hjälp av GPT kan fråga mer flytande, som ”Hur varmt är vattnet i Johannelundsbadet?”, eller till och med ”Jag skulle vilja bada i Hjulsbro, hur varmt är det?”.

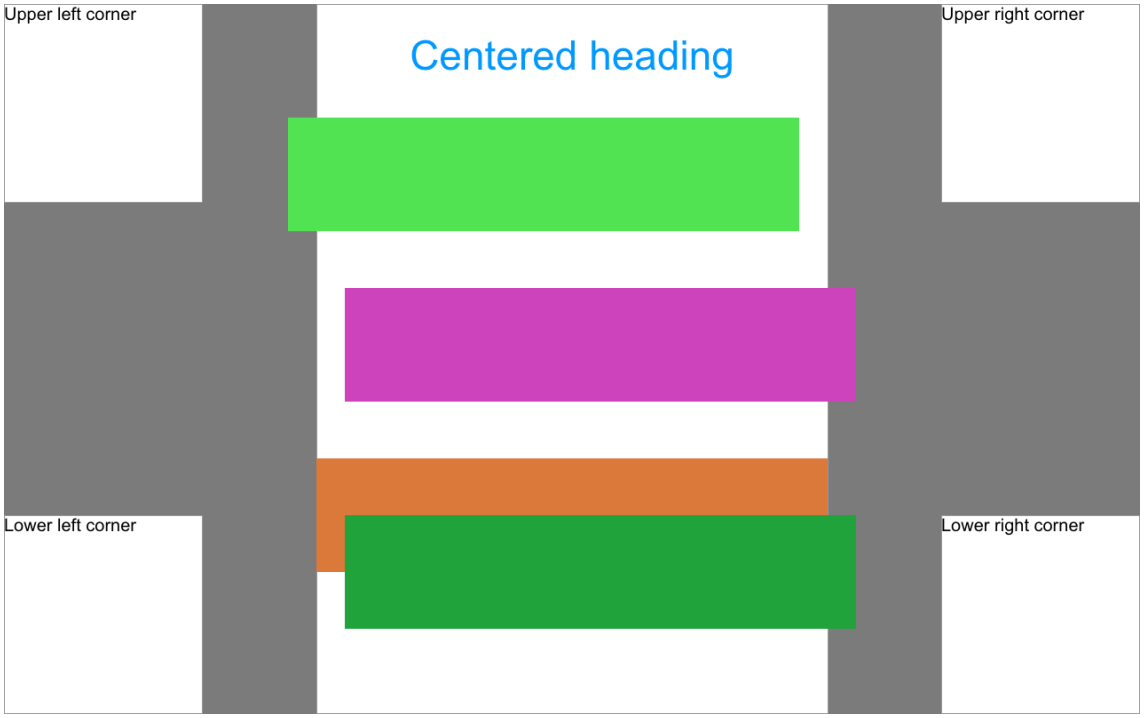
8. Kombinera ovanstående så att den kan svara på både vem som har namnsdag idag, och vad vattentemperaturen är? Använd en FORK-prompt först, för att sedan göra rätt API:anrop, du kan även lägga med badtemperatur eller namnsdag i kontexten till en sista prompt för att svara på frågan med hel mening.

5. Webbutveckling

23 juli

1. Skapa en valfri hemsida, den kan handla om dig själv, mattekollo, ditt favoritsportlag, ditt husdjur eller vad som helst. Använd gärna länkar, bilder och rubriker.

2. Försök efterlikna sidan som är i figur 5.1 så noga som möjligt. Med både CSS och HTML.



Figur 5.1: Boxar

3. Skapa en knapp som man ”inte” ska trycka på och en annan knapp som man ska trycka på. Om man trycker på fel knapp ska flera pop-up rutor komma fram, med innehåll i stil med: ”Jag sa tryck INTE här, lär dig att lyssna...”, ”Jag varnade dig... Tryck inte på OK den här gången, så går det bättre. För de som är döva (blinda) tar jag det igen. TRYCK INTE PÅ OK”, ”OK, detta är det sista meddelandet.”, ”HA! HA! HA! HA! HA! Känn dig blåst, det var inte det sista...”

Trycker man på rätt knapp kan det också komma upp rutor: ”Tack för att du tryckte här...”

4. — **Valfri, skippa om du vill.** Skapa följande:

   a) Gör en funktion som heter `prank`. För att denna funktion ska anropas ska man trycka på en länk i sidan där det står radera din hårddisk...

   b) När funktionen körs ska det komma upp en fråga som frågar om man vill radera hårddisken (vill du radera din hårddisk?). Alternativen är OK och Cancel

   c) Trycker man på OK så ska webbläsaren fråga vilken drive man vill radera (vilken drive vill du radera?). Besökaren ska få fylla i ett fält vilken drive han vill radera, drive C ska vara det som först ska vara ifyllt.

   d) Vad man än trycker på så ska en ruta säga att det inte går att radera sin hårddisken (du kan inte radera din hårddisk.).

   e) Skulle man i början trycka på Cancel så ska en ruta säga att det inte går att radera sin hårddisken (du kan inte radera din hårddisk.) (precis som förut alltså).

5. Skapa en inloggningsruta med lösenord som visar dig en hemlig del av sidan. Kanske genom att länka dig till en dold sida, eller genom att visa en pop-up med hemligt information

6. Gör en norsk miniräknare. Om man till exempel skriver in enkla tal som 0+0, 1+1 eller 0*0 så ska den lagga ihop och få problem. Testa till exempel den jag la upp på `www.technox.se/norsk.html`

7. Gör ett spel där man ska gissa talet mellan 0 och 100. Det skrivs ut i en textarea vad du nyss gissar, huruvida det var för stort eller för litet, och totalt antal gissningar. Förslagsvis skrivs gissningen in i ett <input type=”number” /> fält.

8. Låt sidan visa en random katt, genom att använda Math.random(); Katter kan ni t.ex. ladda ned här: `www.technox.se/katter.zip` Kanske ha en sida som visar ”Dagens katt”, alltså har en specifik bild för varje veckodag.

9. — **Extra.** Ladda ned filerna från denna sida `www.technox.se/mattekollo_webb1.zip`. Skapa CSS filen och försök att styla sidan för att bli så lik figur 5.2 som möjligt. Du får inte ändra något i HTML filen.

10. — **Extra.** Välj en sida på nätet och försök härma den. Några förslag:

    a) `www.vaxtia.se`

    b) `www.savannen.com`

    c) `sites.google.com/view/lisalokteva`

    d) `www.roysmatobar.se`

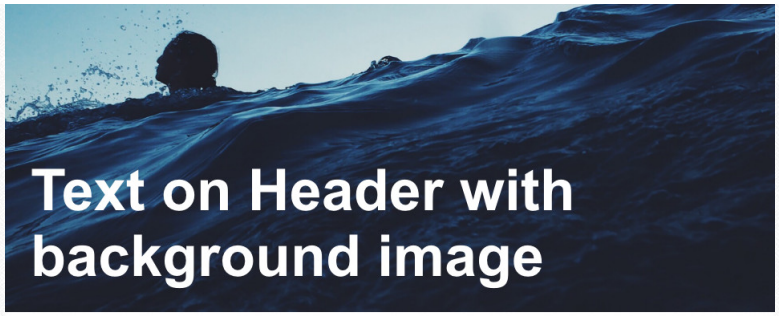


Image post 1

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

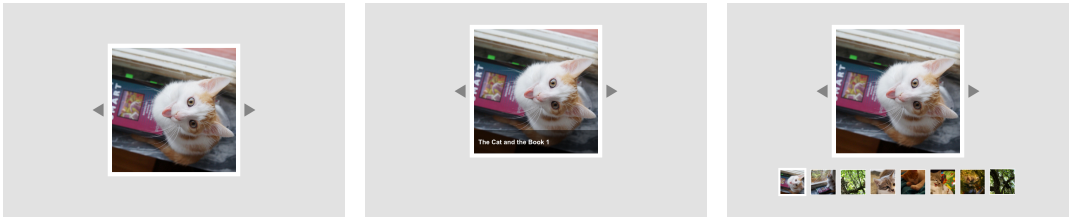
Image post 2

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Figur 5.2: En exempelsida

11. — **Extra.** Du ska nu skapa ett bildspel för katter. Du kan använda bilderna i denna zip-fil om du inte vill googla fram egna katter. `www.technox.se/katter.zip`

a) Skapa en hemsida (HTML och CSS) som ser ut som figur 5.3a.



(a) Bildspel (b) Beskrivningar (c) Tumnaglar

Figur 5.3: Bildspel

Nu ska du göra bildspelet interaktivt. Implementera följande med hjälp av JavaScript, HTML och CSS:

c) Skapa trianglar till höger och vänster om bilden som reagerar när användaren för musen över dem. Du kan till exempel ändra genomskinlighet, färg, storlek etc.

d) Ändra formen av muspekaren till en hand, detta hjälper användaren att förstå att hen kan klicka på triangeln.

e) Ändra bilden som syns i mittenrutan när användaren klickar på trianglarna. Bildspelet ska loopa, dvs när användaren klickar på show next imageöch den sista bilden visas så ska bildspelet börja om med att visa den första bilden igen.

f) Försök att göra koden enkel att modifiera så att fler (eller färre) bilder kan läggas till enkelt. Använd till exempel en array/lista för bilderna.

g) Lägg till bildbeskrivningar på bilderna i en semitransparent box över bilden. Se figur 5.3b

h) Gör så att beskrivningen bara syns när man hovrar över bilden.

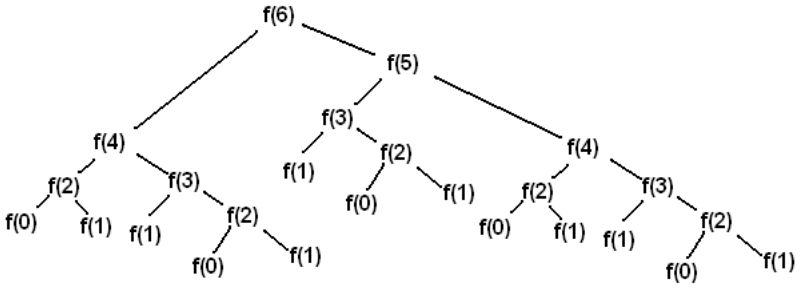
i) Lägg till tumnaglar för alla bilder i bildspelet under den större bilden. Du ska markera den bild som visas just nu, genom att till exempel lägga till en vit ram runt den, eller någon annan markör. Se figur 5.3c. Använd mindre bilder (färre bytes) som tumnaglar så att sidan laddas snabbare (finns i zip-filen med katter).

6. Dynamisk Programmering (Curie)

23 juli

Vad är Dynamisk programmering?

Dynamisk programmering är huvudsakligen när man optimerar en rekursion på ett effektivt sätt så att det tar färre beräkningar att få fram det man vill beräkna. Det är mest effektivt när ett problem kan lösas i mindre delproblem.



Sekvenser:

1. Givet ett positivt heltal n, beräkna $n!$.
2. Fibonacci:
   - **(a)** Skapa en rekursiv funktion som kan räkna ut det n:te Fibonacci talet.
   - **(b)** Lägg till en memoisering i det rekursiva funktionen.
   - **(c)** Lös problemet med en iterativ process istället.
3. Få 100 poäng på följande problem: `https://open.kattis.com/problems/triolingopush`

Ett DP-träd?:

4. Givet ett rutnät med n rader och m kolumner. Om du börjar i rutan i det övre vänstra hörnet, så får du antingen flytta dig nedåt, eller åt höger. Beräkna antalet olika sätt du kan ta dig till det nedre högra hörnet.

5. Samma problem som problem 4, men vi kallar koordinaterna för hörnet längst ner till höger $(1,1)$, och rutan längst upp till vänster har koordinaterna (m,n). Man får fortfarande endast röra dig nedåt eller åt höger, men du får inte gå på rutor vars koordinater (x,y) uppfyller $x-y \equiv 3 \pmod 7$.

6. `https://open.kattis.com/problems/tight`
7. `https://open.kattis.com/problems/gourmeten`
8. `https://open.kattis.com/problems/variedamusements`

DP i listor:

9. Givet en lista a med $0 \leqslant N \leqslant 10^3$ heltal och elementen $[a_0, a_1, \cdots, a_N]$, får du välja ut en delmängd ur listan a så att den nya listan har elementen $[a_{i_1}, a_{i_2}, \cdots, a_{i_k}]$, men även att $i_1 < i_2 < \cdots < i_k$ samtidigt som $a_{i_1} < a_{i_2} < \cdots < a_{i_k}$. Hitta den största antalet element som kan finnas i en sådan lista.

   Till exempel skulle en lista med flest antal element som är delmängd ur $a = [10, 2, 5, 20, 6, 15, 12]$ vara $[2, 5, 6, 15]$.

10. `https://open.kattis.com/problems/skolvagen`
11. `https://open.kattis.com/problems/centsavings`
12. `https://open.kattis.com/problems/trainsorting`

Blandade extra uppgifter:

13. `https://open.kattis.com/problems/nikola`
14. `https://open.kattis.com/problems/mailbox`
15. `https://open.kattis.com/problems/knapsack`
16. `https://po2punkt0.kattis.com/problems/pocamp23.botfoll`

Harry's gömda arkiv:

17. `https://open.kattis.com/problems/dutub/en`
18. `https://open.kattis.com/problems/downpayment`
19. `https://open.kattis.com/problems/welcomehard`
20. `https://open.kattis.com/problems/exactchange2`
21. `https://open.kattis.com/problems/colorland`
22. `https://open.kattis.com/problems/mububa`
23. `https://open.kattis.com/problems/silverstarstandsalone`
24. `https://open.kattis.com/problems/applescherriesmangos`

7. Introduktion till C++

24 juli

Under dagens lektion kommer vi att gå igenom lite om hur man skriver kod i C++. C++ är ett väldigt kraftfullt språk, och det kan därför vara nice att ha åtminstone testat på det. Tidigare har vi ju mest fokuserat på python på mattekollo, men ifall man till exempel vill skriva ett väldigt stort program, och/eller vill att det ska köra snabbt på datorn så är C++ kanske ett bättre val.

Korta uppgifter för att komma igång

1. Skriv ett program som läser in ett heltal från stdin. Skriv ut summan av alla positiva heltal som är delbara med 0, men som är mindre än det tal du skrev in. Hur lång tid tar det att köra ditt program för olika stora tal? Försök att hitta hur högt du kan komma ifall ditt program ska köra i cirka en sekund. Ifall ni vill så kan ni jämföra det med python för att få en intuitiv förståelse för hur snabbt det kan gå i C++.

2. Skriv ett program som läser in ett heltal och skriver ut det tecken som har samma ascii-värde som det tal du skrev in.

Datastrukturer i C++

Det finns en hel del datastrukturer i C++ som är del av standarbiblioteket. Vi ska nu testa att göra ett par småsaker med varje för att se hur de kan användas.

3. Skriv en funktion som konkatenerar två stycken vectors. Testa att köra funktionen med ett par olika vectors och skriv ut resultatet för att se till så att det fungerar.

4. Skriv en funktion som använder sig utav en map för att räkna ut hur många gånger varje sak förekommer i en vector. Funktionen bör ta in en vector som input och returnerar antingen en map eller en `unordered_map` där ni mappar varje element till dess frekvens.

5. Skriv ett program som först läser in alla element som användaren skriver in via stdin och lägger in dem i ett set. Kolla sedan upp vilket tal som är närmast 26. Tips: det finns ett par funktioner som heter `lower_bound` och `upper_bound` för set. Kolla upp dem online för exempel på hur man kan använda dem ifall det behövs.

   Testa nu att byta ut ditt set till ett `unordered_set`. Varför fungerar det inte längre?

Referens eller inte?

I C++ kan vi antingen skicka variabler till en funktion som en referens eller som en kopia. Vilket som är bäst att använda beror på vad vi vill göra. Nu ska vi utforska lite exempel där vi övar på detta.

6. Skriv en funktion som gör att följande kod fungerar:

```
vector<int> v({1,2,3,4,10,20,30,40});
get_5th_element(v) = 110;
```

7. Skriv en funktion som räknar ut summan av varannat element i en vector.

En egen datastruktur

Skriv en struct för en matris som innehåller doubles. Representera din matris som en vector av vectors för att göra det så lätt som möjligt. Din struct bör innehålla följande saker:

8. Minst en konstruktor så att ni kan skapa en matris där ni anger hur hög och bred den ska vara.

9. En funktion som returnerar det element som finns på rad `i` och kolumn `j` i matrisen. Ni får själva välja om ni vill returnera detta som en referens eller inte. Det beror på vad ni tycker är mest praktiskt för användning senare.

10. En funktion som adderar ihop två matriser och en annan funktion som subtraherar två matriser. Tips: ni kan använda en hjälpfunktion för att minimera hur mycket kod ni måste skriva.

11. En funktion som skriver ut varje rad av matrisen.

Övningsuppgifter på kattis

Här kommer ett par uppgifter på kattis där ni dels kan träna lite mer på att skriva kod i C++ och där ni även kan öva på eran problemlösningsförmåga. Ifall ni inte vill skriva in hela url:en i webbläsaren så kan ni även söka på den sista delen av url:en direkt på kattis.

12. https://open.kattis.com/problems/annoyedcoworkers
13. https://open.kattis.com/problems/anotherbrick
14. https://open.kattis.com/problems/anotherdice
15. https://open.kattis.com/problems/eulerianpath
16. https://open.kattis.com/problems/beeproblem
17. https://open.kattis.com/problems/maxcolinear

8. Pussellösare

24 juli

Under dagens lektion ska vi konstruera en pussellösare till ett lämpligt logikpussel. Ni vår välja vilket pussel ni vill lösa, men lättast är förmodligen klassiska pussel så som sudoku. Till att börja med kan det även underlätta att begränsa pusslets storlek (t.ex. sudoku på ett 4×4 rutnät). Ni kan även välja ut ett valfritt pussel från kluringtävlingen och skrive ett program som löser det.

Representation av pusslet

1. För att kunna lösa ett pussel krävs först att vi kan representera pusslet i datorn. Välj ett spel och leta upp ett exempel av erat pussel online. Skapa en textfil som innehåller en textversion av pusslets innehåll (tänk på att all information om pusslet måste bevaras i textfilen ni skapar. Dvs ni måste kunna lösa pusslet enbart utifrån den information ni sparar i textfilen). Ifall ni skriver en lösning för sudoku så är det förmodligen lättast att representera pusslet som ett rutnät med siffror i sig.

2. Skriv kod för att läsa in textfilen ni precis skapade. Se sedan till att eran kod lagrar information på ett lämpligt sätt. T.ex. kan ni spara en sudoku som en matris av siffror. Ifall ni har ett mer komplicerat pussel kan ni exempelvis spara mer information med hjälp av en klass.

   Då erat program kanske blir ganska långt under denna lektion kan det vara smart att dela upp programmet i funktioner. Ni måste inte, men det är smart att ha en funktion som sköter inläsningen.

3. Skriv en funktion som skriver ut pusslet på ett fint sätt. Tänk på att det ska vara lätt att läsa, så att ni snabbt kan kontrollera ifall det ser rätt eller fel ut. Ni kommer senare att använda den här funktionen för att skriva ut era lösningar. Ifall ni har buggar och liknande så är det även bra med en print funktion för att debugga vad erat program testar för nånting.

4. Testa att erat program hittills fungerar. Kan ni läsa in en textfil med ett pussel och sedan skriva ut samma sak?

Lösaren

Nu är det dags att skriva kod för att faktiskt lösa pusslet. Vi kommer göra detta genom en rekursiv funktion som ”gissar” upprepade gånger. Ifall en gissning leder till att det blir omöjligt att lösa pusslet efteråt, kan vi då konstatera att den gissningen var dålig. Trotts att det finns en otroligt stor mängd alternativ att gissa på är förhoppningen att vi kan få ett snabbt program. Detta kommer vi att uppnå genom att gissa på ett smart sätt.

5. Till att börja med behöver vi ett sätt att avgöra ifall pusslet är löst eller inte. Skriv en funktion som avgör detta. Funktionen bör returnera True ifall pusslet

är avklarat och False annars. Utan denna funktion skulle vår lösare inte veta när den ska sluta gissa.

6. Skriv en solve-funktion. Givet ett pussel ska denna funktion returnera True ifall det går att lösa pusslet. Ifall det går att lösa pusslet ska även pusslet uppdateras. Ifall pusslet inte går att lösa ska ni returnera False och pusslet ska inte ha förändrats.

Som exempel, låt säga att ni har valt att representera en Sudoku som en 9×9 matris. Innan ni anropar solve-funktionen kommer det då att finnas vissa delar av matrisen som inte är ifyllda. Ifall eran solve-funktion returnerar True bör matrisen vara helt fylld med siffror i slutet. Ifall funktionen returnerar False bör matrisen fortfarande ha ofyllda delar i slutet.

Inuti funktionen: välj den tomma ruta som är högst upp. Ifall det finns fler tomma rutor som delar plats som högsta, ta den som är längst till vänster utav dem. Loopa sedan igenom alla möjliga alternativ för den rutan och använd dem som gissningar, ett i taget. För varje gissning: fyll i rutan med eran gissning och gör sedan ett rekursivt anrop till solve. Ifall solve returnerar False, testa nästa alternativ osv. Ifall inget av alternativen fungerade, returnera False.

7. Testa att köra erat program. Vad händer? Varför? Finns det något vi kan göra bättre?

Ifall ingenting verkar hända kan det vara bra att försöka leta reda på varför det är fallet. Antingen har ni en bugg nånstans i eran kod, eller så kanske detta är förväntat.

Ifall erat program klarar av att lösa uppgiften är det värt att tänka på ifall tiden det tog känns rimlig. Testa med ett par olika pussel av olika svårighetsgrader.

8. Skriv en funktion som räknar ut ett bra ställe att gissa på. Ni får själva välja vad som menas med bra, men funktionen bör returnera den ruta ni tror är smartast att gissa på.

9. Modifiera solve-funktionen så att den gissar på eran ”smarta ruta” som ni räknade ut i förra uppgfiten.

10. Testa återigen att köra erat program. Lyckas ni lösa pusslet? Hur lång tid tar det? Ifall eran kod inte löser pusslet inom en hyffsat kort tid är det förmodligen dags att göra två saker: felsöka koden och hitta smartare rutor att gissa på.

Mätning av programmets förmåga

Nu när vi har ett program som klarar av att lösa vårt pussel är det dags att se hur mycket det klarar av. Klarar den av även de svåraste pusslena? Hur många pussel kan vi lösa inom ett kort tidsintervall?

11. Inför en räknare för antalet rekursiva anrop som görs. Detta kan vara en global variabel som uppdateras i början av varje solve-anrop. Detta kommer ge ett mått på hur bra våra gissningar är: ju färre anrop, desto bättre gissningar.

12. Kör programmet på ett par pussel och tajma hur lång tid det tar. Skriv även

ut hur många anrop till solve som görs.

Extra problem

13. Försök att få ner antalet rekursiva anrop så långt som möjligt. Tänk dock på att programmet fortfarande ska vara minst lika snabbt. Det är ju inte värt att skriva en ”smartare” algoritm ifall den är långsamare.

14. Leta upp ett liknande pussel till det pussel du lyckades lösa. Till exempel om du valde sudoku kan du nu testa att lösa diagonalsudokus (vanliga sudokuregler förutom att diagonalerna även måste innehålla alla tal från 1 till 9 exakt en gång). Kopiera din lösare till en ny fil. Har du strukturerat koden på ett bra sätt behöver du endast ändra på den funktionen som kollar ifall pusslet är löst och den funktionen som letar upp var du ska gissa.

    Efter modifieringarna, testa ifall programmet klarar av den nya typen av pussel.

15. Välj en typ av pussel som inte går ut på att fylla i ett rutnät med siffror. Till exempel nonogram eller staketpussel (slitherlink på engelska). Vad behöver man nu ändra på för att lösa ett sådant pussel? Tänk lite och försök sedan att skriva en lösare på liknande sätt som tidigare.

16. Kan man använda en lösare för att skapa pussel? Försök tänka ut ett sätt att modifiera din kod så att du producerar pussel istället för att lösa pussel. Hur kan du modifiera svårighetsgraden på de pussel du skapar?

Häftet innehåller lektionsmaterial från tre nivåindelade grupper i både matematik (åk 6–8, medel, avancerad) och programmering (Ada, Babbage, Curie).

Tack till alla ledare och deltagare!