UPPGIFT 1 – KANINER

Kaniner är bra på att föröka sig. I den här uppgiften tänker vi oss att det finns obegränsat med hannar och att inga kaniner dör. Vi ska försöka simulera hur många kaninhonor det finns efter varje månad om det vid tiden 0 finns en enda nyfödd kaninhona. Vi antar att det tar x månader innan varje kaninhona är könsmogen och att hon från och med denna ålder (x månader) varje månad föder en kull innehållande y kaninungar av honkön.

Indata: Programmet ska fråga efter honornas könsmognadsålder $x, 1 \leq x \leq 5$, samt antal honor per kull $y, 1 \leq y \leq 5$.

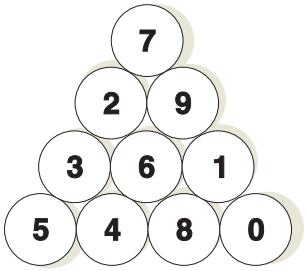
Utdata: Programmet ska skriva en rad för varje månad under totalt 10 månader

Exempel:

```
Könsmognadsålder ? 2
Honor per kull   ? 1
Efter  1 m:   1
Efter  2 m:   2
Efter  3 m:   3
Efter  4 m:   5
Efter  5 m:   8
Efter  6 m:  13
Efter  7 m:  21
Efter  8 m:  34
Efter  9 m:  55
Efter 10 m:  89
```

Förklaring: Från början finns 1 nyfödd hona, låt oss kalla henne Lisa. Efter 1 månad är Lisa fortfarande ensam. Efter 2 månader har Lisa fått en kull. Efter 3 månader finns 3 honor eftersom Lisa fått ytterligare en kull. Efter 4 månader finns 5 honor eftersom både Lisa och hennes första unge har fått varsin kull med en hona i varje.

UPPGIFT 2 – SIFFERTRIANGELN



FIGUR 1.

De tio siffrorna 0...9 har i figur 1 ordnats i en triangel. Summan av de fyra talen utefter var och en av triangelns tre sidor är alla 17. Talet 6, i mitten, deltar inte i någon av summorna.

Skriv ett program, som för en given *centrumsiffra*, beräknar antalet sätt på vilket man kan ordna talen i triangeln, så att summorna av talen utefter sidorna alla blir lika.

Indata: Programmet frågar efter den siffra, som ska vara placerad i triangelns centrum.

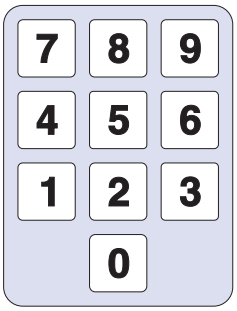
Utdata: Antalet möjligheter att placera de övriga 9 siffrorna så att de tre summorna blir lika.

Exempel:

```
Centrumsiffra ? 6
Det finns 720 möjligheter
```

Förklaring: Här räknar vi alla arrangemang som olika, även om det finns speglingar eller rotationer som påminner om varandra.

UPPGIFT 3 – TELEFONNUMMER



FIGUR 2. *Så här är siffrorna placerade på Lars telefon*

Lars ska byta telefonnummer. Telefonbolaget har givit honom en obruten följd av lediga telefonnummer att välja mellan, men han behöver hjälp av dig. Lars tycker nämligen inte om när telefonen ringer så han vill ha det nummer som kräver den längsta förflyttningen av fingret på en knapptelefon, i hopp om att detta ska avskräcka folk från att ringa. Om det finns flera nummer som ger den längsta sträckan vill han ha det nummer som har flest olika siffror, eftersom det då är större chans att folk glömmer bort numret.

Knapparnas placering visas i figur 2. För att beräkna den fingerförflyttning som krävs för att slå ett nummer startar vi med fingret på första siffran i numret. För att komma till nästa siffra flyttar vi fingret så få steg som möjligt, där varje steg kan vara en knapp *uppåt*, en knapp *neråt*, en knapp *åt höger* eller en knapp *åt vänster*. Den totala förflyttningen för ett n-siffrigt telefonnummer är en summa av $n - 1$ sådana minimala förflyttningar. Exempelvis kräver numret 722019 totalt $3 + 0 + 1 + 2 + 4 = 10$ steg.

Indata: Programmet ska fråga efter ett start- och ett slutnummer. Dessa och alla mellanliggande nummer är de lediga numren. Du kan räkna med att start- och slutnumret har samma antal siffror och att inget av dem börjar med 0. Antalet siffror kan variera mellan 2 (i Älgträsk) och 7 (i Stockholm).

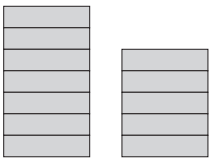
Utdata: Programmet ska skriva det telefonnummer som kräver flest förflyttningssteg. Om flera olika nummer ger detta maximala antal steg, ska du välja det som innehåller det största antalet olika siffror. Om det finns flera nummer med både maximalt antal steg och maximalt antal olika siffror kan du skriva vilket som helst av dem.

Exempel:

```
Startnummer ?  462
Slutnummer  ?  466
Bästa telefonnumret: 462
```

Förklaring: Både 462 och 464 kräver 4 steg men 462 innehåller tre olika siffror medan 464 bara innehåller två. De andra numren kräver 3 steg (463 och 465) respektive 2 steg (466) och är sålunda sämre.

UPPGIFT 4 – SPEL MED BRICKOR



FIGUR 3.

Figur 3 ska föreställa två högar av brickor. I den ena finns 7 och i den andra 5. Två spelare turas om att göra drag, där det, för att vinna gäller att ta bort den eller de sista brickorna.

I varje drag kan spelaren vid draget, antingen

- ta bort *så många brickor* han vill från *en hög*. Spelaren väljer vilken.
- eller ta bort så många brickor han vill från *båda högarna*, men då *lika många* från varje

Skriv ett program som avgör om en given ställning är en *vinst-* eller *förlustställning* för den spelare som ska börja, under förutsättning att båda spelarna spelar optimalt.

Input: Antalet brickor n_1 och n_2 i de båda högarna $1 \leq n_1, n_2 \leq 10$.

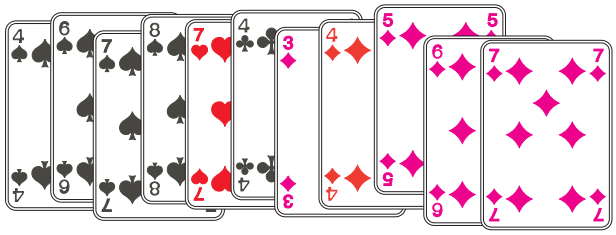
Utdata: Texten `VINST` om inmatad ställning är en vinstställning eller texten `FÖRLUST`, om det är fråga om en förlustställning för spelaren som ska börja.

Exempel:

```
Antal brickor i den ena högen   ? 7
Antal brickor i den andra högen ? 5
VINST
```

Förklaring: Spelet som presenterats ovan kan inte sluta *oavgjort*. Detta betyder att varje ställning är en *vinst-* eller *förlustställning* för spelaren vid draget. Den som har en *vinstställning* kan aldrig bli tvingad till ett drag, som leder till en *förlustställning* för denne, på samma sätt, som att den spelare som har en *förlustställning*, inte kan finna något drag som leder till en *vinstställning* för denne.

UPPGIFT 5 – GIN RUMMY



FIGUR 4.

Gin Rummy är ett kortspel för två spelare, där en vanlig kortlek används. Varje spelare får 10 kort i given och målet är att kombinera dessa kort i *bilder*.

Innan vi går vidare i texten definierar vi kortens beteckningar. Varje kort beskrivs med två tecken. Det första betecknar *färg* och det andra *valör*. I tredje raden ges kortens *belastningsvärde*.

Spader	Hjärter	Ruter	Klöver	Äss	2	3	4	5	6	7	8	9	10	Knekt	Dam	Kung
S	H	D	C	E	2	3	4	5	6	7	8	9	T	J	Q	K
				1	2	3	4	5	6	7	8	9	10	10	10	10

En *bild* är något av följande två alternativ:

- 3 eller 4 kort av samma *valör*. Till exempel tre kungar, `HK SK DK` eller fyra femmor `S5 C5 D5 H5`.
- 3 eller fler kort i *sekvens* i samma färg. Till exempel `C3 C4 C5`, `D8 D9 DT DJ` eller `CE C2 C3 C4 C5`. Ässet är det lägsta kortet i en sekvens och Kungen det högsta. Man kan inte gå ”runt hörnet”, `HD HK HE` är därför ingen giltig sekvens.

Spelaren vid tur tar upp ett kort från talongen och har just då 11 kort på handen. Genom att kombinera 10 av dessa kort i bilder på ett så fördelaktigt sätt som möjligt och kasta ett valfritt kort på *högen* försöker spelaren minimera handens sammanlagda *belastningsvärde* – de sammanlagda belastningsvärdena hos de kort, som inte ingår i bilder och som inte heller är det kastade kortet.

Skriv ett program som tar emot uppgift om handens 11 kort och som kombinerar dem på ett sätt som minimerar det sammanlagda belastningsvärdet.

Indata: Handens utseende i form av 11 strängar, en för varje kort. Strängarna består av två tecken enligt reglerna ovan.

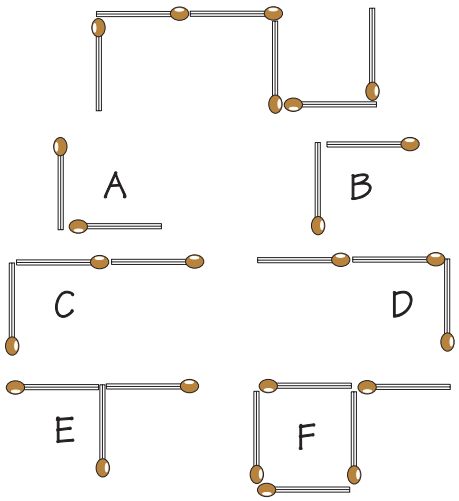
Utdata: Det minsta sammanlagda belastningsvärdet, som kan åstadkommas för inmatad hand.

Exempel: för handen i figur 4, `S4 S8 D5 S6 H7 D3 D4 D7 C4 S7 D6`

```
Kort  1 ?  S4
Kort  2 ?  S8
...
Kort 11 ?  D6
Belastningsvärdet 3
```

Förklaring: Genom att kasta `H7` och kombinera bilderna `S4 D4 C4`, `S6 S7 S8` och `D5 D6 D7` blir endast `D3` över vilket ger belastningsvärdet 3.

UPPGIFT 6 – TÄNDSTICKSFIGURER



FIGUR 5. *Rad 1: En godkänd tändsticksfigur med 6 stickor. Rad 2: A och B anses identiska i detta problem. Rad 3: C och D är däremot inte identiska. Rad 4: E är grenad och därför ej godkänd. F vidrör sig själv och är därför heller inte godkänd*

Om man lägger ett antal tändstickor efter varandra som på bilden ovan uppstår ormliknande figurer. I den här uppgiften ska du svara på hur många olika figurer du kan konstruera med ett givet antal tändstickor.

Endast räta vinklar får förekomma och figuren får inte vara *grenad* eller *vidröra* sig själv.

- Om det går att rotera en figur så att den blir identisk med en annan anses dessa två figurer vara samma, och ska därför bara räknas en gång (se A och B i figuren).
- Om man måste spegelvända den ena figuren för att den ska bli identisk med den andra är figurerna däremot olika (se C och D i figuren).

Indata: Programmet ska fråga efter antal tändstickor n, där $1 \leq n \leq 15$

Utdata: Programmet ska skriva ut hur många olika figurer som kan konstrueras med n tändstickor. Detta tal är alltid mindre än en miljon.

Exempel:

```
Antal tändstickor ? 3
Möjliga figurer ? 6
```