UPPGIFT 1 – FULL TANK

När man tankar bilen på MacMack tillämpas *tiokronorsavrundning*. Om till exempel beloppet, oavrundat, slutar på 164.99 kronor avrundas det nedåt till 160 kronor, medan beloppet 165.00 avrundas uppåt till 170 kronor.

Skriv ett program som tar emot uppgifter om tankad *volym* (alltid ett heltal antal liter) och motsvarande, avrundade, pris i kronor för $n, n \leq 5$ tankningar och som med hjälp av dessa data beräknar priset för den senaste tankningen (given volym i liter). Ett körningsexempel:

```
Antal tankningar ? 3
Volym (liter) ? 32
Pris (kronor) ? 210
Volym (liter) ? 24
Pris (kronor) ? 150
Volym (liter) ? 38
Pris (kronor) ? 240
Senaste tankning (liter) ? 43
Du ska betala 280 kronor
```

Bensinpriset antas vara det samma för samtliga tankningar och tillhör intervallet $[6.00, 15.00]$ kronor/liter, i kronor och hela ören. Testdata är alltid så valda att de bara finns ett korrekt svar. Literpriser mellan 6.41 och 6.44 kr/liter, i testexemplet, ger alla 280 kr.

UPPGIFT 2 – TÅGET

```
MPRP  PPRP  MMPP  PRPP  MPPP  PPPP  MMPG
PRPG  MPPG  PPPG  MPGG  PGGG  PPGG
```

FIGUR 1. *Dessa 13 tåg kan skapas med fyra vagnar*

Tåg kan genom olika vagnar komponeras på olika sätt. Här ska du skriva ett program som tar reda på hur många. Våra tåg kan innehålla fyra olika typer av vagnar: *postvagn (M), personvagn (P), restaurangvagn (R)* och *godsvagn (G)*. Loket räknas inte till vagnarna och ingår därför inte i våra kompositioner.

Följande regler gäller för sammansättningen:

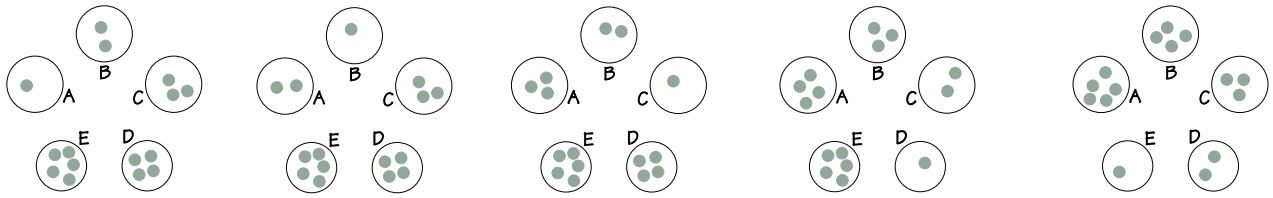
- *Postvagn* kan bara inleda tåget, kopplas direkt efter loket eller efter en annan postvagn. Det kan högst finnas två postvagnar, men behöver inte finnas någon.
- *Restaurangvagn* måste alltid befinna sig mellan två *personvagnar*. Det kan finnas hur många som helst men behöver inte finnas någon.
- Det måste finnas minst en *personvagn* och kan finnas hur många som helst.
- *Godsvagn* kopplas alltid sist i tåget. En godsvagn kan endast ha en annan godsvagn efter sig. Det får finnas högst tre godsvagnar, men behöver inte finnas någon.

Programmet ska fråga efter hur många vagnar tåget innehåller och därefter beräkna och skriva ut antalet olika tåg som kan komponeras. Körningsexempel:

```
Antal vagnar: 13
1042 olika tåg kan skapas
```

Antalet vagnar i testerna kommer att vara ≤ 30

UPPGIFT 3 – MANCALA



FIGUR 2.

Mancala är ett urgammalt spel från Afrika, från vilket vi lånar principerna i denna uppgift. Figur 2 visar 5 ställningar. I varje ställning ser vi 5 *skålar* i vilka ligger ett varierat antal *frön*. Längst till vänster ser vi exempel på en *utgångsställning* och längst till höger exempel på en önskad *slutställningen*. För att nå slutställningen behöver 4 drag göras.

Ett drag består av följande delar:

1 Välj ut en skål och ta upp samtliga frön som finns i den.

2 Välj en riktning, *medurs* eller *moturs*.

3 *Så* först ett frö i den skål du just tömde. Så sedan ett frö i var skål, utefter den riktning du valde, så långt de räcker.

Upprepa sedan punkterna $1-3$ tills önskad slutställning uppnås.

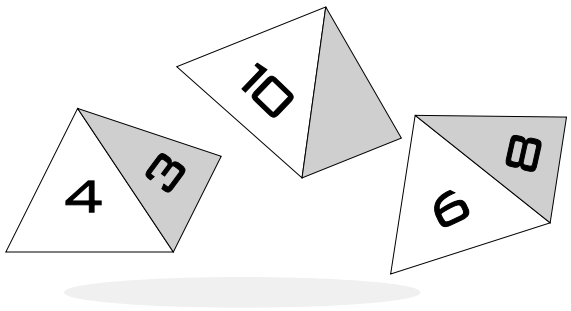
I exemplet i figur 2 utförs följande drag: Först väljer vi skål B och sår i riktning moturs. I nästa drag väljer vi skål C och sår återigen moturs. I tredje draget väljer vi skål D och riktning moturs. I fjärde och sista draget hämtar vi fröna från skål E. Nu kan vi välja vilken riktning som helst, resultatet blir ändå det samma. Därmed har vi från utgångsställningen $1,2,3,4,5$ nått slutställningen $5,4,3,2,1$ på fyra drag (skålarnas innehåll presenteras i ordning A, B, C, D, E).

Skriv ett program som tar emot en utgångsställning och en slutställning och som tar reda på det minsta antalet drag som krävs för att nå målet. I våra tester kommer det aldrig att behövas fler än 6 drag. Dialogen för vårt exempel:

```
Utgångsställning: 1 2 3 4 5
Slutställning   : 5 4 3 2 1
Det krävs 4 drag
```

Det största antalet frön i en skål i utgångsställningen är 10.

UPPGIFT 4 – PYRAMIDTÄRNINGARNA



FIGUR 3.

I den här uppgiften använder vi tre *tetraedrar*, fyrsidiga kroppar, som tärningar. En kastad tedraedertärning läses av genom att lyfta upp den och ta reda på talet i botten. De tillsammans tolv sidorna har var och en ett unikt tal, 1...12. Vi får i filen `tarning.in` givet en serie av kast med de tre tärningarna och ska genom denna avgöra vilka tal som finns på samma tärning.

Vår serie består alltid av 20 kast, 20 rader i filen, med samtliga de tre tärningarna. Tärningarna läses av i godtycklig ordning. Här de 20 raderna i filen, som av utrymmesskäl här återges i tre spalter:

```
 2  9  7   |  11  4 10   |  12  4  8
 1 10 11   |   3  5  7   |   3 12  7
 5  3  8   |   5  3 11   |  11  5  1
 7  4  5   |   7 10  4   |  11  4  5
10  8  2   |   8  5  3   |   1  8  9
 1 12  8   |   6  9  1   |   1  7 10
 6 10  4   |   8  4 10   |
```

Utskriften ges som tre rader. Ordningen är godtycklig:

```
Tärning:  3  2  4  1
Tärning:  5  9 10 12
Tärning:  6  8  7 11
```

Kastserien kommer alltid vara sådan att det endast finns ett unikt svar (bortsett från ordningsföljden).

UPPGIFT 5 – UTSKRIFT

Pelle har en text inskriven på datorn som han nu vill redigera och skriva ut på skrivaren. Han vill ha utskriften i ett teckensnitt som är ”monospaceat”. Det vill säga, alla tecken har samma bredd (tar upp lika stor plats på en rad). Dessutom vill han anpassa utskriften till en given radlängd, n tecken/rad.

Han inser att det knappast är möjligt att få alla rader att innehålla exakt n tecken, eftersom det ska vara exakt **ett** mellanslag mellan två ord. Han är dock nöjd om han lyckas formatera utskriften, så att han kommer så nära det önskade resultatet som möjligt. Som mått på resultatet definierar han avvikelsen i antal tecken hos den ”sämsta” raden från de önskade n tecknen. Det är detta mått han nu vill minimera.

Skriv ett program som läser texten från filen `text.in` och skriver ut den med radbrytningar på sådana ställen att ovanstående mått minimeras.

Indata: Filen `text.in` inleds med en rad som anger $n, 5 \leq n \leq 80$. På nästa rad finns ett tal $k, k \leq 1000$ som anger antalet ord, inget längre än 20 tecken. Därefter följer k rader med ett ord på varje rad. Orden innehåller inga mellanslag, men kan innehålla skiljetecken som i så fall behandlas som en del av ordet och inte får säras från detta. Texten som du ska skriva ut består av dessa k ord i just denna ordning.

Utdata: Programmet ska skriva ut texten med radbrytningar. Den givna exempelfilen `text.in` (med n=30 och k=32) ska ge följande utskrift:

```
Hej mor och far! Här är det varmt
och skönt, solen lyser var dag
och det är varmt i vattnet! Just
nu ligger jag på stranden och
läser en bok. Hälsningar Pelle
```

Det är alltså den första raden som här är den sämsta. Den har blivit tre tecken för lång (33 tecken). De övriga fyra raderna har längder $30, 32, 29, 30$ det vill säga avvikelserna $0, 2, 1, 0$ från den optimala radlängden 30 tecken. Observera för övrigt att en rad aldrig inleds eller avslutas med mellanslag och att det ska finnas exakt ett mellanslag mellan två ord.

UPPGIFT 6 – RIMORD

Du får givet en lång rad med n bokstäver, för enkelhets skull valda bland $A \ldots Z$. Genom att börja och sluta läsa raden på valfria ställen, kan du plocka ut ord, totalt $n(n+1)/2$ stycken. Vi begränsar oss alltså inte till verkliga svenska ord utan accepterar vilken delsträng som helst av den givna raden som ett ord. Dock är vi endast intresserade av ord med ett bestämt antal, v, vokaler. Som vokaler räknar vi A, E, I, O, U, Y.

Din uppgift är att ta fram den största gruppen av sådana ord som alla rimmar på varandra. För att undvika missuppfattningar ger vi här en enkel definition på rim som förhoppningsvis stämmer någorlunda överens med ditt sätt att skriva vers. Två ord rimmar om de uppfyller följande tre villkor:

- Båda orden ska innehålla minst en vokal.
- Från och med första vokalen i varje ord och ända till slutet ska orden vara identiska. Det är alltså bara fram till första vokalen som orden får skilja sig åt.
- Orden får inte vara helt identiska.

Exempelvis rimmar `SJUNGA` och `UNGA`, `VARG` och `KARG`, `I` och `BLI` men inte till exempel `VEDSTAPEL` och `KONSTAPEL`, `DRYG` och `ODRYG`, `TA` och `TA`.

Notera att om ord_1 rimmar på ord_2 och ord_2 rimmar på ord_3, så rimmar automatiskt ord_1 på ord_3.

Indata: Läses från filen `rim.in`. På första raden står ett heltal $n, 1 \leq n \leq 10000$ som anger hur många bokstäver den långa raden av bokstäver innehåller. På andra raden ett heltal $v, 1 \leq v \leq 10$, som anger hur många vokaler de bildade orden ska innehålla. På tredje och sista raden står en obruten följd av n bokstäver $A \ldots Z$. Antalet vokaler bland dem är alltid större än v.

Utdata: Ett heltal som anger det maximala antalet ord som

- går att bilda från bokstavsföljden genom att börja och sluta läsa den på valfria ställen och
- innehåller exakt v vokaler och
- rimmar på varandra.

Orden får överlappa varandra i den ursprungliga följden och de får börja eller sluta på samma position i den ursprungliga följden, men observera att samma ord inte får räknas flera gånger.

Ett körningsexempel. Då filen `rim.in` innehåller

```
26
2
KVASTASTAHASTARMEDATTBASTA
```

ska ge utskriften

```
Den största gruppen innehåller 9 ord
```

Förklaring: Den största gruppen tvåvokaliga rimord är: `ASTA`, `BASTA`, `TBASTA`,`TTBASTA`, `HASTA`, `TASTA`, `STASTA`, `VASTA` och `KVASTA`

UPPGIFT 7 – KULTURKROCKAR

Historiker är ofta intresserade av i vilken kultur och vid vilken tidpunkt en viss uppfinning först uppstod. Detta är dock inte så lätt att veta eftersom olika kulturer då och då mötts och utbytt information. Om man vet vilka sådana utbyten som har ägt rum och vilka kulturer som känner till uppfinningen idag, kan man ibland räkna ut var den har uppstått. Du ska skriva ett program som gör just detta.

Vi ska använda en mycket förenklad modell av kulturers möten. Vi tänker oss n kulturer (numrerade från 1 till n) som lever helt isolerade utom vid vissa tidpunkter då det sker ett fullständigt utbyte av information mellan två kulturer. Vi behöver inte bry oss om huruvida det rör sig om fredligt samarbete eller krigserövringar, det enda viktiga är att om uppfinningen var känd i den ena kulturen före mötet, så är den känd i båda kulturerna efter mötet. Notera att antalet kulturer aldrig ändras. Efter mötet fortsätter de två kulturerna att vara isolerade och kan samverka på olika sätt med andra kulturer. Det enda vi vet säkert, är att omedelbart efter mötet bär dessa två kulturer på exakt samma kunskap.

Indata: Läses från filen `kult.in`. På första raden i filen står två heltal n och k, där $2 \leq n \leq 1000$ och $1 \leq k \leq 10000$. n är antalet kulturer i vår modell och k är det totala antalet möten vi betraktar. Därefter kommer k rader med vardera två heltal a och b, båda i intervallet $1 \ldots n$. Varje rad beskriver ett möte mellan kultur nummer a och kultur nummer b. Mötena inträffar i precis den ordning som anges i filen. Det exakta årtalet för varje möte spelar ingen roll. Om det underlättar kan du anta att mötet beskrivet på den femte raden inträffade vid tid 5, den sjätte raden vid tid 6 och så vidare. Varje möte antas vara fullkomligt avslutat innan nästa möte äger rum.

Efter mötesbeskrivningarna följer en rad med ett heltal $m, 2 \leq m \leq n$, antalet kulturer som idag känner till uppfinningen, och på följande rad m heltal i intervallet $1 \ldots n$, som anger numren på de kulturer som idag känner till uppfinningen.

Utdata: Programmet ska skriva en lista med heltal i intervallet $1 \ldots n$. Varje tal anger numret på en kultur i vilken uppfinningen kan ha uppstått. Eller mer precist uttryckt: Om det för kultur x finns en möjlig tidpunkt för uppfinningen som gör att spridningen av uppfinningen idag är exakt så som anges i indatafilen, då ska x finnas med i listan, annars inte. Alla talen i listan ska vara olika men ordningen spelar ingen roll. Ett körningsexempel. Då filen `kul.in` innehåller

```
6 7
5 6
5 4
2 4
3 5
6 2
1 3
4 5
4
1 3 4 5
```

ska resultatet bli:

```
Möjliga ursprungskulturer: 3 5
```

Notera att det alltid finns minst två möjliga ursprungskulturer, eftersom $m \geq 2$ vilket betyder att det måste ha skett minst ett utbyte av information om uppfinningen. Därmed kan det inte vara entydigt bestämt vilken kultur som var först med uppfinningen.