Informatikaren oinarriak

C lengoaian ebatzitako problemak

Industria Ingeniaritza Teknikoko eta Industria Graduko ikasleei zuzendua

Olatz Ansa Osteriz
Rosa Arruabarrena Santos
M. Carmen Ocariz Sanz
Montserrat Ferreira

Escuela Universitaria Politécnica
Unibertsitate Eskola Politeknikoa

eman ta zabal zazu
Universidad del País Vasco
Euskal Herriko Unibertsitatea

www.ehu.es
ISBN: 978-84-9860-464-1

ISBN: 978-84-9860-464-1
L.G.: BI-3.231-2010

Bilbo, 2010eko abendua

www.argitalpenak.ehu.es

AURKIBIDEA

PROGRAMAZIOKO ARIKETA LABURRAK

Ondoren programazio ariketa sinpleak azaltzen zaizkizu C programazio lengoaian idatziak ala idatzi beharrekoa. Lehendabizi 20 enuntziatuen zerrendaketa azaltzen zaizu, saia zaitez burutzen ondoren agertzen diren soluzioak kontsultatu aurretik.

Proposatutako soluzioak Dev-C++ 4.9.9.2 bertsioan probatu dira. Aipatutako softwarea librea http://www.bloodshed.net/ helbidetik eskura liteke.

ENUNTZIATUAK

1. Idatzi zein den hurrengo programaren irteera

```
#include <stdio.h>
#include <stdlib.h>

main (){
 int s, j, k, i, l;
 s=0;
 for (j=1;j<=5;j++)
 { printf("%d", j );
   if (j%2 ==0) { s=s+j; }
 }
 printf("\n%d",s);
 i= 10;
 while (i>0)
   i=i-1;
 printf("\n%d",i);
 printf("\n\n");
 system("PAUSE");
}
```

2. Idatzi programa honen irteera

```
#include <stdio.h>
#include <stdlib.h>

main (){
 int i, j;
 for (i=1; i<=10; i++)
 {   for (j=1; j<=10-i+1; j++)
     {  printf(" %d",j); }
     printf("\n");
 }
 printf("\n\n");
 system("PAUSE");
}
```

3. Ondorengo programaren irteera idatzi.

```
#include <stdio.h>
#include <stdlib.h>

main (){
 int i, j, batu;
 for (i=1; i<=3; i++)
 {   batu=0;
     for (j=1; j<=i; j++)
     {  batu=batu+j;
        if (i!=j) {printf("%d/%d + ",i,batu);}
        else {printf("%d/%d \n",i,batu);}
     }
 }
 printf("\n\n");
 system("PAUSE");
}
```

4. Adierazi jarraian datorren programak zer idatziko lukeen pantailan:

```
#include <stdio.h>
#include <stdlib.h>

#define LerroKop 6

main ()
{ int lerroa, zutabea, zenbakia;
  zenbakia=0;
  for (lerroa=1;lerroa<=LerroKop; lerroa++)
  { zutabea=1;
    while (zutabea<=lerroa)
    { zenbakia ++;        // edo  zenbakia= zenbakia+1;
      printf("%5d", zenbakia);
      zutabea ++;
     }
     printf("\n");
  }
  printf("\n\n");
  system("PAUSE");
}
```

5. Esan, zer idatziko lukeen pantailan jarraian datorren programak:

```
#include <stdio.h>
#include <stdlib.h>

main ()
{ int Minimoa=1, Maximoa=5, i, datua=25;

  for(i=Maximoa;Minimoa<=i;i--)
  {  printf("%d garrena doana %d da.\n", i, datua-i); }
  printf("Egina dago");

  printf("\n\n");
  system("PAUSE");
}
```

6. Esan, jarraian datorren programak zer idatziko lukeen pantailan:

```
#include <stdio.h>
#include <stdlib.h>

main ()
{ int Minimoa=1, Maximoa=3, i,j;

  for(i=Minimoa;i<=Maximoa;i++)
    for(j=Minimoa;j<=Maximoa;j++)
      printf("%d balio du I-k eta J-k  %d balio du.\n", i, j);

  printf("\n\n");
  system("PAUSE");
}
```

7. Idatz ezazu jarraian datorren programak zer idatziko lukeen pantailan.

```
#include <stdio.h>
#include <stdlib.h>
```

```
main ()
{ int i,k;

  for (i=1; i<=5; i++)
  { for (k=1; k<=i; k++)
      printf("*");
    printf("\n");
  }

  for (i=4; 1<=i; i--)
  { for (k=1; k<=i; k++)
      printf("*");
    printf("\n");
  }

  printf("\n\n");
  system("PAUSE");
}
```

8. Esan, jarraian datorren programak zer idatziko lukeen pantailan.

```
#include <stdio.h>
#include <stdlib.h>

main (){
 int a,b,n,x,y;
 n=6;
 x=0;
 y=0;
 for (a=1; a<=n; a++)
 { if (a%2== 0)
   { for (b=a; b<n; b++) {x=x+1;}
     printf("\n%d >>> %d", a,x);
   }
   else
   { for (b=1; b<a; b++) {y=y+1;}
     printf("\n%d >>> %d", a,y);
   }
 }
 printf("\n\n");
 system("PAUSE");
}
```

9. Ondorengo programaren irteera zein den idatzi:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

main (){
 char katea[]="etxeko teilatuan teilak daude?";
 int kont,eKop,oraingoz, noraino, ind, luz;
 kont=0;
 luz= strlen(katea);       // katea aldagaiaren luzera
 for (ind=0; ind<luz; ind++)
 { if (katea[ind]=='e') {kont++;}
 }
 printf("'%s' katean %d 'e' daude.\n",katea, kont);
 for (eKop=1;eKop<=kont;eKop++)
```

```
 { printf("%d 'e' ditu kateak: ",eKop);
   oraingoz=0;
   noraino=0;
   while (oraingoz<eKop)
   { if (katea[noraino]=='e') {oraingoz++;}
     noraino++;
   }
   for (ind=0; ind<noraino; ind++) {printf("%c", katea[ind]);}
   printf("\n");
 }
 printf("\n\n");
 system("PAUSE");
}
```

10. Jarraian dauden bi programa zati hauek baliokideak diren edo ez esan, zure erantzuna arrazoitu.

    (Programa zati bi baliokideak dira baldin eta bata besteaz ordezkatzerakoan programa osoak gauza bera egiten badu.)

A.-

`if (x<10) {x:=x+1;}` `if (x>=10) {x:=x-1;}`	`if (x<10) {x:=x+1;}` `else {x:=x-1;}`

B.-

`for (i=10;i<=25; i++)`
`  printf("  %d \n",i);`
`printf("Bukatu dut");`
	`i=10;`
	`while (i<=25)`
	`{ printf("  %d",i);`
	`  printf("Bukatu dut\n");`
	`}`

11. Efektu berdina lortzen da ondorengo programa biekin? Baiezkoa bada lortzen den efektua adierazi ezazu eta ezezkoa bada zergatik ez diren berdinak azal ezazu:

A	`dago= 0;`
	`ind = 0;`
	`while ((ind<N) && (dago==0))`
	`{ if (x==tau[ind]){dago=1;}`
	`  ind=ind+1;`
	`}`
	`if (dago==1) {printf("%d dago %d posizioan", x, ind-1);}`
	`else {printf("%d ez dago", x);}`

B	`dago=0;`
	`for (ind=0;ind<N;ind++)`
	`  if (x==tau[ind]) {dago=1;}`
	`if (dago==1) {printf("%d dago %d posizioan", x, ind-1);}`
	`else {printf("%d ez dago", x);}`

12. Efektu berdina lortzen da ondorengo programa biekin? Baiezkoa bada lortzen den efektua adierazi ezazu eta ezezkoa bada zergatik ez diren berdinak azal ezazu. (Suposatu N eta Kont aldagaiak osokoak direla).

A	`printf("Eman zenbaki bat: ");`
	`scanf("%d", &n);`
	`kont=1;`
	`while (kont<=n)`
	`{ printf("%d \n", kont);`

	kont=kont+2; }

B	printf("Eman zenbaki bat: ");
	scanf("%d", &n);
	for(kont=1;kont<=n;kont++)
	{ **if** (kont%2==0) {printf("%d \n", kont);}
	}

13. Jarraian datorren programak zer egiten duen esan; hots, ariketari dagokion enuntziatua idatzi.

```
#include <stdio.h>
#include <stdlib.h>
main ()
{ int x, y, osokoBat;
  printf("Datuak eman: ");
  scanf("%d",&osokoBat);
  x=0;
  y=0;
  while (osokoBat!=0)
  { if (osokoBat%2==0) {y++;}
    x++;
    scanf("%d",&osokoBat);
  }
  printf("\nX-aren balioa: %d", x);
  printf("\nY-aren balioa: %d", y);

  printf("\n\n");
  system("PAUSE");
}
```

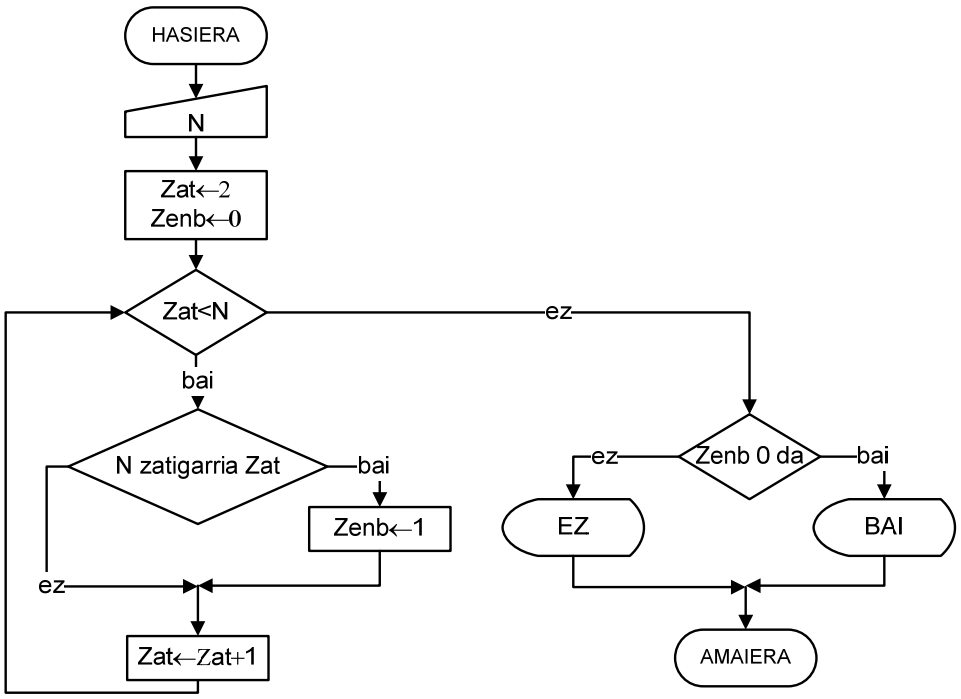
14. Ondorengo programa zatia zuzena den ala ez ikusi. Zuzena ez bada, zergatik azaldu eta erroreak zuzentzeko beharko liratekeen aldaketak azaldu.

```
#define <stdio.h>
#define <stdlib.h>

#include GoiMuga 10;

main (){
 int tau [GoiMuga], i, lag;

 for (i=1; i<=GoiMuga; i++)
 {   tau[i]=i;
     i=i+1;
 }
 i=1;
 while (i<=GoiMuga)
 { lag=tau[i];
   tau[i]=tau[i+1];
   tau[i+1]=lag;
 }

 printf("\n\n");
 system("PAUSE");
}
```

15. Jarraian datorren **IF** sententzia **switch** sententzi bakar bat bezala idatzi:

```
#include <stdio.h>
#include <stdlib.h>

main ()
{ int n;
  printf("Eman zenbaki osokoa: "); scanf("%d", &n);
  if ((n==5) || (n==6) ){n=n+4;}
  else if ((n>7) && (n<=10)) { n=n-5;}
  else if (n==7) {n=n+1;}
  else {n=n-1; }
  printf("Osoko berria: %d", n);

  printf("\n\n");
  system("PAUSE");
}
```

16. Jarraian datorren fluxu diagramak zer egiten duen esan; hots, ariketari dagokion enuntziatua idatzi. Honetaz gain, fluxu diagramak adierazten duen C kodea idatzi:



17. Ondorengoa egiten duen fluxu diagrama eta C programa idatzi: Erabiltzaileak teklatutik emandako bi balio osoko eta 50 baino txikiagoak irakurri behar ditu. Balioak 50 baino txikiagoak ez direnean, balio-pare berria eskatuko du. Behin balioak zuzenak ditugunean, balio txikiena bosnaka inkrementatuz joango da eta handiena, aldiz, binaka dekrementatuz, bi balioak gurutzatu arte. Gainera, hasierako bi balioetarako lortzen diren balio guztien segida idatziko du pantailan.

18. Puntu karaktereaz amaitzen den karaktere-sekuentzia batean 'a' karakterea 't' karaktere baten atzetik zenbat aldiz agertzen den kontatzen duen programa egin nahi izan dugu, baina ez dugu lortu. Badakigu programan **5 errore daudela**, bila itzazu eta zuzendu. (Aldaketak programaren gainean egin)

    Adibidea: letra lodiaz erabiltzaileak sakatzen duena

```
Idatzi esaldi bat (puntua bukatzeko): abba ttkktajjtaiitaktaoi.
4 aldiz agertzen da – ta –
```

```
#include <stdio.h>
#include <stdlib.h>
main ()
{ int lehen, taKop=0;
  char orain;
  lehen='z';
  printf("Eman esaldi bat '.'-z amaitzen dena: \n");
  scanf("%c", &orain);
  while (orain!='.')
    if (orain=='a' || lehen=='t') then {taKop ++;}
    lehen=orain;
    scanf("%c", &orain);

  printf("%taKop aldiz dago '-ta-' esaldian.", taKop);

  printf("\n\n");
  system("PAUSE");
}
```

19. Egizu programa bat erabiltzaileari N zenbaki osokoa eskatuko diona, eta 1etik hasi eta launaka gorantz N-rainoko osokoen karratuen batura kalkulatuko duena.

    Exekuzio adibidea: (erabiltzaileak sakatzen duena lodiz)

```
Eman osoko bat: 10
Soluzioa: 1² + 5² + 9²
```

20. Jarraian datorren **FOR** sententzia **WHILE** sententzia bezala idatzi.

```
#include <stdio.h>
#include <stdlib.h>
main ()
{ int x, n=0;
  for (x=45; 1<=x; x--)
    if (x<30) {n=n-x;}
    else {n=n+x;}
  printf("%d", n);
  printf("\n\n");
  system("PAUSE");
}
```

SOLUZIOEN PROPOSAMENAK

1. 12345
   6
   0

2. 1 2 3 4 5 6 7 8 9 10
   1 2 3 4 5 6 7 8 9
   1 2 3 4 5 6 7 8
   1 2 3 4 5 6 7
   1 2 3 4 5 6
   1 2 3 4 5
   1 2 3 4
   1 2 3
   1 2
   1

3. 1/1
   2/1 + 2/3
   3/1 + 3/3 + 3/6

4.
```
    1
    2   3
    4   5   6
    7   8   9  10
   11  12  13  14  15
   16  17  18  19  20  21
```

5. 5 garrena doana 20 da.
   4 garrena doana 21 da.
   3 garrena doana 22 da.
   2 garrena doana 23 da.
   1 garrena doana 24 da.
   Egina dago

6. 1 balio du I-k eta J-k 1 balio du.
   1 balio du I-k eta J-k 2 balio du.
   1 balio du I-k eta J-k 3 balio du.
   2 balio du I-k eta J-k 1 balio du.
   2 balio du I-k eta J-k 2 balio du.
   2 balio du I-k eta J-k 3 balio du.
   3 balio du I-k eta J-k 1 balio du.
   3 balio du I-k eta J-k 2 balio du.
   3 balio du I-k eta J-k 3 balio du.

7.
```
*
**
***
****
*****
****
***
**
*
```

8. 1 === 1
   2 >>> 5
   3 === 4
   4 >>> 8
   5 === 9
   6 >>> 9

9. ‘etxeko teilatuan teilak daude?’ katean 5 ‘e’ daude.
   1 ‘e’ ditu kateak: e
   2 ‘e’ ditu kateak: etxe
   3 ‘e’ ditu kateak: etxeko te
   4 ‘e’ ditu kateak: etxeko teilatuan te
   5 ‘e’ ditu kateak: etxeko teilatuan teilak daude

10. A) Ez dira baliokideak. x 9 denean, lehenengo zatiko bi aginduak exekutatu ostean, x aldagaiak 9 balioa du berriz ere. Beste zatian berriz, **if** aginduko baldintza betetzen denez, x aldagaiak 10 balioa lortuko du agindu bakarraren exekuzioaren ondorioz.

    B) Ez dira baliokideak. Lehenengo zatiak 17 lerro idazten ditu: lehenengoan 10 zenbakia, bigarrengoan 11, …, azken aurrekoan 25, azkenekoan ‘Bukatu dut’. Bigarren zatiak, aldiz, ziklatu egiten du ‘10Bukatu dut’ testua behin eta berriz idatziz.

11. Ez, dute efektu bera ekoizten x balioa taulan dagoenean: A zatiak X taulako zein posizioan metatuta dagoen adierazten du (posizioen zenbakitzea 0tik hasten delarik); B zatiak aldiz, X taulan dagoenean beti taulako osagai kopurua ken 1 itzultzen du.

12. Bi zatiek ez dute efektu bera ekoizten. A zatiak 1 N tarteko zenbaki bakoitiak idazten ditu eta B zatiak, aldiz, 1 N tarteko bikoitiak. Kodeek efektu bera ekoizteko aski da B zatiko **if** aginduko baldintza ukatzea (hau da, kont%2!=0 edo kont%2==1 baldintzetako bat ezartzea dagoena ordezkatuz). Aipatutako aldaketaren ondorioz, N izanik sakatutako osokoa, 1 N tarteko zenbaki bakoitiak idatziko dituzte, lerro bakoitzean zenbaki bakar bat erakutsiz.

13. X aldagaiak (0-a izan ezik) sakatutako osokoen kopurua jasotzen du, eta Y aldagaiak (0-a izan ezik) sakatutako bikoitien kopurua

14.
```
#define <stdio.h>                    ordezkatu  #include <stdio.h>
#define <stdlib.h>                   ord.  #include <stdio.h>

#include GoiMuga 10;                 ord.  #define GoiMuga 10

main (){
 int tau [GoiMuga], i, lag;

 for (i=1; i<=GoiMuga; i++)          ord.  for (i=0; i<GoiMuga; i++)
 {   tau[i]=i;
     i=i+1;                          ezabatu
 }
```

```
 i=1;                                ord.  i=0;
 while (i<=GoiMuga)                  ord.  while (i<GoiMuga)
 { lag=tau[i];
   tau[i]=tau[i+1];
   tau[i+1]=lag;                     gehitu  i=i+1;
 }

 printf("\n\n");
 system("PAUSE");
}
```

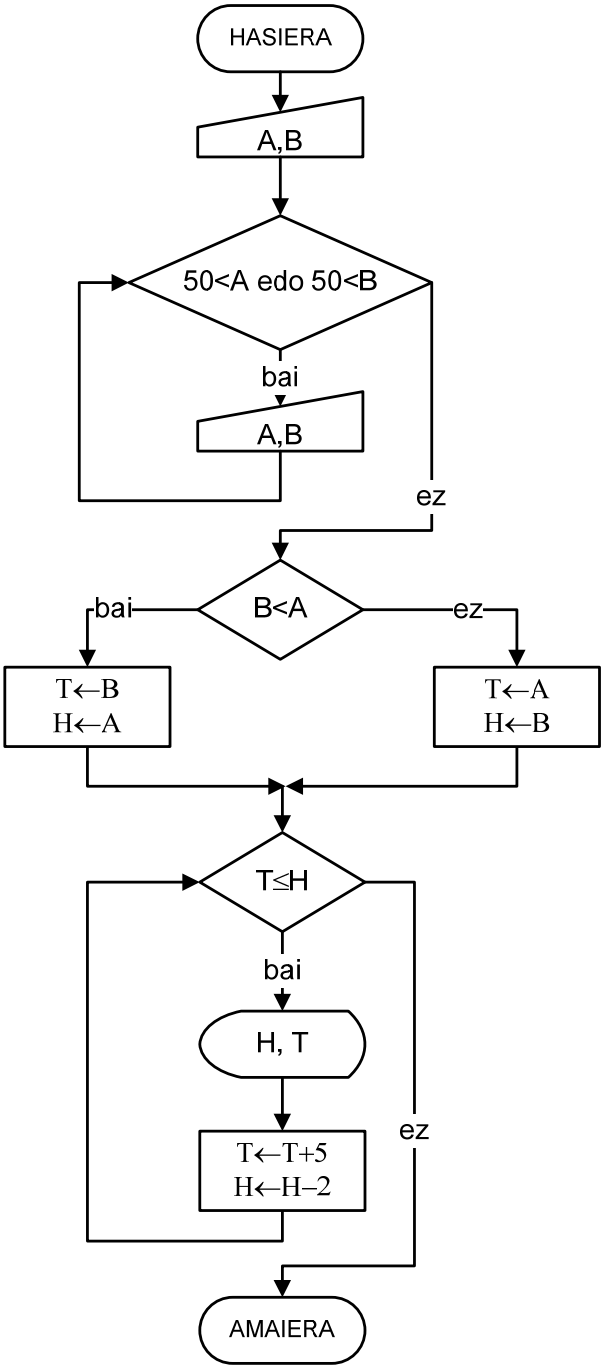
15.
```
 switch (n) {
 case 5: case 6: n=n+4; break;
 case 7: n=n+1; break;
 case 8: case 9: case 10: n=n-5; break;
 otherwise: n=n-1;
 }
```

16. N zenbakia lehena den edo ez esaten duen fluxu diagrama eta programa idatzi. N zenbakia lehena bada programak ‘Bai’ idatziko du eta bestela ‘Ez’ idatziko du.

```
#include <stdio.h>
#include <stdlib.h>

main ()
{ int n, zat,zenb;
  printf("Eman zenbaki bat: ");
  scanf("%d", &n);
  zat=2;
  zenb=0;
  while (zat<n)
  { if (n%zat==0) {zenb=1;}
    zat=zat+1;
  }
  if (zenb==0) {printf("BAI");}
  else {printf("EZ");}

  printf("\n\n");
  system("PAUSE");
}
```

17.
```
#include <stdio.h>
#include <stdlib.h>

main ()
{ int a,b,t,h;
  printf("Eman 50 baino txikiagoak diren bi osoko "
         "(zuri-unez banaturik): ");
  scanf("%d %d", &a, &b);
  while (50<a || 50<b)
  { printf("Eman 50 baino txikiagoak diren bi osoko "
           "(zuri-unez banaturik): ");
    scanf("%d %d", &a, &b);
  }
  if (b<a) {t=b; h=a;} else {t=a; h=b;}
  while (t<=h)
```

```
  { printf("Txkiena: %5d    Handiena: %5d\n",t,h);
    t=t+5;
    h=h-2;
  }

  printf("\n\n");
  system("PAUSE");
}
```



18.
```
#include <stdio.h>
#include <stdlib.h>

main ()
```

```
    { int taKop=0;
      char orain, lehen;
      lehen='z';
      printf("Eman esaldi bat '.'-z amaitzen dena: \n");
      scanf("%c", &orain);
      while (orain!='.')
      { if (orain=='a' && lehen=='t') {taKop ++;}
        lehen=orain;
        scanf("%c", &orain);
      }
      printf("%d aldiz dago '-ta-' esaldian.", taKop);

      printf("\n\n");
      system("PAUSE");
    }
```

19.
```
#include <stdio.h>
#include <stdlib.h>
main ()
{ int n, z;
  long batu;
  printf("Eman osoko bat: ");
  scanf("%d",&n);
  batu=0;
  z=1;
  while (z<=n)
  { batu= batu + z*z;
    z=z+4;
  }
  printf("Soluzioa: %ld", batu);
  printf("\n\n");
  system("PAUSE");
}
```

20.
```
#include <stdio.h>
#include <stdlib.h>
main ()
{ int x, n=0;
  x=45;
  while (1<=x)
  { if (x<30) {n=n-x;}
    else {n=n+x;}
    x--;
  }
  printf("%d", n);
  printf("\n\n");
  system("PAUSE");
}
```

ARIKETA LUZEAK

Ondoren azterketa moduko programazio ariketa luzeak azaltzen dira, haien enuntziatua, analisia eta programa C lengoaian idatzita daudela. Saia zaitez ariketa hauek burutzen agertzen diren soluzioak kontsultatu aurretik.

Proposatutako soluzioak Dev-C++ 4.9.9.2 bertsioan probatu dira. Aipatutako softwarea librea http://www.bloodshed.net/ helbidetik eskura liteke.

GAILU ELEKTRONIKOAK

Gailu elektronikoak muntatzen dituen enpresa batean, osagaien eskaerak prozesatzen dituen programa bat egin nahi da. Enpresak 10 gailu elektroniko mota muntatzen ditu. Gailu bakoitza osatzeko osagai elektroniko batzuk behar dira, mota bakoitzean osagai kopurua desberdina delarik. Osagai bakoitzeko gailu batean gehienez unitate bat egon daiteke. Guztira, enpresak 100 osagai elektroniko desberdin erabiltzen ditu.

Ondoren aipatzen diren funtzioak egiten dituen programa bat egin behar duzu:

1. Gailu bakoitzaren osagaien datuak irakurri behar dira. Erabiltzaileak sartu behar ditu 10 aparatu-moten osagai-zerrenda. Gailu bakoitzeko, lehenbizi osagai-kopurua irakurriko da.
2. Hilean enpresak produzitzen dituen gailuen kopuruak irakurri behar ditu gailu-mota bakoitzeko.
3. Hileko eskaerak egiteko, programak osagai bakoitzeko behar den kantitatea zenbatu behar du eta pantailatik idatzi.
4. Bukatzeko, idatzi behar da zein den gehien erabiltzen den osagai elektronikoa.

Lehendabizi erabili beharreko datu-motetan pentsatu (taulak) eta ez ahaztu fluxu-diagramak eta/edo algoritmoek lagunduko zaituztela programaren eraikuntzan. Programaren zati bakoitza ondo identifikatu iruzkinen bidez.

Adibidea: (letra lodiz dauden datuak erabiltzaileak sartutakoak dira)

```
Gailu bakoitzaren osagaiak sartu:
0. gailuaren osagai-kopurua idatzi: 3
    1. Osagaia: 1
    2. Osagaia: 3
    3. Osagaia: 99
1. gailuaren osagai-kopurua idatzi: 20
    1. Osagaia: 4
    2. Osagaia: 6
    ...
    20. Osagaia: 85
...
9. gailuaren osagai-kopurua idatzi: 1
    1. Osagaia: 65

Sartu hilean muntatutako gailu elektroniko kopurua:
0. gailua: 100
1. gailua: 1000
...
9. gailua: 300
```

```
Hilean, erabilitako osagai elektronikoen kopuruak:

0. osagaia: 100
...
99. osagaia: 100

Gehien erabiltzen den osagai elektronikoa 7. da, 10000 unitate
behar direlako.
```

PROBLEMAREN ANALISIA

Enuntziatuko konstanteak

1. *GailuKop* = 10. Enpresak muntatzen dituen gailu mota kopurua
2. *OsagaiKop* = 100. Enpresak erabiltzen dituen osagai elektroniko kopurua

Sarrerarako beharrezkoak diren aldagaien adierazpidea

3. *GailuKop* x *Osagaikop* posizio dituen matrizea behar dugu gailu bakoitzak behar dituen osagai elektronikoak gordetzeko. Bertan batekoa duten posizioek gailu horrek osagai hori erabiltzen duela adieraziko du eta bestela zero gordeko da.

osagaiak	*0*	*1*	*...*	*OsagaiKop-1*
0				
1				
...				
GailuKop-1				

4. *GailuKop* posizio dituen bektorea behar dugu. Bertan, gailu bakoitzetik hilabetean sortzen diren unitate kopurua gordeko da.

	0	*1*	*...*	*GailuKop-1*
hilGailuak				

Emaitzetarako beharrezkoak diren aldagaien adierazpidea

5. *OsagaiKop* posizio dituen bektorea behar dugu. Bertan, osagai bakoitzetik hil osoko gailuak muntatzeko behar diren unitate kopurua gordeko da.

	0	*1*	*...*	*OsagaiKop-1*
hilOsagaiak				

Problema ebazteko jarraibideak

1. *osagaiak* matrizea 0ra hasieratu; honek adieraziko du gailuek ez dutela inongo osagairik lotuta.
2. Gailu bakoitzarentzat:
   a. Irakurri gailu horren osagai kopurua (*gailuOsagKop*)
   b. *gailuOsagKop* osagaien kodeak irakurri eta *osagaiak* matrizean gailu horri dagokion lerroan eta irakurritako osagaiari dagokion zutabean 1 jarri.
3. Enpresak hilabetean gailu bakoitzetik egiten dituen unitateak irakurri eta *hilGailuak* bektorean gorde.
4. Osagai bakoitzerako (*osagai*):
   a. Osagai kontadorea 0ra jarri

b. Gailu guztiak (*gailu*) begiratu eta gailu horrek osagai hori erabiltzen badu (*osagaiak[gailu][osagai]==1*) orduan *hilOsagaiak* bektorean osagai horri dagokion posizioan gailu horretatik egiten diren unitateak gehitu.

5. Osagairik erabiliena kalkulatzeko nahikoa da *hilOsagaiak* bektoreko balio maximoaren posizioa zein den aurkitzea. Hau lortzeko eman beharreko pausoak:

   a. Suposatuko dugu *hilOsagaiak* bektoreko lehen posizioan (*gehien*) dagoen balioa maximoa dela.

   b. Gainontzeko osagaietarako egin:

      i. Konparatu osagai horren erabilpena *gehien* osagaiaren erabilpenarekin.

      ii. Handiagoa bada, osagai berri hori izango da orain arteko osagairik erabiliena eta *gehien* aldagaian gordeko dugu.

   c. Azkenik, pantailan *gehien* aldagaiak duen balioa idatziko da, osagairik erabiliena zein den adieraziz.

C KODEKETA

```
#include <stdlib.h>
#include <stdio.h>
#define GailuKop 10
#define OsagaiKop 100
main(){
int osagaiak[GailuKop][OsagaiKop];
int hilGailuak[GailuKop];
int hilOsagaiak[OsagaiKop];
int gailu,osagai,gailuOsa,gailuOsaKop,gehien;

/* osagaiak matrizea 0ra hasieratu. Osagairik ez gailuko*/
for (gailu=0; gailu<GailuKop; gailu=gailu+1){
  for (osagai=0; osagai<OsagaiKop; osagai=osagai+1){
    osagaiak[gailu][osagai]=0;
  }
}
/* Gailu-mota bakoitzaren osagaiak irakurri eta matrizean gorde.
Gailu batek osagai bat baldin badauka matrizean posizio horretan
bateko bat egongo da */
printf("Gailu bakoitzaren osagaiak sartu:\n");
for (gailu=0; gailu<GailuKop; gailu=gailu+1){
  printf("%d. gailuaren osagai-kopurua idatzi: ", gailu);
  scanf("%d", &gailuOsaKop);
  for (gailuOsa=0; gailuOsa<gailuOsaKop; gailuOsa=gailuOsa+1){
    printf("%d. Osagaia: ", gailuOsa);
    scanf("%d", &osagai);
    osagaiak[gailu][osagai]=1;
  }
}
/* Hilean enpresak produzitzen dituen gailuak irakurri */
printf("Sartu hilean muntatutako gailu elektroniko kopurua: \n");
for (gailu=0; gailu<GailuKop; gailu=gailu+1){
  printf("%d. gailua: ", gailu);
  scanf("%d", &hilGailuak[gailu]);
```

```
}
/* Hilean enpresak behar dituen osagai-kopuruak kalkulatu, hau egiteko gailu bakoitzak zer osagai ditu eta zenbat unitate egiten diren kontutan hartu behar da */
printf("Hilean, erabilitako osagai elektronikoen kopuruak : \n");
for (osagai=0; osagai<OsagaiKop; osagai=osagai+1){
  hilOsagaiak[osagai]=0;
  for (gailu=0; gailu<GailuKop; gailu=gailu+1){
    if (osagaiak[gailu][osagai]==1) {
     hilOsagaiak[osagai]=hilOsagaiak[osagai]+hilGailuak[gailu];
    }
  }
  printf("%d. osagaia: %d\n", osagai, hilOsagaiak[osagai]);
}
/* Gehien erabiltzen den osagai elektronikoa */
gehien=0;
for (osagai=1; osagai<OsagaiKop; osagai=osagai+1){
  if (hilOsagaiak[gehien]<hilOsagaiak[osagai]) {
    gehien=osagai;
  }
}
printf("Gehien erabiltzen den osagai elektronikoa %d. da %d unitate behar direlako.\n", gehien, hilOsagaiak[gehien]);

printf("\n\n");
system("PAUSE");
}
```

KONTSUMO OHITURAK: EDARIAK

Eusko Jaurlaritzak gazteriaren edari eta alkohol kontsumo ohituren azterketa bat egin nahi du. Hori dela eta, galdeketa bat egiteko 100 herri aukeratu ditu. Herri bakoitzeko udaletxean aurreko urtean herriko tabernetan 8 edari motetan kontsumitutako litro kopuruak jasoko dira. 8 edari motak honakoak dira:

Edari motak:

0. Naturalak: ura, esnea, zukuak, mostoak, infusioak, ...
1. Gaseosoak: freskagarriak,...
2. Ardoak
3. Garagardoak
4. Whisky-ak
5. Likoreak
6. Energetikoak: Aquarius, Red Bull,...
7. Konbinatuak: kubatak, gin-tonikak,...

C programa baten bidez azterketa egin dezazula eskatzen zaizu, non azterketak ondorengo pausoez osaturik egon beharko duen:

1. Jaso eta metatu herri bakoitzeko: aipatutako edari mota bakoitzeko kontsumitutako litro kopurua. Gogoratu azterketarako 100 herriek informazioa emango dutela. Halere, informazioa ez da zertan ordenaturik emango, edari motarekiko edo litro kopuruarekiko; eta edari motaren batekiko informaziorik ez jasotzea gerta dakizuke.

   Honela, erabiltzaileak herri bakoitzerako (Edari Mota, Litroak) bikoteak sakatu beharko ditu eta herri horren datu gehiagorik ez duela adierazteko edari mota bezala -1a sakatzearekin aski izango da. Enuntziatuaren amaieran gehitutako adibideari begiratu atal bakoitzerako.

2. Herri guztien artean gehien edan den "edari mota" zein den kalkulatu; hau da, herri guztien artean litro kontsumitu kopuru handien duen edari mota.

3. Jaso eta metatu zein edari mota klasek duen alkohola. Horretarako, alkohola duten edari motak soilik zenbakitu ditzala erabiltzaileari eskatu, alkoholdun edari mota gehiagorik ez dagoela adierazteko -1 sakatu beharko duelarik.

Adibidea: (erabiltzaileak sakatutako datua letra lodiz)

Eman alkoholdun edarien zerrenda (amaitzeko -1 sakatu):

7 2 5 4 3 -1

Ondoren, ondoko galderei erantzun

a. AlkoholDUN edari moten artetik, zein da edanena?

b. Guztira, zein herrik edan du alkohola gehien?

Adibidea: (Letra lodiz erabiltzaileak sakatutakoa agertzen da)

```
Ikasketa egiteko herrika jasoko dugu informazioa:
Herri bakoitzeko: Bikoteak sartu (EdariMota, litroak).
Amaitzeko, azkeneko bikoteko lehenengo osagaiak -1 izan behar
du.
0 herria:
    0  8
    4  7
    2  0
    3  50
    1  10
    -1  0
1 herria:
    0  18
    1  15
    4  0
    2  6
    3  36
    -1  32
2 herria:
    4  16
    ...
99 herria:
    2  5
    0  12
    -1  9
Edari mota edanena da: 3
 {Litrotan gehien edan den edari mota garagardoa dela suposatzen
                                                        badugu}
Orain, bakarrik alkohola duten edari motak zenbaki ditzazula
behar dugu, -1 amaitzeko:   7    2    5    4    3    -1
AlkoholDUN edari mota edanena da: 4
        {“konbinatu”, “ardo”, “likore”, “whisky” eta “garagardo”
      alkoholdun edari mota artean litrotan hori dela suposatzen
                                                        badugu.}
      {OHARRA: Oro har, “edari mota” eta “alkoholdun edari mota”
                      edanenak ez dute zertan bat etorri behar}
Alkohola gehien kontsumitzen duen herria da: 21
        {“konbinatu”, “ardo”, “likore”, “whisky” eta “garagardo”
  motetako litroak batuz, 21 herria da litro gehien saldu duena}
```

PROBLEMAREN ANALISIA

Enuntziatuko konstanteak

1. *HerriKop* = 100. Azterketa egingo den herri kopurua.
2. *EdariKop* = 8. Aztertuko den edari kopurua.

Sarrerarako beharrezkoak diren aldagaien adierazpidea

3. *azterktea* izeneko eta *HerriKop x EdariKop* posizio dituen matrizea behar dugu; bertan herri bakoitzak edari bakoitzetik kontsumitutako litroak gordeko ditugu.

azterketa	0	1	...	EdariKop-1
0				
1				
...				
HerriKop-1				

4. *alkoholduna* izeneko eta *EdariKop* posizio dituen bektorea behar dugu; bertan, edari bakoitza alkoholduna den (1 balioaz adierazia) edo ez (0 balioaz adierazia) gordeko da.

	0	1	...	EdariKop-1
alkoholduna				

Emaitzetarako beharrezkoak diren aldagaien adierazpidea

5. *edarika* izeneko eta *EdariKop* posizio dituen bektorea behar dugu; bertan, edari mota bakoitzetik herri guztien artean kontsumitutako litro kopurua gordeko dira.

	0	1	...	EdariKop-1
edarika				

6. *herrika* izeneko eta *HerriKop* posizio dituen bektorea behar dugu; bertan, herri bakoitzerako alkoholdun edarietatik kontsumitutako litro kopurua gordeko da.

	0	1	...	HerriKop-1
herrika				

Problema ebazteko jarraibideak

1. *azterketa* matrizea 0ra hasieratu; herri bakoitzak edari bakoitzetik ez duela litrorik kontsumitu (oraingoz) adieraziz.
2. Herri bakoitzerako kontsumitutako edaria eta litroak adierazten duten bikoteak irakurri. Herri batentzat, edari betentzako bikote bat baino gehiago baldin badator, edari horrentzat emandako azken bikotearen litroak izango dira kontsumitutakoak (hau da, ez dira emandako litroak metatu behar).

   Horretarako, herri bakoitzerako (*h*):

a. Lehen edari kontsumo bikotea irakurri (*eda, lit*)

b. Irakurritako edari mota (*eda*) -1ren desberdina den bitartean egin:

   i. *azterketa* matrizean herriari (*h*) dagokion edarian (*eda*) litroak gorde.

   ii. Hurrengo edari kontsumo bikotea irakurri (*eda, lit*)

3. Edari bakoitzeko kalkulatu herri guztien artean kontsumitutako litroak, ondoren gehien kontsumitu den edaria zein den kalkulatu ahal izateko.

   Edari bakoitzerako (*e*):

   a. *edarika* bektorea 0ra hasieratu, oraingoz edari horretatik litrorik ez dela kontsumitu adieraziz.

   b. Herri guztietarako begiratu edari horretatik zenbat kontsumitu den eta e*darika* matrizean gehitu.

   Behin edari bakoitzetik kontsumitutako litroak kalkulatuta eta e*darika* bektorean gordeta, bektoreko balio maximoaren posizioa bilatu behar da, kontsumo handiena izan duen edaria lortzeko. Maximo hau kalkulatzeko prozesua honako hau da:

   c. Lehenik 0 edaritik kontsumitu dela litro gehien suposatuko dugu (e*danena*).

   d. Gainontzeko edarientzat (*e*) konprobatu behar da, *e* edaritik kontsumitutako litroak e*danena* edaritik kontsumitutako litroak baino handiagoa den (e*darika[e]*>*edarika[edanena]*). Horrela bada, *e* edaria bihurtuko da momentuko edaririk kontsumituena, e*danena*.

   e. Pantailan inprimatu zein den edaririk kontsumituena (e*danena*).

4. Alkoholdun edariak irakurri, horretarako eman beharreko pausoak:

   a. *alkoholduna* bektorea 0ra hasieratu, honen bidez edari guztiak alkohol gabeak direla kontsideratuko dugu.

   b. Edari bat irakurri (*eda*).

   c. Emandako edaria -1ren desberdina den bitartean:

      i. *alkoholduna* bektoreko *eda* posizioan 1 jarri, edari hori alkoholduna dela adieraziz.

      ii. Hurrengo edaria irakurri (*eda*).

5. Edari alkoholdunen artean edanena zein den kalkulatu. Honetarako bi bektoreen informazioaz baliatuko gara, *alkoholduna* eta e*darika*.

   a. Kontsumitutako alkohol litro maximoak gordeko dituen aldagaia 0ra hasieratu (*maxAlkohola*).

b. Ondoren edari bakoitzerako (*e*), edari alkoholduna bada (*alkoholduna[e]==1*) eta edari horretatik kontsumitutako litro kopurua (*edarika[e]*) orain arte kontsumitutako alkohol litro maximoa (*maxAlkohola*) baino handiago bada, *e* edaria izango da orain arte edari alkoholdun kontsumituena (*posMaxAlkohola=e*) eta gehien kontsumitutako litroak *e* edariarenak izango dira (*maxAlkohola=edarika[e]*).

c. Pantailan inprimatu alkoholdun edaririk kontsumituena (*posMaxAlkohola*)

6. Alkohol gehien kontsumitu duen herria kalkulatu. Lehenik, herri bakoitzarentzat kontsumitutako alkoholdun edarien litroak kalkulatu behar dira. Horretarako;

   Herri bakoitzerako (h):

   a. 0ra hasieratu kontsumitutako litroak *herrika*.

   b. Edari guztietarako (*e*), edari alkoholduna bada (*alkoholduna[e]==1*) orduan gehitu edari horretatik kontsumitutako litroak (*herrika[h]=herrika[h]+azterketa[h][e]*)

   Behin herri bakoitzak alkoholdun edarietatik kontsumitutako litroak kalkulatu ditugula, zein herrik kontsumitu dituen litro gehien kalkulatu behar da. Horretarako, *herrika* bektoreko balio maximoaren posizioa kalkulatu (aurreko atalean egindakoaren parekoa egin beharko genuke) eta ondoren pantailan idatziko dugu.

C KODEKETA

```
#include <stdio.h>
#include <stdlib.h>
#define HerriKop 10
#define EdariKop 8
main(){
int azterketa[HerriKop][EdariKop];
int edarika[EdariKop], alkoholduna[EdariKop];
int herrika[HerriKop];
int e, h, edanena, eda, lit, herriMax;
int maxAlkohola, posMaxAlkohola;

/*Azterketa matrizea 0ra hasieratu herri guztietako edari guztien
informazioa ez jasotzea gerta baliteke*/
for (h=0; h<HerriKop; h=h+1){
  for (e=0; e<EdariKop; e=e+1){
        azterketa[h][e]=0;
  }
}
/*Datuen jasotzea: Herrika, (edari mota, litroak)*/
printf("Ikasketa egiteko herrika jasoko dugu informazioa:\n");
printf("Herri bakoitzeko: Bikoteak sartu (EdariMota, "
       "litroak).\n");
printf("Amaitzeko, azkeneko bikoteko lehenengo osagaiak -1 izan"
       " behar du\n");
for (h=0; h<HerriKop; h=h+1){
  printf("%d herria: \n", h);
  scanf("%d %d", &eda,&lit);
  while (eda!= -1) {
        azterketa[h][e]=lit;
        scanf("%d %d", &eda, &lit);
  }
}
/*Edari edanena, herri guztietan edandako litroak kontuan edukiz*/
for (e=0; e<EdariKop; e=e+1){
     edarika[e]=0;
```

```
     for (h=0; h<HerriKop; h=h+1){
        edarika[e] = edarika[e] + azterketa[h][e];
     }
}
/*Edari kontsumituena kalkulatu*/
edanena=0;
for (e=1; e<EdariKop; e=e+1){
  if (edarika[edanena]<edarika[e]){
     edanena= e;
  }
}
printf("Edari mota edanena da: %d \n", edanena);


/*Alkoholdun edarien jasotzea. Denak 0-ra hasiera sakatzen ez bada
alkoholik ez duela adierazteko. Sakatzen bada edari mota bat,
alkohola duenez taulan 1 balioa jasoko dugu*/
for (e=0; e<EdariKop; e=e+1){
  alkoholduna[e]=0;
}


printf("Orain, bakarrik alkohola duten edari motak zenbaki ");
printf("ditzazula behar dugu, -1 amaitzeko: \n");
scanf("%d", &eda);
while (eda!=-1){
  alkoholduna[eda]=1;
  scanf("%d", &eda);
}


maxAlkohola=0;
for (e=1; e<EdariKop; e=e+1){
  if ((maxAlkohola < edarika[e]) && (alkoholduna[e]==1)){
      posMaxAlkohola = e;
      maxAlkohola = edarika[e];
  }
}
printf("AlkoholDUN edari mota edanena da: %d \n",posMaxAlkohola);
```

```
/*herri bakoitzak kontsumitutako alkohol litroak kalkulatu*/
for (h=0; h<HerriKop; h=h+1){
  herrika[h]=0;
  for (e=0; e<EdariKop; e=e+1){
    if (alkoholduna[e]==1){
        herrika[h] = herrika[h]+azterketa[h][e];
    }
  }
}
/*Alkohol gehien kontsumitu duen herria kalkulatu*/
herriMax=0;
for (h=1; h<HerriKop; h=h+1){
  if (herrika[herriMax]<herrika[h]){
      herriMax= h;
  }
}
printf("Alkohola gehien kontsumitzen duen herria da: %d\n",
herriMax);


printf("\n\n");
system("PAUSE");
}
```

IRRATSAIOA

Irratsaio batek lehiaketa bat prestatu nahi du: entzuleei 10 abestien izenburuak proposatuko dizkie, 0tik 9ra zenbakitu dituenak, eta parte hartu nahi duen entzuleak gustukoen dituen 3 izenburuak izendatu beharko ditu gustuarekiko ordena beherakorrean (berraz, gustukoena lehenengoz aipatu du).

Entzuleak aipatzen duen lehengo abestiari 3 puntu emango zaizkio, 2 bigarrenari eta puntu 1 hirugarrengo aipamenari.

Ondorengo ataza egingo dituen programa idaztea eskatzen da:

1. Entzule bakoitzak emititu dituen botoak irakurri eta metatu. Suposa ezazu gehienez 100 entzulek parte hartzen dutela, eta deitzen duten heinean jarraikiko zenbakiak esleitzen zaizkiela entzuleei. Datuen sarrera hirukoteen bidez egingo da, hirukote bakoitzak entzule baten botoak adieraziko duelarik. Datuen sarrera amaituko da lehen abesti bezala -1 ematen digutenean, berdin zaigularik zein beste bi balio datozen.

   ```
   Adibidea: (erabiltzailearen datuak letra lodiz)
             0 entzuleak:  6  2  1
             1 entzuleak:  3  7  5
             2 entzuleak:  3  0  1
             3 entzuleak:  0  5  2
             4 entzuleak:  -1  0  0
   ```

2. Abesti bakoitzak lortu dituen puntuak kalkulatu bai eta zeintzuk diren gehien eta bigarren gehien puntu lortu dituzten abesti biak (irabazlea eta 2.a).

   ```
   Adibidea:
             0 abestiak:    5 puntu
             1 abestiak:    2 puntu
             2 abestiak:    3 puntu
             3 abestiak:    6 puntu
             4 abestiak:    0 puntu
             5 abestiak:    3 puntu
             6 abestiak:    3 puntu
             7 abestiak:    2 puntu
             8 abestiak:    0 puntu
             9 abestiak:    0 puntu
             Abesti irabazlea: 3 abestia
             2.a: 0 abestia
   ```

3. Irratsaioaren amaieran parte hartu duten entzuleen artean puntuak banatzen dira ondorengo metodoa jarraituz: 30 puntu entzuleak bozkatu dituen 3 abestien artean abesti irabazlea badago, 20 puntu bigarrena badago eta 10 puntu gehiago bozkatutako hiru izenburuen artean irabazlea eta 2.a badaude.

Puntu gehien duen entzule bat mugatu nahi da opari bat emateko (entzule batek baino gehiago badute puntuazio maximoa, bat bakarrik eman behar du programak).

```
Adibidean:
          0 entzuleak:  0 puntu
          1 entzuleak: 30 puntu
          2 entzuleak: 60 puntu
          3 entzuleak: 20 puntu
          Irabazlea: 2 entzulea
```

PROBLEMAREN ANALISIA

Enuntziatuko konstanteak

1. *BotoKop* = 3. Parte-hartzaile bakoitzak eman ditzakeen boto kopurua.
2. *EntzuleKop* = 100. Lehiaketan parte hartu dezakeen entzule kopurua.
3. *AbestiKop* = 10. Lehian dauden abesti kopurua.

Sarrerarako beharrezkoak diren aldagaien adierazpidea

4. *botoak* izeneko eta *EntzuleKop* x *BotoKop* posizio dituen matrizea behar dugu. Bertan, entzule bakoitzak emaniko botoak zein abestiri eman dizkion gordeko dugu. Beraz, *botoak[i][j]=X* bada, *i* entzuleak *j*.botoa *X* abestiari eman diola adieraziko du.

botoak	0	1	...	BotoKop-1
0				
1				
...				
EntzuleKop-1				

Emaitzetarako beharrezkoak diren aldagaien adierazpidea

5. *abestiak* izeneko eta *AbestiKop* posizio dituen bektorea behar dugu, abesti bakoitzak entzule guztien artean izandako botoak gordetzeko.

	0	1	...	AbestiKop-1
abestiak				

Problema ebazteko jarraibideak

1. Entzuleen botoak irakurri:
   a. Lehiaketan parte-hartzen duten entzuleen kontadorea (*deiKopurua*) 0era hasieratu.
   b. Lehiakidearen hiru botoak irakurri (*b1, b2, b3*).
   c. Emandako hirukotea baliagarria den bitartean (lehen botoa -1ren desberdina) egin
      i. Entzuleak emandako botoak *botoak* matrizean gorde emandako ordena mantenduz.
      ii. Entzuleen kontadorea (*deiKopurua*) 1ean handitu.
      iii. Hurrengo lehiakideen botoak irakurri (*b1, b2, b3*).
2. Boto gehien jaso dituzten bi abestiak kalkulatu.
   a. Lehenik, abesti bakoitzak jaso dituen botoak kalkulatuko ditugu eta *abestiak* bektorean gordeko dugu. Horretarako emango ditugun pausoak:

i. Abesti bakoitzak dituen botoen kontadorea 0ra jarri; hau da, *abestiak* bektorea 0ra hasieratu.

ii. Lehiaketan parte hartu duen entzule bakoitzarentzat (*e*) emandako botoei dagozkien puntuak metatu *abestiak* bektorean, jakinik lehen botoari 3 puntu dagozkiola, bigarrenari 2 puntu eta hirugarrenari puntu 1.

iii. Pantailan inprimatu abesti bakoitzak izandako puntuazioa.

b. Ondoren, puntuazio altuena duten bi abestiak kalkulatuko dira:

i. Lehen bi abestiek izandako botoak konparatzen dira eta boto gehien izan dituena *lehena* izango da eta bestea *bigarrena*.

ii. Gainontzeko abesti bakoitzarentzat (*a*):

(1) Baldin *a* abestiak oraingoz puntu gehien dituen abestiak (*lehena*) baino puntu gehiago baditu orduan bi maximoak eguneratu ordena honetan: *bigarrena*k *lehena*k duen balioa jasoko du eta *lehena*k *a*k duen balioa.

(2) Bestela, *a* abestiak oraingoz bigarren abesti puntuatuena (*bigarrena*) baino puntu gehiago baditu, orduan *bigarrena*k *a*ren balioa hartuko du.

iii. Pantailan idatzi boto gehien izan dituzten bi abestiak (*lehena*, *bigarrena*).

3. Entzule irabazlea kalkulatu.

a. Orain arteko puntu maximoak 0ra hasieratu (*puntuMax*).

b. Lehiaketan parte hartu duen entzule bakoitzarentzat

i. Kalkulatu lortutako puntu kopurua, 0 puntu, 20 puntu, 30 puntu edo 60 puntu.

ii. Lortutako puntuazioa orain arteko maximoa gainditzen badu, orain arteko maximoa (*puntuMax*) eta entzule irabazlea (*txapelduna*) ere eguneratu.

c. Pantailan inprimatu puntu gehien lortu dituen entzulea (*txapelduna*).

C KODEKETA

```
#include <stdio.h>
#include <stdlib.h>
#define BotoKop 3
#define EntzuleKop 100
#define AbestiKop 10
main(){
int botoak[EntzuleKop][BotoKop];
int abestiak[AbestiKop];
int deiKopurua, lehena, bigarrena, lehenarenPuntuak;
int entzulea, txapelduna, b1, b2, b3, a, ind, puntuMax, ePuntuak;

/*Botoen irakurketa eta metaketa*/
deiKopurua=0;
printf("Sartu entzuleen hirukoteak. Lehendabizi gustukoena\n");
printf("Amaitzeko -1 sakatu lehen abesti bezala\n");
printf("1 entzulea: ");
scanf("%d %d %d", &b1, &b2, &b3);
while (b1!=-1){
  botoak[deiKopurua][0]= b1;
  botoak[deiKopurua][1]= b2;
  botoak[deiKopurua][2]= b3;
  deiKopurua=deiKopurua+1;
  printf("%d entzuleak: ", deiKopurua);
  scanf("%d %d %d", &b1, &b2, &b3);
}

/*Abesti bakoitzak lortu duen puntu kopurua*/
for (a=0; a<AbestiKop; a=a+1){
  abestiak[a]=0;
}
for (entzulea=0; entzulea<deiKopurua; entzulea=entzulea+1){
  abestiak[botoak[entzulea][0]]= abestiak[botoak[entzulea][0]]+3;
  abestiak[botoak[entzulea][1]]= abestiak[botoak[entzulea][1]]+2;
  abestiak[botoak[entzulea][2]]= abestiak[botoak[entzulea][2]]+1;
}
```

```
printf("\n");
for (a=0; a<AbestiKop; a=a+1){
  printf("%d abestiak %d puntu\n", a, abestiak[a]);
}

/*Puntu gehien duen abestia, abesti irabazlea*/
if (abestiak[0]>abestiak[1]) {
    lehena=0;
    bigarrena=1;
}
else {
    lehena=1;
    bigarrena=0;
}
for (a=2; a<AbestiKop; a=a+1){
  if (abestiak[lehena]<abestiak[a]){
    bigarrena=lehena;
    lehena=a;
  }
  else {
    if (abestiak[a]>abestiak[bigarrena]) {bigarrena = a;}
  }
}
printf("Abesti irabazlea: %d. abestia\n", lehena);
printf("2.a: %d abestia\n", bigarrena);
/*Dei egin duten entzuleen puntuazioak lortzen*/
puntuMax=0;
for (entzulea=0; entzulea<deiKopurua; entzulea=entzulea+1){
  ePuntuak=0;
  if ((botoak[entzulea][0]==lehena) ||
      (botoak[entzulea][1]==lehena) ||
      (botoak[entzulea][2]==lehena)){
    ePuntuak=ePuntuak+30;
  }
  if ((botoak[entzulea][0]==bigarrena) ||
      (botoak[entzulea][1]==bigarrena) ||
```

```
     (botoak[entzulea][2]==bigarrena)){
    ePuntuak=ePuntuak+20;
  }
  if (ePuntuak==50){
    ePuntuak=ePuntuak+10;
  }
  printf("%d  entzuleak: %d puntu\n", entzulea, ePuntuak);
  if (ePuntuak>puntuMax){
    puntuMax=ePuntuak;
    txapelduna=entzulea;
  }
}
printf("\n");
printf("Irabazlea: %d\n", txapelduna);

printf("\n\n");
system("PAUSE");
}
```

GASOLINDEGIA

AMOREBIETA izeneko gasolindegiak, aste oso betean hornigailu desberdinen erabilpenaren azterketa lan bat egitea eskatu digu. Gasolindegiak 12 hornigailu dauzka eta hornigailu bakoitzak 5000 litroko tankea dauka. Gauero, tanke guztiak guztiz betetzen dira, hau da, goizero hornigailu bakoitzak 5000 litro izango ditu saltzeko.

Egunean zehar bezeroak etortzen diren heinean, gasolindegiko langileak konputagailuan bikoteak sartzen ditu, hornigailu zenbakia eta eskatutako litroak adierazten dutelarik. Lanegun bateko datu amaiera adierazteko -1 zenbakia emango du hornigailua eskatzean eta beste edozein balio litroak eskatzean. Prozesu hau asteko egun guztietarako errepikatuko da.

Datu sarreran zehar, eskatutako gasolina kopurua hornigailu batean ez badago, hornigailu horretan daudenak emango zaizkio, hau hustuz, eta zenbat izan diren jakinaraziko zaio langileari.

Adibidea:

```
Eguna 0:
    Hornigailu, litroak: 7        40
    Hornigailu, litroak: 11       300
    Hornigailu, litroak: 11       400
    Hornigailu, litroak:  0       36
            ...
    Hornigailu, litroak: 11       400
           100 litrorekin beteko da.
    Hornigailu, litroak:  0       27
    Hornigailu, litroak: 11       300
           0 litrorekin beteko da.
    Hornigailu, litroak:  2       57
    Hornigailu, litroak: -1       0
Eguna 1:
...
Eguna 6:
...
```

Jasotako datuez gain, asteko informazio hau ere jaso beharko da:

1. Hornigailu bakoitzak gordetzen duen gasolina mota, bost gasolina mota daudela jakinik.

   0) EuroSuper'95

   1) Extra'98

   2) Star'97

   3) ExtraDiesel

   4) Gasoleo

Adibidea:

```
Hornigailu bakoitzaren gasolina mota sartu:
       0 Hornigailua: 0
       1 Hornigailua: 1
       2 Hornigailua: 0
       …
       11 Hornigailua: 4
```

	0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*
HornigailuMota	0	1	0	1	2	2	3	0	3	0	4	4

2. Beste taula batetan saldutako gasolina mota bakoitzeko litroko irabaziak (hauek Eurotan etorriko dira).

   Adibidea:

```
Eman gasolina mota bakoitzeko irabaziak litroko:
       0 gasolina mota: 0.02
       1 gasolina mota: 0.03
       …
       4 gasolina mota: 0.02
```

	0	*1*	*2*	*3*	*4*
IrabaziMota	0.02	0.03	0.04	0.01	0.02

Ondokoa kalkulatzea eskatzen da:

a) Esan zein egunetan eta zein hornigailu geratu den gasolinarik gabe.

b) Gasolina mota kontutan hartu gabe, zein egunetan saldu da litro gehien.

c) Esan zein gasolina motak eman duen irabazi gehien eta zenbatekoa izan den hau.

PROBLEMAREN ANALISIA

Enuntziatuko konstanteak

1. *HornigailuKop* = 12. Gasolindegiak dituen hornigailu kopurua.
2. *Astea* = 7. Azterketa egingo den egun kopurua.
3. *GasolinaMotaKop* = 5. Gasolindegiak saltzen dituen erregai mota kopurua.

Sarrerarako beharrezkoak diren aldagaien adierazpidea

4. *amorebieta* izeneko eta *Astea* x *HornigailuKop* posizio dituen matrizea behar dugu, non hornigailu bakoitzari saltzeko geratzen zaizkion litroak gordeko diren. Egun bakoitzaren hasieran hornigailu bakoitzak 5000 litro izango ditu saltzeko.

amorebieta	*0*	*1*	*...*	*HornigailuKop-1*
0				
1				
...				
Astea-1				

5. *hornigailuMota* izeneko eta *HornigailuKop* posizio dituen bektore bat behar dugu, non hornigailu bakoitzak saltzen duen erregai mota gordeko dugun.

	0	*1*	*...*	*HornigailuKop-1*
hornigailuMota				

6. *irabaziMota* izeneko eta *GasolinaMotaKop* posizio dituen bektorea behar dugu, non erregai bakoitzak saldutako litro bakoitzeko utziko digun irabazia gordeko dugun.

	0	*1*	*...*	*GasolinaMotaKop-1*
irabaziMota				

Emaitzetarako beharrezkoak diren aldagaien adierazpidea

7. *salmentakEgun* izeneko eta *Astea* posizio dituen bektorea behar dugu, non asteko egun bakoitzean erregai guztiak kontutan izanda saldutako litroak gordeko ditugun.

	0	*1*	*...*	*Astea-1*
salmentakEgun				

8. *irabaziak* izeneko eta *GasolinaMotaKop* posizio dituen bektorea behar dugu, non erregai bakoitzarekin irabazitako euroak gordeko ditugun bertan.

	0	*1*	*...*	*GasolinaMotaKop-1*
irabaziak				

Problema ebazteko jarraibideak

1. *amorebieta* matrizea 5000ra hasieratu.
2. Egun bakoitzerako (*e*):
    a. Lehen salmenta irakurri (*hor, lit*)
    b. Hornigailu zenbakia -1en desberdina den bitartean egin:
        i. Hornigailuan nahiko litro baldin badago (hau da, eskatutako litroak (*lit*) hornigailuan daudenak (*amorebieta[e][hor]*) baino handiagoa edo berdina bada)

            Orduan eskatutako litroak kendu hornigailu horri

            Bestela, geratzen diren litroekin, hau da, emango zaizkion litroekin mezua idatzi pantailan eta hornigailua egun horretan 0 litrorekin utzi.

        ii. Hurrengo salmenta irakurri (*hor, lit*)
3. Hornigailu bakoitzerako saltzen duen erregai mota irakurri eta *hornigailuMota* bektorean gorde.
4. Erregai bakoitzeko irakurri saldutako litro bakoitzak ematen digun irabazia eta *irabaziMota* bektorean gorde.
5. *amorebieta* matrizeko posizio guztiak begiratu eta 0ko bat aurkitzen dugunean pantailan idatzi honen posizioa, honek adieraziko baitu zein egunetan zein hornigailu geratu den erregairik gabe.
6. Salmenta gehien izan duen eguna kalkulatzeko:
    a. Egun bakoitzerako hornigailu guztien artean saldutako litroak batu eta *salmetnakEgun* bektorean gorde. Hornigailu bakoitzak saldutakoa kalkulatzeko 5000ri *amorebieta* matrizean dugun balioa kenduz lortuko dugu.
    b. *salmentakEgun* bektoreko balio maximoaren posizioa kalkulatu eta idatzi.
7. Hornigailu bakoitzerako:
    a. Egun guztien artean saldutako litroak kalkulatu (gogoratu, hornigailu baten egun batean saldutako litroak kalkulatzeko 5000ri matrizean dugun balioa kendu behar diogula) eta *hornigailuan* aldagaian gorde.
    b. Hornigailu horrek saltzen duen erregai mota lortu (*hornigailuMota* bektorean gordea dagoena) eta *erregai* aldagaian gorde.

c. Saldutako litroak (*hornigailuan*, 7-a atalean kalkulatua) batu *irabaziak* bektorean *erregai* aldagaiak dioen posizioan.

(*irabaziak[erregai]=hornigailu*+i*rabaziak[erregai]*)

8. Erregai mota bakoitzerako:

a. *irabaziak* bektorean gordetako balioa bidertu *irabaziMota* bektoreko balioarekin eta emaitza *irabaziak* bektorean utzi.

9. Irabazi gehien eman duen erregaia kalkulatu eta pantailan idatzi. Horretarako *irabaziak* bektoreko balio maximoaren posizioa kalkulatu behar da.

C KODEKETA

```
#include <stdlib.h>
#include <stdio.h>
#define Astea 7
#define HornigailuKop 12
#define GasolinaMotaKop 5
main(){
int amorebieta[Astea][HornigailuKop];
int hornigailuMota[HornigailuKop], salmentakEgun[Astea];
float irabaziMota[GasolinaMotaKop], irabaziak[GasolinaMotaKop];
int g,h,hor,lit,e,eMax,gMax,egunekoa,hornigailuan, erregai;

for (e=0; e<Astea; e=e+1) {
  for (h=0; h<HornigailuKop; h=h+1){ amorebieta[e][h]=5000;}
}
for (e=0; e<Astea; e=e+1){
  printf("Eguna %d:\n", e);
  printf("   Hornigailua, litroak: ");
  scanf("%d %d",&hor,&lit);
  while (hor != -1){
    if (amorebieta[e][hor]>=lit)
      { amorebieta[e][hor]=amorebieta[e][hor]-lit;}
    else {printf("\t%d litrorekin beteko da\n",
                  amorebieta[e][hor]);
        amorebieta[e][hor]=0 ;
         }
    printf("   Hornigailua, litroak: ");
    scanf("%d %d",&hor,&lit);
  }
}
/*hornigailu bakoitzak metatzen duen gasolina motaren jasotzea*/
printf("Eman ordenaturik hornigailu bakoitzak metatzen duen"
        " gasolina mota: \n");
for (h=0; h<HornigailuKop; h=h+1) {
  printf("%d Hornigailua: ", h);  scanf("%d",&hornigailuMota[h]);
}
```

```
/*Saldutako erregai mota bakoitzeko litroko irabaziak*/
printf("Eman ordenaturik erregai mota bakoitzeko litro baten ");
printf("salmentak suposatzen duen irabazia: \n");
for (g=0; g<GasolinaMotaKop; g=g+1) {
  printf("%d gasolina mota:");
  scanf("%f",&irabaziMota[g]);
}

/* a) Zein egunetan zein hornigailu geratu da gasolinarik gabe?*/
for (e=0; e<Astea; e=e+1) {
  for (h=0; h<HornigailuKop; h=h+1) {
    if (amorebieta[e][h]==0)
      {printf("%d egunean %d hornigailua guztiz hustu zen.\n",
              e,h);}
      }
}

/*b) Zein egunetan saldu da litro gehien?*/
for (e=0; e<Astea; e=e+1) {
  salmentakEgun[e]=0;
  for (h=0; h<HornigailuKop; h=h+1)
  { salmentakEgun[e]= salmentakEgun[e]+(5000-amorebieta[e][h]);}
}
eMax=0;
for (e=1; e<Astea; e=e+1) {
  if (salmentakEgun[e]>salmentakEgun[eMax]){ eMax=e; }
}
printf("Gehien saldu den asteko eguna %d izan da,", eMax);
printf("%d litro saldu zirelarik.\n\n", salmentakEgun[eMax]);

/*c) Zein gasolinak eman du irabazi gehien eta zenbatekoa izan da
hau?*/
for (h=0; h<HornigailuKop; h=h+1) {
  hornigailuan=0;
  for (e=0; e<Astea; e=e+1)
    { hornigailuan= hornigailuan+(5000-amorebieta[e][h]); }
  erregai = hornigailuMota[h];
```

```
  irabaziak[erregai]=irabaziak[erregai] + hornigailuan;
}
/* erregai bakoitzak emandako irabazia kalkulatu */
for (g=0; g<GasolinaMotaKop; g=g+1)
  { irabaziak[g]= irabaziMota[g] * irabaziak[g];}
/* irabazi gehien eman duen erregaia kalkulatu */
  gMax=0;
  for (g=1; g<GasolinaMotaKop; g=g+1) {
    if (irabaziak[g]>irabaziak[gMax]) {gMax=g;}
  }
printf("Irabazi gehien eman duen erregaia %d izan da,",gMax);
printf(" irabazia %7.2f eurokoa izanik",irabaziak[gMax]);

printf("\n\n");
system("PAUSE");
}
```

DENDA

Denda txiki batek urte batean zehar saldutako artikulu desberdinen informazioa gordetzen du. Dendan 100 artikulu desberdin saltzen dira. Eginbeharrekoak jarraian datozen ataza hauek dira:

1. Hamabi hiletan zehar artikulu desberdinen salmentak irakurri (unitateetan).
2. Artikulu bakoitzaren prezioa irakurri eta salmenta gehien izan duen hilabetea bilatu.
3. 2001 urteko uztaila eta abenduko tartean artikulu bakoitzeko eginiko salmenta guztien batura irakurri. Ondoren datu hauek tarte bereko 2002ko datuekin konparatu (lehen puntuan irakurriak) eta adierazi zenbat artikulutan gainditu diren 2001eko salmentak eta zeintzuk diren hauek.

Adibidea:

☞ 1 atalerako)

Artikuluen salmentak irakurri jarraian datorren modura:

```
Artikulua, hilabetea, salmentak: 1   5      800   (unitateak
dira)
Artikulua, hilabetea, salmentak: 0   4      700
Artikulua, hilabetea, salmentak: 4   5      100
Artikulua, hilabetea, salmentak: 6   5      800
Artikulua, hilabetea, salmentak: 6   11     800
..............
..............
Artikulua, hilabetea, salmentak: 1   0      800
Artikulua, hilabetea, salmentak: -1  0      0
```

Adibidean ikus daitekeen moduan datuak hirukoteetan (Artikulua, hilabetea, salmenta) etorriko dira, baina ez dira ordenatuta etorriko. Honetaz gain informazioa bakarrik etorriko da salmentak egon diren hilabete eta artikuluentzat. Salmenta balioak artikulu horretatik hilabete horretan saldutako kopurua adieraziko du.

☞ 2. atalerako pantailan lortu beharreko emaitzaren adibidea)

```
Salmenta gehien izan dituen hilabetea 3na da 1000,50
eurorekin.
```

Hau egin ahal izateko aurrez artikuluen prezioak eurotan irakurri behar dira, jarraian datorren moduan:

```
0 artikuluaren prezioa: 123,50
1 artikuluaren prezioa: 12,03
2 artikuluaren prezioa: 1,73
............
............
```

```
99 artikuluaren prezioa: 129
```

☞ 3 atalerako emaitzaren adibidea)

```
2001 urtearekin alderatuta uztaila eta abendua bitarteko
hiletan salmentak gainditu dituzten artikuluak 2, 6, 44, 99
dira.

Gainditu duten artikuluen totala 4 da.
```

Hau egin ahal izateko aurrez artikulu bakoitzeko 2001. urteko salmentak (eurotan) sartuko dira. Ondoko formatuan etorriko direlarik:

```
Uztaila eta abendua arteko hilabeteen artean eginiko
salmentak (eurotan) hauek dira:

0 artikuluaren salmentak:     970,52
1 artikuluaren salmentak:     8870,75
2 artikuluaren salmentak:     1170
...................
...................
99 artikuluaren salmentak:    9098,70
```

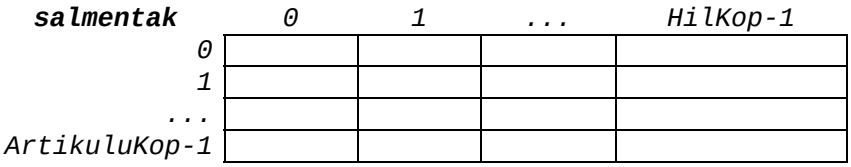
PROBLEMAREN ANALISIA

Enuntziatuko konstanteak

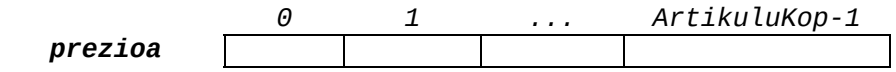
1. *HilKop* = 12. Azterketa egingo den hil kopurua.
2. *ArtikuluKop* = 100. Dendak saltzen dituen produktu kopurua.
3. *Uztaila*=7. 2001eko salmentekin alderatzea eskatzen diguten lehen hilabetea.

Sarrerarako beharrezkoak diren aldagaien adierazpidea

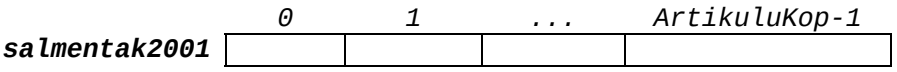
4. *salmentak* izeneko eta *ArtikuluKop* x *HilKop* posizio dituen matrizea behar dugu, bertan hil bakoitzean artikulu bakoitzetik saldutako unitateak gordeko ditugularik.



5. *prezioa* izeneko eta *ArtikuluKop* posizio dituen bektorea behar dugu, non produktu bakoitzaren prezioa gordeko dugun.

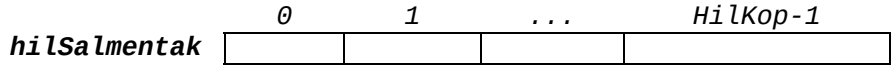


6. *salmentak2001* izeneko eta *ArtikuluKop* posizio dituen bektorea behar dugu, bertan 2001eko urtean uztaila – abendua hilabeteetan artikulu bakoitzarekin irabazitakoa gordeko dugun.

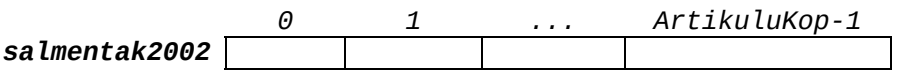


Emaitzetarako beharrezkoak diren aldagaien adierazpidea

7. *hilSalmentak* izeneko eta *HilKop* posizio dituen bektorea behar dugu, non artikulu guztiak kontuan izanik hilabete bakoitzean izandako irabaziak gordeko ditugun.



8. *salmentak2002* izeneko eta *ArtikuluKop* posizio dituen bektorea behar dugu, bertan 2002ko urtean uztaila – abendua hilabeteetan artikulu bakoitzarekin irabazitakoa gordeko dugun.



Problema ebazteko jarraibideak

1. *salmentak* matrizea 0ra hasieratu.

2. Lehen hirukotea irakurri (*art, hil, sal*)
3. Irakurritako hirukoteko lehen balioa (*art*) -1en desberdina den bitartean egin,
    a. Irakurritako salmentak *salmentak* matrizean gorde, *art*-k dioen lerroan eta *hil*-ek dioen zutabean.
    b. Hurrengo hirukotea irakurri (*art, hil, sal*)
4. Artikulu bakoitzerako bere prezioa irakurri eta *prezioa* bektorean gorde.
5. Hil bakoitzerako,
    a. Hilabeteko irabaziak 0ra hasieratu (*hilSalmentak*)
    b. Artikulu bakoitzaren (*art*) irabaziak hileko irabazietara gehitu. Hilabeteko irabaziak kalkulatzeko, artikulu bakoitzetik saldutako unitate kopurua (*salmentak* matrizean gordeta) bidertu artikuluaren prezioarekin (*prezioa* bektorean gordeta).
6. Irabazi gehien izan duen hilabetea kalkulatu eta idatzi. Horretarako *hilSalmentak* bektoreko balio maximoaren posizioa kalkulatu eta idatzi.
7. 2001 urtean uztaila – abendua tartean artikulu bakoitzarekin izandako irabaziak irakurri eta *salmentak2001* bektorean gorde.
8. 2002ko uztaila – abendua bitartean artikulu bakoitzak izandako irabaziak kalkulatuko ditugu, horretarako pausu hauek emango ditugu:

    Artikulu bakoitzerako,
    a. Artikuluaren salmentak 0ra hasieratu (*salmentak2002*)
    b. Uztaila – abendua tarteko hilabete bakoitzerako hil horretan saldutako artikulu kopurua (*salmentak* matrizean gordea) gehitu orain arte gehitutakoei (*salmentak2002* bektorean gordea).
    c. Saldutako artikulu kopurua (*salmentak2002* bektorean gordea) bidertu artikuluaren prezioarekin (*prezioa* bektorean gordea).
9. 2002 urtean irabazi gehien izandako artikuluen kontadorea (*kop*) 0ra hasieratu.
10. Artikulu bakoitzerako (*art*),
    a. 2002. urteko salmentak (*salmentak2002*) 2001eko salmentak (*salmentak2001*) baino handiagoak badira orduan artikulu zenbakia pantailan idatzi eta kontadorea (*kop*) batean handitu.
11. Pantailan idatzi 2002an salmenta handiagoa izan duten artikuluen kontadorea (*kop*).

C KODEKETA

```
#include <stdio.h>
#include <stdlib.h>
#define ArtikuluKop 100
#define HilKop 12
#define Uztaila 7
main(){
int salmentak[ArtikuluKop][HilKop];
int prezioa[ArtikuluKop], salmentak2001[ArtikuluKop];
int salmentak2002[ArtikuluKop], hilSalmentak[HilKop];
int hSalMax, art, hil, sal, kop;

/*Salmentak matrizea 0-z hasieratu ez dugulako informaziorik
irakurriko salmentarik egon ez den kasuetan*/
for (art=0; art<ArtikuluKop; art=art+1){
  for (hil=0; hil<HilKop; hil=hil+1){
    salmentak[art][hil]=0;
  }
}
/*Artikuluen salmentak hilabeteko irakurri*/
printf("Artikuluen salmentak sartu: \n");
printf("Artikulua, hilabetea, salmentak:");
scanf("%d %d %d", &art, &hil, &sal);
while (art!=-1){
  salmentak[art][hil]=sal;
  printf("Artikulua, hilabetea, salmentak:");
  scanf("%d %d %d", &art, &hil, &sal);
}

/*Artikulu bakoitzaren prezioa irakurri*/
printf("Sartu artikulu bakoitzaren prezioa:\n");
for (art=0; art<ArtikuluKop; art=art+1){
  printf("Eman %d artikuluaren prezioa: ", art);
  scanf("%d", &prezioa[art]);
}
```

```
/*Hilabete bakoitzean saldutako produktuengatik irabazitakoa
kalkulatu*/
for (hil=0; hil<HilKop; hil=hil+1){
  hilSalmentak[hil]=0;
  for (art=0; art<ArtikuluKop; art=art+1){
    hilSalmentak[hil]= hilSalmentak[hil]+
                       (salmentak[art][hil]*prezioa[art]);
  }
}
/*hilSalmentak bektoreko maximoaren posizio kalkulatu*/
hSalMax=0;
for (hil=0; hil<HilKop; hil=hil+1){
  if (hilSalmentak[hSalMax]<hilSalmentak[hil]){
    hSalMax = hil;
  }
}
printf("Gehien irabazi den hilabetea %d da",hSalMax);
printf("%d euroekin.\n", hilSalmentak[hSalMax]);

/*2001eko uztaila eta abendu arteko artikuluko irabaziak irakurri*/
for (art=0; art<ArtikuluKop; art=art+1){
  printf("%d artikuaren salmentak: ", art);
  scanf("%d", &salmentak2001[art]);
}
/*2002ko uztaila eta abendu tarteko artikulu irabaziak kalkulatu*/
for (art=0; art<ArtikuluKop; art=art+1){
  salmentak2002[art]=0;
  for (hil=Uztaila; hil<HilKop; hil=hil+1){
    salmentak2002[art]=salmentak2002[art]+salmentak[art][hil];
  }
  salmentak2002[art]=salmentak2002[art]*prezioa[art];
}

/*konparatu 2001 eta 2002ko uztaila eta abendu arteko artikuluen
irabaziak*/
printf("2001 urtearekin alderatuta uztaila eta abendua bitarteko
        hilabeteetan salmentak gainditu dituzten artikuluak \n");
```

```
kop=0;
for (art=0; art<ArtikuluKop; art=art+1){
  if (salmentak2001[art]<salmentak2002[art]){
    printf("%d, ", art);
    kop=kop+1;
  }
}
printf(" dira\n");
printf("Gainditu duten artikuluen totala %d da.\n", kop);

printf("\n\n");
system("PAUSE");
}
```

ZENTRAL ELEKTRIKOA

Zentral elektriko batek egunero hornitzen duen energi kantitateari buruzko azterlan bat egun nahi du. Horretarako gure laguntza eskatu dute programa informatiko bat gara dezagun. Programa honek, egunero hornitutako potentzia tipikoen (megawattetan neurtua) datuak jasotzea ahalbideratu behar du gehienez 52 astetarako (urtebeterako).

Eskatzen dena da:

1. Datu sarrera programa ezazu aste bakoitzeko 7 egunetan hornitutako potentziak jasotzeko.

   Datu sarrera amaitu nahi denean (adibidez, azterlana 52 aste baino gutxiagotarako egin nahi den kasuetarako), asteko lehen eguneko potentzia bezala –1 balioa sartu beharko da.

   Adibidea: (beltzez erabiltzaileak sakatzen dituen datuak)

```
0 astea
       0 egunean emandako potentzia: 207
       1 egunean emandako potentzia: 301
       2 egunean emandako potentzia: 222
       3 egunean emandako potentzia: 302
       4 egunean emandako potentzia: 22
       5 egunean emandako potentzia: 167
       6 egunean emandako potentzia: 125
1 astea
       0 egunean emandako potentzia: 367
       1 egunean emandako potentzia: 60
       2 egunean emandako potentzia: 120
       3 egunean emandako potentzia: 111
       4 egunean emandako potentzia: 301
       5 egunean emandako potentzia: 400
       6 egunean emandako potentzia: 434
2 astea
       0 egunean emandako potentzia: 211
       1 egunean emandako potentzia: 72
       2 egunean emandako potentzia: 441
       3 egunean emandako potentzia: 102
       4 egunean emandako potentzia: 21
       5 egunean emandako potentzia: 203
       6 egunean emandako potentzia: 317
3 astea
       0 egunean emandako potentzia: 401
       1 egunean emandako potentzia: 340
       2 egunean emandako potentzia: 161
       3 egunean emandako potentzia: 297
       4 egunean emandako potentzia: 441
       5 egunean emandako potentzia: 117
       6 egunean emandako potentzia: 206
4 astea
       0 egunean emandako potentzia: -1
```

2. Asteko egun bakoitzerako batez besteko potentzia kalkulatu eta idatzi. Behin hau kalkulaturik, adierazi zein egun izan den batez besteko potentzia handien izan duen eguna.
3. Azterlana egin den tarterako batez besteko potentzia kalkulatu eta idatzi.
4. Aurreko atalean kalkulatutako batez besteko potentzia baino handiagoa izan duten egun kopurua kalkulatu eta idatzi.
5. Merkatu librea dela eta megawatt bakoitzeko irabazten dena astero ezartzen da. Sar itzazu taula batean azterlanaren tarteko aste bakoitzeko irabaziak (€/megawatt). Informazio honekin tratatutako asteetako irabaziak eta zentraleko irabazi totalak kalkula eta pantailan idatzi.

PROBLEMAREN ANALISIA

Enuntziatuko konstanteak

1. *AsteKop* = 52. Azterketa egingo den aste kopurua.
2. *EgunKop* = 7. Aste bakoitzetik tratatutako egun kopurua.

Sarrerarako beharrezkoak diren aldagaien adierazpidea

3. *potentziak* izeneko eta *AsteKop X EgunKop* posizio dituen matrizea behar dugu, asteko egun bakoitzean kontsumitutako potentzia gordetzeko.

potentziak	0	1	...	EgunKop-1
0				
1				
...				
AsteKop-1				

4. *irabA* izeneko eta *AsteKop* posizio dituen bektorea behar dugu, aste bakoitzeko megawattioaren prezioa eurotan gordetzeko.

	0	1	...	AsteKop-1
irabA				

Emaitzetarako beharrezkoak diren aldagaien adierazpidea

5. *eBB* izeneko eta *EgunKop* posizio dituen bektorea behar dugu, non asteko egun bakoitzean kontsumitutako batez besteko potentzia gordeko den.

	0	1	...	EgunKop-1
eBB				

Problema ebazteko jarraibideak

1. Asteko kontadorea 0ra hasieratu (*asteak*).
2. Lehen asteko lehen egunari dagokion potentzia irakurri (*balioaAstelehena*).
3. Irakurritako potentzia (*balioaAstelehena*) -1en desberdina den bitartean eta asteen kontadorea (*asteak*) *AsteKop* baino txikiagoa den bitartean egin:
    a) Irakurritako potentzia *potentziak* matrizean gorde, *asteak* dioen lerroan eta 0 zutabean.
    b) Gainontzeko egunetarako (*egun*) potentzia irakurri eta *potentziak* matrizean gorde *asteak* lerroan eta *egun* zutabean.
    c) *asteak* kontadorea batean handitu.
    d) Asteko lehen egunari dagokion potentzia irakurri (*balioaAstelehena*).

4. Egun bakoitzerako kontsumitutako batez besteko potentzia kalkulatu eta *eBB* bektorean gorde. Honekin batera astean kontsumitutako potentziaren batez bestekoa ere kalkulatu. Horretarako emango diren pausoak:
   a. Asteko batez besteko potentzia 0ra hasieratu (*bbPotentzia*).
   b. Asteko egun bakoitzerako (*e*)
      i) Eguneko potentzien batura 0ra hasieratu (*eBB[e]=0*).
      ii) Tratatutako aste bakoitzerako, egun horretan kontsumitutako potentzia metatu asteko potentziara (*eBB*).
      iii) Kontsumitutako potentzia zatitu tratatutako aste kopuruarekin.
      iv) Eguneko potentzia asteko batez besteko potentziari gehitu
5. Asteko batez besteko potentzia kalkulatu, horretarako eguneko batez besteko potentzia zatitu egun kopuruarekin. Emaitza pantailan idatzi.
6. Kalkulatu kontsumo gehien izan duen asteko eguna. Horretarako *eBB* bektoreko balio maximoaren posizioa bilatu behar da.
7. Batez besteko potentzia gainditzen duten egunen kontadorea (*egunak*) 0ra hasieratu.
8. Asteko egun bakoitzerako (*e*)
   a) Tratatutako aste bakoitzerako (*a*),
      i. Aste horretako egun horretan (*potentziak[a, e]*) kontsumitutako potentzia batez besteko (*potBB*) potentzia baino handiagoa bada, orduan kontadorea (*egunak*) handitu.
9. Pantailan idatzi *egunak* kontadorearen balioa.
10. Tratatutako asteentzat eskatu megawatt bakoitzeko irabaziak eta *irabA* taulan gorde.
11. Asteko irabazia eta irabazi totala kalkulatu, horretarako:

    Zerora hasieratu irabazi totala (*irabaziTotala*)

    Aste bakoitzerako,

    a. Asteko irabazia 0z hasieratu (*astekoIra*)
    b. Egun bakoitzerako kontsumitutako potentzia metatu eta *astekoIra* aldagaian gorde.
    c. Asteko irabazia (*astekoIra*) biderkatu aste horretako megawattioaren prezioarekin (*irabA* bektorean gordea) eta lortutako emaitza pantailan idatzi.

d. Asteko irabazia (*astekoIra*) irabazi totalera gehitu (*astekoIra*).

12. Zentralaren irabazi totala (*irabaziTotala*) pantailan idatzi.

C KODEKETA

```
//Zentral Elektrikoa;
#include <stdio.h>
#include <stdlib.h>
#define AsteKop 52
#define EgunKop 7
main(){
int potentziak[AsteKop][EgunKop];
float eBB[EgunKop];
float irabA[AsteKop];
float bbPotentzia, irabaziTotala, astekoIra;
int asteak, egunak, balioaAstelehena, e, a, eMax;

/*Datuen irakurketa eta jasotzea*/
asteak=0;
printf("%d astea: \n", asteak);
printf("    0 egunean kontsumatutako potentzia: ");
scanf("%d", &balioaAstelehena);
while ((balioaAstelehena!=-1) && (asteak<AsteKop)){
  potentziak[asteak][0]=balioaAstelehena;
  for (e=1; e<EgunKop; e=e+1){
    printf("    %d egunean kontsumatutako potentzia: ", e);
    scanf("%d", &potentziak[asteak][e]);
  }
  asteak=asteak+1;
  printf("%d astea: \n", asteak);
  printf("    0 egunean kontsumatutako potentzia: ");
  scanf("%d", &balioaAstelehena);
}
printf("\n");

/*Asteko egun bakoitzeko bataz bestekoa eta bataz beste handiena
duen eguna*/
bbPotentzia = 0;
for (e=0; e<EgunKop; e=e+1){
  eBB[e]=0;
  for (a=0; a<asteak; a=a+1){
```

```
    eBB[e]=eBB[e]+potentziak[a][e];
  }
  eBB[e]=eBB[e]/asteak;
  printf("%d eguneko bataz bestekoa: %f\n", e, eBB[e]);
  bbPotentzia = bbPotentzia + eBB[e];
}
bbPotentzia=bbPotentzia/EgunKop;
printf("Epe osoko bataz bestekoa: %.2f\n", bbPotentzia);
/*Potentzia handieneko eguna kalkulatzen*/
eMax=0;
for (e=1; e<EgunKop; e=e+1){
  if (eBB[eMax]<eBB[e]){eMax=e;}
}
printf("Bataz besteko handiena duen eguna %d da\n",eMax);

/*Bataz besteko potentzia kontsumoa gainditu duten egunen kopurua*/
egunak=0;
for (e=0; e<EgunKop; e=e+1){
  for (a=0; a<asteak; a=a+1){
    if (potentziak[a][e]>bbPotentzia){egunak=egunak+1;}
  }
}
printf("Bataz besteko potentzia kontsumoa gainditu duten egunak"
       " %d\n", egunak);

printf("Eman lerro berean eta zuriunez banaturik aste bakoitzeko"
       " irabazia KWko: \n");
for (a=0; a<asteak; a=a+1){scanf("%f", &irabA[a]);}

irabaziTotala=0;
for (a=0; a<asteak; a=a+1){
  astekoIra=0;
  for (e=0; e<EgunKop; e=e+1)
    { astekoIra=astekoIra+potentziak[a][e];}
  astekoIra=astekoIra*irabA[a];
  printf("%d asteko irabazia %f \n", a, astekoIra);
  irabaziTotala=irabaziTotala+astekoIra;
```

```
}
printf("Orotara lortu den irabazia: %f\n", irabaziTotala);

printf("\n\n");
system("PAUSE");
}
```

ELCA COMAYOR LAPURRA

Elca Comayor jaunak bere negozioa nola doan jakiteko programa bat egitea erabaki du. Gure lagun Elca Comayor jaunak negozio berezia du, lapurreten munduan sartua dago eta asko interesatzen zaio bere kontuak ondo emango dituen programa bat garatzea.

Berak azkeneko 8 astetako kontuak gorde nahi ditu, kontutan hartu egunero egin dezakeela lan, bere enpresak ez baitu sekula ixten. Horrela, kontuak egiten doan heinean programari hiru datu ematen dizkio: astea (0 eta 7 bitartean), eguna (0 eta 6 bitartean) eta egun horretan zenbat lapurtu duen. Asteari dagokion tokian -1 sartzen duen momentuan programak datu sarrerari amaiera eman nahi zaiola ulertuko du

Adibidea: (beltzez erabiltzaileak sakatzen dituen datuak)

```
Sartu datuak (astea eguna kopurua): 3    5    300
Sartu datuak (astea eguna kopurua): 0    1    50
Sartu datuak (astea eguna kopurua): 5    6    179
Sartu datuak (astea eguna kopurua): 1    2    22
Sartu datuak (astea eguna kopurua): -1   0    0
```

Horretaz gain, 8 aste horietako gastuak kalkulaturik dauzka, hau da, badaki aste bakoitzeko bere gastuak zenbatekoak izan diren.

Adibidea:

```
0 asteko datua: 77
1 asteko datua: 60
...........
...........
7 asteko datua: 99
```

Elca Comayor jaunak, programari datu guzti hauek eman ondoren, honako hau jakin nahi du:

1. Zenbat lapurtu duen aste bakoitzeko eta zein izan den gehien lapurtu duen astea.
2. Zenbat lapurtu duen aste-egun bakoitzeko, hau da, zenbat lapurtu duen astelehenetan, astearteetan, ... eta zein den gehien lapurtzen duen aste-eguna.
3. Zenbat irabazi duen aste bakoitzeko eta guztira. Kontuan har, irabaziak kalkulatzeko aste horretan edukitako gastuak hartu behar direla kontutan.
4. Zein izan den lehenengo eguna ezer lapurtu ez duena.

Horretaz gain, poliziak atxilotzen badu, lapurtutakoaren arabera izango du zigorra:

```
5.000 € baino gutxiago→ Komunitaterako lanak egin beharko ditu
```

```
10.000€ baino gutxiago → 3 hilabete kartzelan
20.000€ baino gutxiago → 6 hilabete kartzelan
20.000€ baino gehiago → Urte bat kartzelan
```

Gure Elca Comayor jaunak interes handia du jakiteko atxilotzen badute, zenbat denbora egon beharko duen kartzelan, ez baitzaio kartzelan egoteko ideia hori gustatzen. Beraz, lapurtutakoaren arabera programak esango dio zenbat denbora egongo den kartzelan.

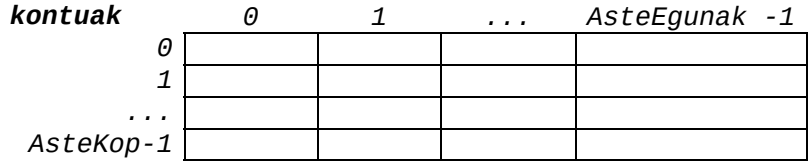
PROBLEMAREN ANALISIA

Enuntziatuko konstanteak

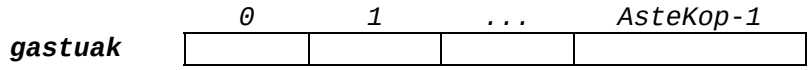
1. *AsteEgunak* = 7. Aste batek dituen egun kopurua.
2. *AsteKop* = 8. Lapurrak aztertuko dituen aste kopurua da.

Sarrerarako beharrezkoak diren aldagaien adierazpidea

3. *kontuak* izeneko eta *AsteKop* x *AsteEgunak* posizio dituen matrize bat behar da. Bertan egunero lapurtutako euroak 8 astetan zehar gordeko dira.

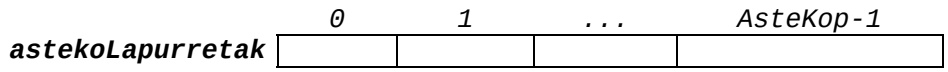


4. gastuak izeneko bektore bat behar dugu, *AsteKop* posizio dituena. Bertan lapurrak astero gastatzen duen euro kopurua metatuko da

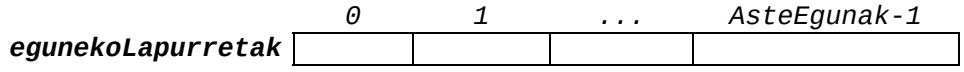


Emaitzetarako beharrezkoak diren aldagaien adierazpidea

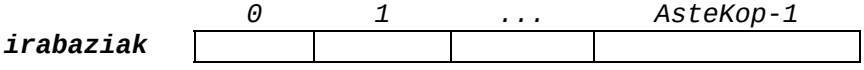
5. *astekoLapurretak* izeneko bektore bat beharko dugu, *AsteKop* posizioekin. Bertan aste bakoitzean lapurtu duen kopurua metatuko dugu.



6. *egunekoLapurretak* izeneko bektore bat beharko dugu, *AsteEgunak* posizioekin. Bertan egun mota bakoitzean lapurtu duen kopurua metatuko dugu.



7. *irabaziak* izeneko bektore bat behar dugu, *AsteKop* posizioekin, astero irabazten duen diru kopurua gordetzeko erabiliko duguna, hau da, lapurtutakoaren eta gastatutakoaren arteko diferentzia egin ostean.



Problema ebazteko jarraibideak

1. *kontuak* 0-z hasieratu.
2. Lapurretaren lehenengo (*a*, *e*, *k*) hirukotea irakurri.
3. *a* astea -1 balioa izan ezean:

   a. *kontuak* matrizean datuak erregistratu.
   b. hurrengo (*a*, *e*, *k*) hirukotea irakurri.
4. Aste bakoitzeko:
   a. Irakurri eta gorde aste horretako gastua.
5. *astekoLapurretak* bektorea hasieratu 0ra.
6. Aste bakoitzeko:
   a. Zenbat lapurtu duen kalkulatu.
7. Aste bakoitzeko:
   a. pantailaratu zenbat lapurtu duen.
8. Kalkulatu eta idatzi lapurreta kopuru maximoa egin duen astea
9. *egunekoLapurretak* bektorea hasieratu 0ra.
10. Asteko egun bakoitzeko:
    a. Zenbat lapurtu duen kalkulatu.
11. Kalkulatu eta idatzi lapurreta kopuru maximoa egin duen eguna.
12. Aste bakoitzeko:
    a. Irabazi duena kalkulatu, lapurtu duen kopuruari izan dituen gastuak kenduz.
13. Aste bakoitzeko irabaziak pantailaratu.
14. Kalkulatu eta idatzi orotara irabazi duena.
15. *amaitu* aldagaia 0-z hasieratu (=false), amaitu ez dela adierazteko.
16. *amaitu* 0 den eta matrize guztia aztertu gabe dagoen bitartean
    a. matrizea lerroka aztertu eta 0 duen lehenengo posizioa bilatu.
    b. 0 balioa duen posizio bat aurkitzen bada, amaitu aldagaiari 1 balioa (=true) esleitu.
17. 0 barne duen lehengo posizioaren koordenatuak idatzi (astea eta eguna).
18. Lapurtu duen kopurua batu.
19. Atxilotzen badute, ezarriko lioketen zigorra kalkulatu eta pantailaratu.

C KODEKETA

```
#include <stdio.h>
#include <stdlib.h>
// Elca Comayor lapurra
#define AsteEgunak 7
#define AsteKop 8
main(){
int kontuak[AsteKop][AsteEgunak];
int astekoLapurretak[AsteKop], gastuak[AsteKop];
int irabaziak [AsteKop], egunekoLapurretak[AsteEgunak];
int a, e, k, maxL, astMax, egMax, irabaziaOrora;
int ororaLapurtutako, ast0, eg0, amaitu;

// kontuak 0-z hasieratu
for (a=0;a<AsteKop;a++)
  for (e=0;e<AsteEgunak;e++)
    kontuak[a][e]=0;

// egunero lapurtu duena erregistratzera goaz
printf("Sartu datuak (astea eguna kopurua): ");
scanf("%d %d %d", &a, &e, &k);
while (a != -1) {
  kontuak[a][e]=kontuak[a][e]+k;
  printf("Sartu datuak (astea eguna kopurua): ");
  scanf("%d %d %d", &a, &e, &k);
}

// asteko gastuen erregistroa
for (a=0;a<AsteKop;a++) {
  printf("%d asteko datua: ", a);
  scanf("%d", &gastuak[a]);
}
printf("\n\n");
// aste bakoitzeko zenbat lapurtu duen eta lapurreta maximoko
// astea bektorearen hasieraketa
for (a=0;a<AsteKop;a++) astekoLapurretak[a]=0;
```

```
// aste bakoitzean lapurtu duena kalkulatu eta idatzi
for (a=0;a<AsteKop;a++) {
  for (e=0;e<AsteEgunak;e++)
    astekoLapurretak[a]=astekoLapurretak[a]+kontuak[a][e];
    printf("%d astean %d euro lapurtu ditu\n", a,
            astekoLapurretak[a]);
}

// lapurreta kopuru maximoko astearen kalkulua
maxL=astekoLapurretak[0];  astMax=0;
for (a=1;a<AsteKop;a++)
  if (maxL<astekoLapurretak[a])
  {maxL=astekoLapurretak[a]; astMax=a;}
// lapurreta kopuru maximoa egin duen astea idatzi
printf("%d astean lapurreta gehien egin duen astea izan"
       " da, %d euro lapurtu ditu.\n", astMax, maxL);

// zenbat lapurtu du asteko egun mota bakoitzeko
// eta zein egunetan egin du lapurreta maximoa
for (e=0;e<AsteEgunak;e++) egunekoLapurretak[e]=0;

for (e=0;e<AsteEgunak;e++)
  for (a=1;a<AsteKop;a++)
     egunekoLapurretak[e]= egunekoLapurretak[e] + kontuak[a][e];

maxL=egunekoLapurretak[0];  egMax=0;
for (e=0;e<AsteEgunak;e++)
  if (maxL<egunekoLapurretak[e])
  {maxL=egunekoLapurretak[e]; egMax=e;}
printf("\n\nLapurreta maximoko eguna %d da\n",egMax);

// astero zenbat irabazi duen eta orotara zenbat, gastuak
// kontuan izanik asteroko irabazi garbia kalkulatzen dugu
for (a=0;a<AsteKop;a++)  {
  irabaziak[a]=astekoLapurretak[a] - gastuak[a];
  printf("%d astean %d euro irabazi du.\n", a, irabaziak[a]);
}
```

```
//irabazitakoa orotara kalkulatzen eta idazten dugu
irabaziaOrora= 0;
for (a=0;a<AsteKop;a++) {
  irabaziaOrora= irabaziaOrora + irabaziak[a];
}
printf("\nOrotara irabazi duena %d euro da.\n", irabaziaOrora);
//lapurretarik gabeko eguna
amaitu = 0;  a=0;
while ((amaitu==0) && (a<AsteKop)) {
  e=0;
  while ((amaitu ==0) && (e< AsteEgunak)) {
    if (kontuak[a][e]==0)  { ast0=a; eg0=e; amaitu=1; }
    e++;
  };
  a++;
}
printf("Lapurretarik egin ez duen lehenengo eguna, %d asteko"
       " %d eguna da.\n", ast0, eg0);
//lapurtutakoaren arabera legokiokeen zigorra kalkulatzen dugu
// orotara lapurtutako kopurua kalkulatzen dugu, ez irabazi duena
ororaLapurtutako=0;
for (a=0;a<AsteKop;a++) {
  ororaLapurtutako= ororaLapurtutako + astekoLapurretak[a];
}

// zein zigor ezarriko litzaioke atxilotuko balute?
if (ororaLapurtutako<5000) {
  printf("\nKomunitaterako lanak egin beharko ditu."); }
else if (ororaLapurtutako<10000)
     { printf("\n3 hilabete kartzelan."); }
else if (ororaLapurtutako<20000)
     { printf("\n6 hilabete kartzelan."); }
else printf("\nUrte bat kartzelan.");

printf("\n\n");
system("PAUSE");
}
```

MEDI IGOERAK

10 mendi tontorren altuera metrotan islatzen duen 10 zenbaki osozko zerrenda dugu sarreran.

Ondoren, bikote zerrenda datorkigu, non bikoteko lehenengoaren balioa -1 denean bukatuko den datu horien sarrera. Bikotea 2 zenbaki osokoek osatzen dute (lehenengoa, 0 eta 6 arteko zenbakia, eta, bigarrena, 0 eta 9 artekoa). Balio hauek mendizalearen zenbakia eta mendizaleak igotako gailurra adierazten dute. Kontuan izan mendizale batek mendi bera behin baino gehiagotan igo dezakeela.

Ondokoa eskatzen da:

1. Problema ebazteko behar diren bektore edota matrizeak bete sarrerako datuekin (mendien altuera eta -1 0-z bukatutako bikote zerrenda).
2. Mendizale gutxien igotako mendien zenbakiak pantailan itzuli.
3. Metro gehien igota dituzten mendizaleen zenbaki kodeak pantailan itzuli.
4. Mendizale bakoitzeko igotako mendirik altuena pantailan itzuli.

PROBLEMAREN ANALISIA

Enuntziatuko konstanteak

1. *GailurKop* = 10. Mendizaleek igo ditzaketen gailur desberdinen kopurua.
2. *MendizaleKop* = 7. Azterketan parte hartzen duten mendizaleak.

Sarrerarako beharrezkoak diren aldagaien adierazpidea

3. *MendizaleKop* x *GailurKop* posizio dituen matrizea behar dugu. Bertan, mendizale bakoitzak gailur bakoitza zenbat aldiz lortu duen erregistratuko dugu.

Gailurrak	*0*	*1*	*...*	*GailurKop-1*
0				
1				
...				
MendizaleKop-1				

4. *GailurKop* posizio dituen bektore batean gailur bakoitzaren altuera metatuko dugu.

	0	*1*	*...*	*GailurKop-1*
Altuerak				

Emaitzetarako beharrezkoak diren aldagaien adierazpidea

5. *GailurKop* posizio dituen bektore bat behar dugu, gailur bakoitza garaitua izan den aldi kopurua jasotzeko.

	0	*1*	*...*	*GailurKop-1*
Igoerak				

6. *MendizaleKop* posizio dituen bektore bat behar dugu. Bertan, mendizale bakoitzak igo dituen metroak metatuko ditugu.

	0	*1*	*...*	*MendizaleKop-1*
IgotakoMet				

Problema ebazteko jarraibideak

1. Gailur bakoitzeko:
   a. Daukan altuera irakurri eta gorde
2. *Gailurrak* matrizea 0-z hasieratu.
3. Lehenengo (*m, g*) bikotea irakurri
4. Mendizalea desberdin -1 den bitartean (*m*!=-1), egin
   a. *Gailurrak* matrizean, gailurreko igoera berria erregistratu
   b. Hurrengo (*m, g*) bikotea irakurri
5. *Igoerak* bektorea hasieratu

6. Gailur bakoitzeko
    a. Mendizale guztiek egin dituzten igoerak batu
7. Gailurra lortu duten mendizale kopuru baxuen duten gailurrak kalkulatu
8. Igoera kopuru minimoa izan duten gailurrak idatzi
9. *IgotakoMet* bektorea hasieratu
10. Mendizale bakoitzeko
    a. Igo dituen metro kopurua kalkulatu dugu; horretarako igo dituen metro kopuruari, gailurraren altuera bider gailurra hartu duen aldi kopurua gehituko zaio.
11. Mendizale batek gehienez zenbat metro igo dituen kalkulatu
12. Metro kopuru maximoa igo dituzten mendizaleak idatzi
13. Mendizale bakoitzeko
    a. Igo duen gailur altuena zein izan den erabaki
    b. Gailurrik igo badu, gailur altuena idatzi; bestela, gailurrik igo ez duela idatzi

C KODEKETA

```
#include <stdio.h>
#include <stdlib.h>
#define GailurKop 10    // Mendi igoerak
#define MendizaleKop 7
main (){
int gailurrak [MendizaleKop][GailurKop];
int altuerak [GailurKop], igoerak[GailurKop];
int igotakoMet[MendizaleKop];
int m, g, minGP, maxMetP, maxBal, gailurra;

// Gailurren altuerak sakatzen dira
printf("\nGailurren altuerak saka itzazu: \n");
for (g=0; g<GailurKop; g++) {
  printf("%d gailurra: ", g);
  scanf("%d", &altuerak[g]);
}

// Gailurrak matrizearen hasieraketa
for (m=0; m<MendizaleKop; m++)
  for (g=0; g<GailurKop; g++)   gailurrak[m][g]= 0;

// Mendizaleek igotako gailurren sarrera
printf("Eman mendizalea eta gailurra: ");
scanf("%d %d", &m, &g);
while (m!=-1){
  gailurrak[m][g]++;
  printf("Eman mendizalea eta gailurra: ");
  scanf("%d %d", &m, &g);
}

// Bisita gutxien izan dituzten gailurren idazketa
// igoerak bektorearen hasieraketa
for (g=0; g<GailurKop; g++)
  igoerak[g]=0;
// Gailur bakoitza zenbat aldiz bisitatua izan den kalkatzen dugu
```

```
for (g=0; g<GailurKop; g++)
  for (m=0; m<MendizaleKop; m++)
   igoerak[g]=igoerak[g]+gailurrak[m][g];
// igoera kopuru minimoa bilatzen dugu
minGP=0;
for (g=1; g<GailurKop; g++)
  if (igoerak[g]<igoerak[minGP]) minGP=g;
// Bisita kopuru minimoa izan duten gailur guztiak idazten ditugu
printf("\nBisita gutxien izan dituzten gailurrak"
       " honako hauek dira: ");
for (g=0; g<GailurKop; g++)
  if (igoerak[g]==igoerak[minGP]) printf(" %d", g);
printf("\n");

// Metro gehien igo dituzten mendizaleen idazketa
// igotakoMet bektorearen hasieraketa
for (m=0; m<MendizaleKop; m++)   igotakoMet[m]=0;
// Mendizale bakoitzak zenbat metro igo dituen kalkulatzen dugu
for (m=0; m<MendizaleKop; m++)
  for (g=0; g<GailurKop; g++)
    igotakoMet[m]=igotakoMet[m] + altuerak[g]*gailurrak[m][g];
// Mendizaleek igo dituzten metro kopuru maximoa kalkulatu
maxMetP=0;
for (m=1; m<MendizaleKop; m++)
  if (igotakoMet[maxMetP]<igotakoMet[m]) maxMetP=m;
// Metro kopuru maximoa igo dituzten mendizaleen idazketa
printf("\nMetro gehien igo dituzten mendizaleak dira: ");
for (m=0; m<MendizaleKop; m++)
  if (igotakoMet[maxMetP]==igotakoMet[m]) printf(" %d",m);
printf("\n");

// Mendizale bakoitzak garaitu duen gailurrik altuena idatzi
printf("\nMendizale bakoitzak garaitu duen gailurrik altuena:\n");
for (m=0; m<MendizaleKop; m++) {
  maxBal=0;       // altuera maximoa
  for (g=0; g<GailurKop; g++) {
    if (0<gailurrak[m][g] &&  maxBal<altuerak[g]) {
```

```
        maxBal=altuerak[g];
        gailurra=g;
    }
  }
  if (maxBal!=0)
    printf("\n%d mendizaleak %d gailurra igo du", m, gailurra);
  else printf("\n%d mendizaleak ez du gailurrik lortu", m);
}

  printf("\n\n");
  system("PAUSE");
}
```

AUTOPISTA

Bilbo-Behobia autopistako gerenteak programa informatiko bat egitea eskatu digu. Programa honek, hilabete batean ohiko bezeroek egiten dituzten kontsumoen datuak jaso behar ditu. Bezero bakoitza ordainleku bakoitzetik zenbat aldiz pasatzen den gorde behar dugu, honela deskontu bereziak aplikatzeko eta zenbait datu estatistiko ateratzeko.

Ohiko bezero izateko kontratu bat sinatu behar da autopistakoekin. Momentu honetan, 1000 ohiko bezero eta 15 ordainleku desberdin ditu autopista honek.

a) Lehenik eta behin bektore batean ordainlekuen prezioak sartuko dira. Horretarako, programak erabiltzaileari ordainleku bakoitzaren prezioak sekuentzialki eskatuko dizkio. Ondoren, bezeroen datuak sartuko dira.

b) Datuen ematea bikoteka egingo da, bikoteko lehena bezeroaren kodea izanik eta bigarrena igarotako ordainlekuaren identifikazio kodea. Kontuan izan behar da, bezero bera ordainleku beretik behin baino gehiagotan igaro daitekeela. Honen ondorioz bezero horrek ordainleku bakoitzetik pasatu den aldi kopurua metatu beharko da.

Sarrerako bikote bakoitza irakurtzearekin batera, beste bi datu hauek ere kalkulatu beharko dira irakurritako bezero horrentzat: orain arte egin dituen bidai kopuru totala eta bidai horien kostu metatua. Kontuan izan, bidai baten kostua kalkulatzeko, tramu horren prezioari deskontu bat aplika dakiokegula ondorengo irizpideen arabera:

- Lehenengo 8 bidaiei, %25 deskontua
- 9.etik 20. bidaiei, %45 deskontua
- 21.etik aurrera %75 deskontua

Adibidea: (erabiltzailearen datuak beltzez azaltzen dira)

```
Kodea eta ordainleku kodea (amaitzeko -1 -1):
Bezeroa, ordainlekua:  1,   2
Bezeroa, ordainlekua:  8,   4
Bezeroa, ordainlekua:  1,   2
...
Bezeroa, ordainlekua: -1,  -1
```

1. Bezeroek maizen igarotzen duten ordainlekua identifikatu.
2. Bezero bakoitzeko maizen igarotzen duen ordainlekua identifikatu.
3. Bezero bakoitzarentzat kalkulatu zenbat aurreztu duen, ohiko bezeroa izateagatik.
4. Bezero bakoitzeko pantailaratu: eginiko bidai kopuru totala hil honetan, ordaindu beharreko kopurua totala eta aurreztutako kopurua ohiko

bezeroa izanagatik. Zerrendaketak honako formatua izan beharko du, adibidez:

```
Bezero kod.       Bidai tot.     Ordaindutakoa  Aurrezpena
0                 20             120            30
1                 10             79             15
```

PROBLEMAREN ANALISIA

Enuntziatuko konstanteak

1. *OrdainlekuKop*= 15. Programak kontrolatuko dituen ordainleku kopurua.
2. *BezeroKop* = 1000. Ohiko bezero kopurua.

Sarrerarako beharrezkoak diren aldagaien adierazpidea

3. *BezeroKop* x *OrdainlekuKop* posizio dituen matrize bat behar dugu ordainlekuak zeharkatzerakoan ohiko bezeroen bidaien ordainsariak metatzeko.

bidaiak	*0*	*1*	*...*	*OrdainlekuKop-1*
0				
1				
...				
BezeroKop-1				

4. *OrdainlekuKop* posizio dituen bektore bat behar dugu ordainleku bakoitzeko bidesariak metatzeko.

	0	*1*	*...*	*OrdainlekuKop-1*
bidaisariak				

Emaitzetarako beharrezkoak diren aldagaien adierazpidea

5. *BezeroKop* posizio dituen bektore bat behar dugu ohiko bezero bakoitzak eginak daramazkin bidai kopuruak metatzeko.

	0	*1*	*...*	*BezeroKop-1*
bidaiTotalak				

6. *BezeroKop* posizio dituen bektore bat behar dugu ohiko bezeroak ordaindu behar duen euro kopurua.

	0	*1*	*...*	*BezeroKop-1*
ordainduBehar				

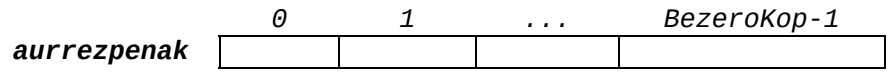
7. *OrdainlekuKop* posizio dituen bektore bat behar dugu ordainleku bakoitza zenbat aldiz zeharkatu den metatzeko.

	0	*1*	*...*	*OrdainlekuKop-1*
ordainlekuZeharkatzeKop				

8. *BezeroKop* posizio dituen bektore bat behar dugu, deskonturik aplikatuko ez balitzaio bezero batek ordaindu beharko lukeen euro kopurua metatzeko.

	0	*1*	*...*	*BezeroKop-1*
berezkoOrdainketa				

9. *BezeroKop* posizio dituen bektore bat behar dugu, ohiko bezero bakoitzaren euro aurrezpena metatzeko.



Problema ebazteko jarraibideak

1. Matrizea 0ra hasieratu
2. Ordainleku bakoitzeko:
   a. Ordainsaria irakurri eta gorde *bidaisariak* bektorean.
3. *bidaiTotalak* bektorea 0-z hasieratu
4. *ordainduBehar* bektorea 0-z hasieratu
5. Irakurri lehenengo (*b,o*) bezero ordainleku bikotea
6. Bezeroa eta ordainlekua negatiboak ez diren bitartean:
   a. Irakurritako bidaia erregistratu
   b. Bezeroari irakurritako bidaia gehitu
   c. Une horretaraino bezeroaren datuen arabera, bezeroak ordaindu behar duen ordainsaria kalkulatu eta metatu.
   d. (*b,o*) bikotea irakurri
7. *ordainlekuZeharkatzeKop* bektorea 0-z hasieratu
8. Ordainleku bakoitzeko:
   a. Ohiko bezeroek zeharkatu duten aldi kopurua kalkulatu eta metatu.
9. *ordainlekuZeharkatzeKop* bektoreko balio maximoa finkatu, jakiteko zein den ohiko bezeroek maizen zeharkatu duten ordainlekua.
10. Bezero bakoitzeko:
    a. Erabaki zein ordainleku den maizen zeharkatu duena.
11. *berezkoOrdainketa* bektorea 0-z hasieratu.
12. Bezero bakoitzeko:
    a. Erabaki zenbat ordaindu behar izango zukeen deskonturik gabe.
13. Bezero bakoitzeko
    a. Erabaki zenbat aurreztu duen
14. Bezero bakoitzak aurreztu duen kopurua erakutsi
15. Bezero bakoitzeko:
    a. Eginiko bidaiak erakutsi
    b. Ordaindu behar duena erakutsi

c. Aurreztu duena erakutsi

C KODEKETA

```
// Autopista;
#include <stdio.h>
#include <stdlib.h>
#define OrdainlekuKop 15
#define BezeroKop 1000
main () {
int bidaiak[BezeroKop][OrdainlekuKop];
int bidaiTotalak[BezeroKop];
int ordainlekuZeharkatzeKop[OrdainlekuKop];
float bidaiSariak[OrdainlekuKop], ordainduBehar[BezeroKop];
float berezkoOrdainketa[BezeroKop], aurrezpenak[BezeroKop];
int be, ol, oLeku;
float  maxBal;

// bidaiak matrizea 0-z hasieratu
for (be=0; be<BezeroKop; be ++)
  for (ol=0; ol<OrdainlekuKop; ol++)
    bidaiak[be][ol]=0;

// Ordainleku bakoitzeko prezioak irakurri
for (ol=0; ol<OrdainlekuKop; ol++) {
  printf("%d ordainlekuko bidesaria sartu: ", ol);
  scanf("%f", &bidaiSariak[ol]);
};
// Datuen sarrera
// bidaiTotalak bektorearen hasieratzea
// ordainduBehar bektorearen hasieraketa
for (be=0; be<BezeroKop; be++) {
  bidaiTotalak[be]=0;
  ordainduBehar[be]=0;
}
// Bezero bakoitzak osatu dituen tramuen
// irakurketa eta erregistroa
printf("Kodea eta ordainleku kodea (amaitzeko -1 -1):\n");
printf("Bezeroa, ordainlekua: ");
```

```
scanf("%d, %d", &be, &ol);
// negatiboa bada baten bat, amaitutzat emango da sarrera
while ((be>=0) && (ol>=0)) {
  bidaiak[be][ol]= bidaiak[be][ol]+1;
  bidaiTotalak[be]= bidaiTotalak[be]+1;
  if (bidaiTotalak[be]<=8) {
    ordainduBehar[be]= ordainduBehar[be]+
                       bidaiSariak[ol]*(1-0.25);}
  else if (bidaiTotalak[be]<=20) {
    ordainduBehar[be]= ordainduBehar[be]+
                       bidaiSariak[ol]*(1-0.45); }
  else {   // 20 bidai baino gehiago egin ditu
    ordainduBehar[be]= ordainduBehar[be]+
                       bidaiSariak[ol]*(1-0.75);}
  printf("Bezeroa, ordainlekua: ");
  scanf("%d, %d", &be, &ol);
}

// Ordainleku bisitatuenaren kalkulua
for (ol=0; ol<OrdainlekuKop; ol++) {
  ordainlekuZeharkatzeKop[ol]=0;    // hasieratzea
  // Ordainleku bakoitza zeharkatu den aldi kopuruaren kalkulua
  for (be=0; be<BezeroKop; be++)
    ordainlekuZeharkatzeKop[ol] = ordainlekuZeharkatzeKop[ol]
                                  + bidaiak[be][ol];
}

// Bezeroek gehien zeharkatu duten ordainlekua
maxBal=ordainlekuZeharkatzeKop[0];
oLeku=0;
for (ol=1; ol<OrdainlekuKop; ol++)
  if (ordainlekuZeharkatzeKop[ol]> maxBal) {
    maxBal =ordainlekuZeharkatzeKop[ol];
    oLeku =ol;
  };
printf("Ordainleku zeharkatuena %d da\n.", oLeku);
```

```
// bezero bakoitzak gehien zeharkatzen duen ordainlekua
for (be=0; be<BezeroKop; be++) {
  maxBal=bidaiak[be][0];
  oLeku=0;
  for (ol=1; ol<OrdainlekuKop; ol++)
    if (bidaiak[be][ol]>maxBal)
      { maxBal= bidaiak[be][ol]; oLeku=ol; }
  printf("%d bezeroak gehien erabiltzen duen ordainlekua "
         "%d da\n", be, oLeku);
}

// ohiko bezero bakoitzaren aurrezpena hil honetan
for (be=0; be<BezeroKop; be++) {
  berezkoOrdainketa[be]=0;          // hasieraketa
  // berez, ordaindu behar zukeena
  for (be=0; be<BezeroKop; be++)
    for (ol=0; ol<OrdainlekuKop; ol++)
      berezkoOrdainketa[be]=berezkoOrdainketa[be] +
                            bidaiSariak[ol]*bidaiak[be][ol];
}
// ordaindu dutena kentzen diegu
for (be=0; be<BezeroKop; be++) {
  aurrezpenak[be]= berezkoOrdainketa[be] - ordainduBehar[be];
  // bezero bakoitzak aurreztu duena idazten da
  printf("%d bezeroak %5.2f euro aurreztu du.\n", be,
          aurrezpenak[be]);
}
// Emaitzak idatzi
printf("Bezero kod.  Bidai tot.  Ordaindutakoa  Aurrezpena\n");
for (be=0; be<BezeroKop; be++) {
  printf("\t%d   \t%d    \t%5.2f   \t%5.2f\n ", be,
         bidaiTotalak[be], ordainduBehar[be], aurrezpenak[be]);
}
  printf("\n\n");
  system("PAUSE");
}
```

JOKO ARETOA

Joko areto batek programa informatiko bat egitea eskatu digu, zeinak momentu bakoitzean aretoan dagoen jokalari bakoitzaren egoera ezagutzea ahalbideratzen digun.

Programa egin ahal izateko joko aretotik egunean gehienez 1000 pertsona pasako direla eta aretoan 80 joko daudela esan digute.

Programa hau egunero exekutatuko da. Programak egin behar duen lehenengo gauza erabiltzaileari joko bakoitzean lortutako puntu bakoitzarengatik irabazten den euro kantitatea eskatzea da. Datu hauek *euroJokoak* izeneko array batean gordeko ditugu. Hau egin ondoren, erabiltzaileari menu bat agertuko zaio pantailan non jokalari bakoitzak joko bakoitzean lortutako puntuak sartzeko aukera emango dion bai eta ondoren zehaztuko diren hainbat kalkulu egiteko aukera ere. Jakin, jokoak 0tik 79ra zenbakituak daudela eta jokalari bakoitzari aretoan sartzerakoan 0tik 999 bitarteko zenbaki bat esleituko diotela.

EuroJoko arrayaren irakurketaren adibidea (erabiltzaileak sakatutako datuak beltzez agertzen dira):

```
Joko bakoitzean lortutako puntu irabazia eman:

   0 jokoa: 6
   1 jokoa: 1
   2 jokoa: 4
   3 jokoa: 6
   ...
   29 jokoa: 15
   ...
   79 jokoa: 2
```

Adibidea:

	0	*1*	*2*	*3*	.....	*29*	.....	*79*
euroJokoak	6	1	4	6	.....	15	.....	2

Datu hauen irakurketa amaituta pantailan ondoko menua azaldu beharko da. Menuko aukera bat sakatu eta bere emaitza lortu ondoren, programak berriro ere menua aurkeztu beharko digu pantailan, 5. aukera sakatu ezean.

Adibidea:

```
Joko aretoko menua:

   1.  Jokalari baten puntuak sartu
   2.  Joko batean jokalari batek irabazita daramatzan puntuak
   3.  Jokalari batentzat parte hartu duen joko bakoitzean
       lortutako puntuak jakitea
   4.  Jokalari batek irabazitako euro kopurua
   5.  Bukatu
   Eman zure aukera:
```

Aukera bakoitza sakatzean jarraian datorrena egin beharko da:

1. Jokalari baten puntuak sartu.

   Programak jokalari zenbakia eskatuko du, eta ondoren jolastutako joko bakoitzeko lortutako puntuak sartuko dira. Datu sarrera egitean, kontuan izan, jokalari batek behin baino gehiagotan joka dezakeela joko berbera, beraz lortutako puntuak batu behar dira. Datu amaiera -1 -1 bikoteak adieraziko du. Adibidean ikusten den moduan, posible da jokalari batek 0 puntu lortzea joko batean, beraz egoera hau eta joko bat ez jokatu izana desberdindu beharko dira.

   Egunaren edozein momentutan jokalari bati datu gehiago sartzeko aukera eman behar du programak.

   Exekuzio adibidea*:*

```
Jokalari zenbakia: 10
     Jokoa eta puntuak:  4   35
     Jokoa eta puntuak:  1   0
     Jokoa eta puntuak:  4   10
     Jokoa eta puntuak:  30  10
     Jokoa eta puntuak: -1   -1
```

2. Joko batean jokalari batek irabazita daramatzan puntuak.

   Exekuzio adibidea*:*

```
Jokalari zenbakia:   10
Joko zenbakia:   4

Joko horretan lortutako puntu kopurua:   45
```

3. Jokalari batentzat parte hartu duen joko bakoitzean lortutako puntuak jakitea.

   Exekuzio adibidea*:*

```
Jokalari zenbakia:   10
       0 jokoa        Puntuak:   5
       4 jokoa        Puntuak:   45
       29 jokoa       Puntuak:   10
```

4. Jokalari batek irabazitako euro kopurua.

   Egunaren hasieran bete den *EuroJoko* arraya erabilikoa da kalkulatzeko.

   Exekuzio adibidea*:*

```
Jokalari zenbakia:   10
   Irabazitako euro kopurua: 420
```

5. Bukatu.

   Erabiltzaileak aukera hau egunaren amaieran hautatuko du eta programa amaitzeaz gain datu hauek ere kalkulatu behar ditu: (a) joko aretoak

egun horretan banatutako euro kopurua eta (b) zein jokok entregatu duen euro gehien.

PROBLEMAREN ANALISIA

Enuntziatuko konstanteak

1. *JokalariKop*= 1000. Egunean zehar joko aretotik pasa litekeen jokalari kopuru maximoa.
2. *Jokoak* = 80. Aretoan dauden jokoak.

Sarrerarako beharrezkoak diren aldagaien adierazpidea

3. *JokalariKop* x *Jokoak* posizio dituen matrizea behar da. Bertan jokalari bakoitzak parte hartu duen joko bakoitzean lortu dituen puntuak metatuko dira.

puntuazioa	*0*	*1*	*...*	*Jokoak-1*
0				
1				
...				
JokalariKop-1				

4. *Jokoak* posizio dituen *euroJokoak* bektorea behar da, joko bakoitzeko puntu bakoitzak zenbateko euro kopuruaren irabazi adierazten duen metatzeko.

	0	*1*	*...*	*Jokoak-1*
euroJokoak				

Emaitzetarako beharrezkoak diren aldagaien adierazpidea

5. *Jokoak* posizio dituen *jokoakEmandakoa* bektorea behar da, aretoak joko bakoitzeko eman duen euro kopurua egunaren amaieran azaltzeko.

	0	*1*	*...*	*Jokoak-1*
jokoakEmandakoa				

Problema ebazteko jarraibideak

1. *euroJokoak* bektorea bete.
2. *puntuazioa* matrizea –1 balioaz hasieratu. Balio honi esker jolastutako baina 0 balioa lortutako jokoak eta jolastu ez diren haiek desberdindu ahal izango dira.
3. *do-**while*** begizta erabili programak pantailan 5 aukera dituen menua aurkezteko: 1-4 tarteko aukera bat egikaritu ostean, pantailan berriro menua aurkeztuko da, eta prozesu hau behin eta berriro errepikatuko da 5 aukera (Bukatu) sakatu ezean.
4. Erabiltzaileak hautatuko aukera egikaritzeko, ***switch*** agindua erabili:

   1 aukera. Jokalari baten puntuak sartu.

   a. Jokalariaren identifikazio zenbakia irakurri.

b. Joko bat irakurri eta puntuak.

c. Jokoak edota puntuak -1 ez diren bitartean egin:

i. Joko horretan dagoeneko parte hartu badu *puntuazioa* matrizean metatu, bestela puntuak zuzenean *puntuazioa* matrizean sartu.

ii. Jokalari horren hurrengo jokoa eta puntuak irakurri.

2 aukera. Joko batean jokalari batek irabazita daramatzan puntuak.

a. Jokalari zenbakia eta jokoa irakurri.

b. *puntuazioa* matrizeko jokalari joko koordenatuek adierazten duten posizioan -1 balioa ez badago, orduan jokalariak dituen puntuak azaldu; bestela, joko horretan oraindik ez duela parte hartu adierazi.

3 aukera. Jokalari batentzat parte hartu duen joko bakoitzean lortutako puntuak jakitea.

a. Jokalari zenbakia irakurri.

b. Joko bakoitzarentzat:

i. Jokalariaren puntuazioa joko horretan –1 ez bada (jolastu du), orduan puntuazioa idatzi.

4 aukera. Jokalari batek irabazitako euro kopurua.

a. Jokalari zenbakia irakurri.

b. Joko bakoitzarentzat:

i. Puntuazioa 0 baino handiagoa bada, orduan jokoan lortutako puntuak *euroJokoak* bektoreko euroekin biderkatu eta *irabazia* aldagaian metatu.

c. Guztira irabazi duena pantailaratu.

5. Menutik irteterakoan (5 aukera), aretoak joko bakoitzeko entregatu dituen euro kopurua kalkulatu bai eta egun horretan guztira entregatu duena ere.

a. Joko bakoitzeko

b. Jokalari guztiak aztertu

i. Puntuazioa 0 baino handiagoa bada, puntuazioa *euroJokoak* bektoreko puntuekin biderkatu eta emaitza *jokoakEmandakoa* bektorean metatu.

c. Jokoz aldatu aurretik *emandakoa* aldagaian joko bakoitzak eman duena metatu.

6. Diru gehien eman duen jokoa mugatu, horretarako balio maximoa jasotzen duen posizioa aurkitu behar da *jokoakEmandakoa* bektorean.

C KODEKETA

```
// joko aretoa
#include <stdio.h>
#include <stdlib.h>
#define JokalariKop 1000
#define Jokoak 80
main () {
int puntuazioa [JokalariKop][Jokoak];
float euroJokoak[Jokoak], jokoakEmandakoa[Jokoak];
int jokalari, puntuak, joko, aukera, gehienItzuliDuenJ;
char jarraitu;
float irabazia, emandakoa;

// joko bakoitzeko irabazia (puntutan) irakurri eta
// euroJokoak bektorean jaso
printf("Joko bakoitzean lortutako puntu irabazia (eurotan) "
       " eman:\n");
for (joko=0; joko<Jokoak; joko++) {
  printf("%d jokoa: ", joko); scanf("%f",&euroJokoak[joko]);
}
// puntuazioa matrizearen hasieraketa
for (jokalari=0; jokalari<JokalariKop; jokalari++)
  for (joko=0; joko<Jokoak; joko++)
    puntuazioa[jokalari][joko]=-1;

do { // pantailan menua eta bere aukerak erakutsi
  system("CLS");
  printf("Joko aretoko menua:\n");
  printf("1. Jokalari baten puntuak sartu\n");
  printf("2. Joko batean jokalari batek irabazita daramatzan"
         " puntuak\n");
  printf("3. Jokalari batentzat parte hartu duen joko "
         "\n    bakoitzean lortutako puntuak jakitea\n");
  printf("4. Jokalari batek irabazitako euro kopurua\n");
  printf("5. Bukatu\n");
  printf("Eman zure aukera\n");
```

```
scanf("%d",&aukera);
switch (aukera) {        // hautatutako aukeraren exekuzioa
case 1: // Jokalari baten puntuen erregistroa parte hartzen
        // duen joko bakoitzeko
        printf("\nJokalari zenbakia: ");
        scanf("%d", &jokalari);
        printf("\tjokoa puntuak: ");
        scanf("%d %d", &joko, &puntuak);
        // Bikoteko balio biak > -1 badira, begiztan sartu
        while ((joko>-1) && (puntuak>-1))
        { // jokoan aldez aurretik jokatu badu, akumulatu.
          // Bestela, puntuazioa zuzenean ezarri
          if (puntuazioa[jokalari][joko]>=0)
          { puntuazioa[jokalari][joko]
                  = puntuazioa[jokalari][joko]+ puntuak;  }
          else  {puntuazioa[jokalari][joko]=puntuak;}
          printf("\tjokoa puntuak: ");
          scanf("%d %d", &joko, &puntuak);
        } // while
         break;
case 2: // Joko batean jokalari batek irabazita daramatzan
        // puntuak
        printf("\nJokalari zenbakia: ");
        scanf("%d", &jokalari);
        printf("Joko zenbakia: ");
        scanf("%d", &joko);
        // jokalari joko koordenatu parean, puntuazio matrizeak
        // duen balioari erreparatzen diogu.
        // -1 bada, ez du oraindik inoiz jolastu horretan
        // bestela, jokalariak jokoan lortutako puntuak azaldu
        if (puntuazioa[jokalari][joko]!=-1)
        { printf("Joko horretan lortutako puntu kopurua: %d\n",
                  puntuazioa[jokalari][joko]); }
        else { printf("%d jokalariak ez du %d jokoan parte "
                      "hartu  oraindik\n", jokalari, joko); }
        printf("\n");
        fflush(stdin);
```

```
        printf("Jarraitzeko teklaren bat sakatu ... \n");
        scanf("%c", &jarraitu);
        break;
case 3: // Jokalari batentzat parte hartu duen joko bakoitzean
        // lortutako puntuak jakitea
        printf("\nJokalari zenbakia: ");
        scanf("%d", &jokalari);
        // Jokalariarentzat puntuazio matrizeko joko guztiak
        // aztertuko dira eta puntuazioa erakutsiko da,
        // jolastu duenean (ez bada -1)
        for (joko=0; joko<Jokoak; joko++)
        {  if (puntuazioa[jokalari][joko]!=-1)
           { printf("\t %d jokoa \t Puntuak: %d \n",
                     joko, puntuazioa[jokalari][joko]);}
        }  // for
        printf("\n");
        fflush(stdin);
        printf("Jarraitzeko teklaren bat sakatu ... \n");
        scanf("%c", &jarraitu);
        break;
case 4: // Jokalari batek irabazitako euro kopurua
        printf("Jokalari zenbakia: ");
        scanf("%d", &jokalari);
        irabazia=0;
        for (joko=0; joko<Jokoak; joko++)
        { if (puntuazioa[jokalari][joko]>0)
          {irabazia=irabazia+ puntuazioa[jokalari][joko]*
                               euroJokoak[joko];}
        } // for
        printf("\t Irabazitako euro kopurua: %5.2f \n",
                irabazia);
        printf("\n");
        fflush(stdin);
        printf("Jarraitzeko teklaren bat sakatu ... \n");
        scanf("%c", &jarraitu);
        break;
case 5: break;
```

```
  default: printf("Aukera okerra!!!\n");
           printf("\n");
           fflush(stdin);
           printf("Jarraitzeko teklaren bat sakatu ... \n");
           scanf("%c", &jarraitu);
  } // end switch

} while(aukera!=5);

// eguna amaitzean menutik irteten gara.
// joko aretoak egun horretan banatutako euro kopurua kalkulatu
// bai eta zein jokok entregatu duen euro gehien ere
emandakoa=0;
for (joko=0; joko<Jokoak; joko++) {
  jokoakEmandakoa[joko]=0;
  for (jokalari=0; jokalari<JokalariKop; jokalari++) {
    if (puntuazioa[jokalari][joko]>0)
      { jokoakEmandakoa[joko]= jokoakEmandakoa[joko]+
           puntuazioa[jokalari][joko]*euroJokoak[joko]; }
  } // barneko for-ra
   emandakoa=emandakoa+jokoakEmandakoa[joko];
} // for  nagusia
printf("\nGuztira entregatu den euro kopurua: %5.2f \n",
       emandakoa);
// diru gehien eman duen jokoaren kalkulua
gehienItzuliDuenJ=0;
for (joko=0; joko<Jokoak; joko++) {
  if (jokoakEmandakoa[gehienItzuliDuenJ]<jokoakEmandakoa[joko])
    gehienItzuliDuenJ=joko;
 } // for
 printf("\n %d jokoa da euro gehien eman duen jokoa, zehazki %5.2f"
        " euro entregatu dituelarik \n", gehienItzuliDuenJ,
        jokoakEmandakoa[gehienItzuliDuenJ]);
  printf("\n\n");
  system("PAUSE");
}
```

FAROLAK

Hiriko auzo bateko farolen kudeaketa egiten duen programa bat egitea eskatu digute udaletxetik. Hiriko auzo honetan bost kale daude eta kale bakoitzean 15 farol daude 0tik 14ra zenbakituak.

Aurrezteko asmoz, udaletxeak kaleko argi guztiak gauero ez piztea erabaki du. Horren ordez kale bakoitzean aldi berean gehienez zenbat farol egon daitezkeen piztuak erabaki du eta programa hasieran maximo hauek emango dizkigute *KalekoMaximoa* bektorean gordeko ditugularik.

Eskatzen diguten programak, ondoko 7 ataza hauek egin behar ditu:

1. Kale bakoitzean aldi berean piztuta egon daitezkeen farol kopurua irakurri. Adibidea: (beltzez erabiltzaileak sakatutakoak)

```
0 kalean gehienez piztu litekeen farol kopurua sakatu: 2
1 kalean gehienez piztu litekeen farol kopurua sakatu: 3
2 kalean gehienez piztu litekeen farol kopurua sakatu: 5
...
4 kalean gehienez piztu litekeen farol kopurua sakatu: 15
```

2. Farolak piztu. Eskatu zeintzuk farola piztu nahi diren adibidean azaltzen den moduan; kaleko maximoa gaindituko balu abisua eman behar du eta ez piztu farola. Adibidea:

```
Kalea eta farola sakatu (-1 -1 amaitzeko): 0  4
Kalea eta farola sakatu (-1 -1 amaitzeko): 0  3
Kalea eta farola sakatu (-1 -1 amaitzeko): 0  14
   Errorea!!! 0 kalean ezin dira 2 baino farola gehiago piztu!
Kalea eta farola sakatu (-1 -1 amaitzeko): 2  4
Kalea eta farola sakatu (-1 -1 amaitzeko): 4  11
Kalea eta farola sakatu (-1 -1 amaitzeko): -1 -1
```

3. Kale bakoitzean piztuta dauden farol kopurua lortu eta baita aldi berean piztuta egon daitekeen farol kopuru maximoa ere. Adibidea:

```
0 kalean 2 farola piztu dira eta topea 2 da
1 kalean 0 farola piztu dira eta topea 3 da
2 kalean 1 farola piztu dira eta topea 2 da
...

4 kalean 1 farola piztu dira eta topea 15 da
```

4. Kale bakoitzarentzako erabaki ezazu zer den seguruagoa eskuin aldetik ala ezker aldetik joatea. Horretarako, farola gehien piztuak duen aldea da seguruena. Farola kopuru bera piztua baldin badago bi aldeetan edo denak itzalita baldin badaude, bi aldeak segurtasun bera dutela esango dugu. Suposa ezazu eskuin aldekoak farola bikoitiak direla eta ezker aldekoak bakoitiak. Adibidea:

```
0 kalean ezkerretik ala eskuinetik joatea berdin da
1 kalean ezkerretik ala eskuinetik joatea berdin da
```

```
2 kalean eskuinetik joatea seguruagoa da
...
```

5. Farol guztiak itzalita dauzkaten kale kopurua lortu. Adibidea

```
Farola guztiak itzalita dituzten kale kopurua 2 da
```

6. Kaleak ondorengo irizpideen arabera sailkatu:

    - Farol guztiak itzalita: **'Ilunpetan'**
    - Kaleko maximoak adierazten duen kopurua bezainbeste farol piztuta baldin badago eta farol gehiago piztuta itzalita baino egon ezkero: **'Oso argitsua'**
    - Kaleko maximoak adierazten duen kopurua bezainbeste farol piztuta baldin badago baina farol gehiago itzalita piztuta baino egon ezkero: **'Argitsua'**
    - Beste edozein kasutan: **'Itzaltsua'**

    Adibidea:

```
0 kalea argitsua da.
1 kalea ilunpetan dago.
2 kalea itzaltsua da.
...
```

7. Farol guztiak itzali.

PROBLEMAREN ANALISIA

Enuntziatuko konstanteak

1. *KaleKop* = 5. Aztergai dagoen kale kopurua.
2. *FarolaKop* =15. Kale bakoitzeko dagoen farola kopurua

Sarrerarako beharrezkoak diren aldagaien adierazpidea

3. *farolakNon* matrizea beharko dugu, *KaleKop* x *FarolaKop* dimentsioak izango ditu. Bertan 0 bat egoteak farola itzalita dagoela adieraziko du eta 1 badago, aldiz, farola piztuta dagoela.

farolakNon	0	1	...	FarolaKop-1
0				
1				
...				
KaleKop-1				

4. *maxPiztuak* bektorea beharko dugu, *KaleKop* gelaxka izango dituena. Bertan, eta kale bakoitzean, aldi berean kale horretan pizturik dauden farola kopurua jasoko du.

	0	1	...	KaleKop-1
maxPiztuak				

Emaitzetarako beharrezkoak diren aldagaien adierazpidea

5. *piztuak* bektorea beharko dugu, baita ere *KaleKop* gelaxka dituena. Bertan, farolak pizten goazen heinean inkrementatzen joango gara jasotako balioa eta horrela kalean aldi berean pizturik egon litezkeen topea ez gainditzeko balioko digu.

	0	1	...	KaleKop-1
piztuak				

Problema ebazteko jarraibideak

1. *maxPiztuak* bektorea bete.
2. *farolakNon* matrizea hasieratu 0-ra (farola guztiak itzalita).
3. Kalean piztuak dauden farolen kopurua bektorea 0-z bete (*piztuak*).
4. Farolen pizketa. Kalea eta farola eskatu
5. Parea -1 balioren desberdina den bitartean,
   a. Kale horretan piztuak dauden farolen kopurua kaleko topea baino baxuagoa bada, orduan farola piztu (1-a matrizean islatu) eta kale horretan beste farola bat gehiago piztu dela jaso; bestela ezin dela piztu mezu errorea azaldu.
   b. Hurrengo kalea farola bikotea irakurri.

6. Kale bakoitzeko piztuak dauden farola kopurua erakutsi bai eta gehienez piztu litezkeen topea.
7. Kaleetako segurtasuna. Kale bakoitzeko egin:
    a. Ezkerreko eta eskuineko farolen kontagailua 0-ra jarri.
    b. Kaleko farola bakoitzeko egin:
        i. Farola zenbakia bakoitia bada eta farola piztua badago, ezkerreko farolen kontagailua inkrementatu; bestela, farola zenbakia bikoitia bada eta farola piztua badago, eskuineko farola kontagailua inkrementatu behar da.
    c. Kaleko farola guztiak aztertu ostean, kontagailuak konparatu eta kalea nolakoa den adierazi.
8. Farola guztiak itzalita dituzten kale kopurua. *denak* kontagailua 0-ra jarri. Kale bakoitzeko egin:
    a. Kale horretako *piztuak* bektoreko balioa 0 bada, orduan *denak* kontagailua inkrementatu.
9. Kaleen klasifikazioa. Kale bakoitzeko egin:
    a. Piztuta dauden farola kopura kaleko maximoaren berdina bada eta piztuta dauden farol kopurua itzalita daudenak baino handiagoa bada, kalea oso argitsua dela adierazi.
    b. Bestela, piztuta dauden farola kopura kaleko maximoaren berdina bada baina piztuta dauden farol kopurua itzalita daudenak baino txikiagoa bada, kalea argitsua dela adierazi.
    c. Bestela, denak itzalita badaude, kalea ilunpetan dagoela adierazi.
    d. Eta bestela, kalea beldurgarria dela esan.
10. Farolak itzali, hau da, *farolakNon* matrizea 0-ra jarri.

C KODEKETA

```
#include <stdio.h>
#include <stdlib.h>
#define KaleKop 5
#define FarolaKop 15
main (){
int  farolakNon[KaleKop][FarolaKop], maxPiztuak[KaleKop];
int  piztuak [KaleKop], fa,ka,ezker,eskuin,denak;

// 1.- Kale bakoitzean piztu litezken farol kopuru maximoa
for (ka=0; ka<KaleKop; ka++) {
  printf("%d kalean gehienez piztu litekeen farol "
         "kopurua sakatu: ",ka);
  scanf("%d", &maxPiztuak[ka]);
}
// 2.- Hasieran farolak itzalita daude: Matrizea 0-ra
for (ka=0; ka<KaleKop; ka++)
  for (fa=0; fa<FarolaKop; fa++) farolakNon[ka][fa]=0;

// Kale bakoitzeko farolen kontagailua 0-ra ezartzen dugu
 for (ka=0; ka<KaleKop; ka++)   piztuak[ka]=0;

// piztuak bektorean kontrolatuko ditugu dagoeneko zenbat
// farol dauden pizturik, topea ez gainditzeko
printf(" Kalea eta farola sakatu (-1 -1 amaitzeko) ");
scanf("%d %d", &ka,&fa);
while (not(ka==-1 && fa==-1)) {  // enuntziatuak ezarria
  if (piztuak[ka]<maxPiztuak[ka]) {
    { farolakNon[ka][fa]=1;  // farola pizten dugu
       piztuak[ka]++; }
  else printf("  Errorea!!! %d kalean ezin dira %d baino farola "
              " gehiago piztu!\n\n", ka, maxPiztuak[ka]);
  printf(" Kalea eta farola sakatu (-1 -1 amaitzeko) ");
  scanf("%d %d", &ka,&fa);
}
```

```
// 3.- Kale bakoitzean zenbat farola piztu diren eta topea
for (ka=0; ka<KaleKop; ka++) {
  printf("\n%d kalean %d farola piztu dira eta topea "
         " %d da", ka, piztuak[ka], maxPiztuak[ka]);
}

// 4. Segurtasuna kaleetan
for (ka=0; ka<KaleKop; ka++) {
  ezker=0;  eskuin=0;
  for (fa=0; fa<FarolaKop; fa++) {
    if (fa%2!=0 && farolakNon[ka][fa]==1) {ezker++;}
    else if (fa%2==0 && farolakNon[ka][fa]==1) {eskuin++;}
  }
  if (ezker==eskuin)
    { printf("\n%d kalean ezkerretik ala eskuinetik joatea "
             "berdin da", ka); }
  else if (ezker > eskuin)
    {printf("\n%d kalean ezkerretik joatea seguruagoa da",ka);}
  else {printf("\n%d kalean eskuinetik joatea seguruagoa"
               " da",ka);}
} // for  nagusia

// 5.- Farola guztiak itzalita dituzten kale kopurua
denak=0;
for (ka=0;ka<KaleKop;ka++) {if (piztuak[ka]==0) {denak++;}}
printf( "\nFarola guztiak itzalita dituzten kale kopurua"
        " %d da", denak);

// 6- Kaleen klasifikazioa
for (ka=0; ka<KaleKop; ka++) {
  if (piztuak[ka]==maxPiztuak[ka] &&
      piztuak[ka]>FarolaKop-piztuak[ka])
    { printf("\n%d kalea oso argitsua da", ka);}
  else if (piztuak[ka]==maxPiztuak[ka] &&
           piztuak[ka]<FarolaKop-piztuak[ka])
    { printf("\n%d kalea argitsua da", ka);}
```

```
  else if (piztuak[ka]==0)
    { printf("\n%d kalea ilunpetan dago", ka);}
  else printf("\n%d kalea beldurgarria da", ka);
}

// 7.- Farolak itzali
for (ka=0; ka<KaleKop; ka++)
  for (fa=0; fa<FarolaKop; fa++)
    farolakNon[ka][fa]=0;

  printf("\n\n");
  system("PAUSE");
}
```

HERRIEN ARTEKO DISTANTZIAK

Gipuzkoako foru Aldundiak, Gipuzkoako herrien azterketa bat egin nahi du. Azterketa honetan herri batetik besterako distantziak emanda hainbat galdera erantzuten dituen programa egitea eskatu digute.

Foru Aldundiak esan digu Gipuzkoan 85 udalerri daudela eta udalerri bakoitzetik beste udalerrira dauden kilometroak emango dizkigutela.

Hasieran, programak lehen herritik gainontzeko herri guztietara zein distantzia dagoen eskatuko du. Kontuan izan, datuak ematerako garaian bikote posible bakoitza behin bakarrik eskatuko dela eta ez dela herri batetik herri berdinerako distantzia eskatuko. Hau da, programak ez duela honelako informaziorik eskatuko:

- 1 herritik 2. herrirako distantzia eskatu badu, ez duela ondoren 2. herritik 1 herrirako daturik eskatuko
- 1 herritik 1 herrirako distantzia ez duela ere eskatuko.

Murriztapen hauek kontuan izanda, datu sarrera honelakoa izango da (letra lodiz erabiltzaileak emandako datuak daude):

Adibidea: (beltzez erabiltzaileak sakatutako datuak)

```
Gipuzkoako herrien arteko distantziak irakurtzen:
Eman 0 herritik ondoko herrietarako distantziak (km-tan):
    1 herria: 25
    2 herria: 68
    ....
    84 herrira: 12.3

Eman 1 herritik ondoko herrietarako distantziak (km-tan):
    2 herria: 105.5
    ....
    84 herrira: 26.5

    ....
```

Behin datu hauek eskatuta pantailan ondoko menua azaldu beharko da. Menuko aukera bat sakatu eta bere emaitza lortu ondoren, programak berriro ere menua aurkeztu beharko du pantailan, 6. aukera sakatu ezean. Adibidea:

```
Herrien azterketarako menua:
    1. Herrien arteko distantzien taula ikusi.
    2. Urrutien dauden bi herriak eman.
    3. Herri batetik urrutien dagoen herria eman.
    4. Herririk zentrikoena eman.
    5. Postariak egin beharreko kilometro kopurua kalkulatu.
    6. Bukatu
    Eman zure aukera:
```

Aukera bakoitza sakatzean jarrian datorrena egin beharko da:

1. Herrien arteko distantzien taula ikusi. **Adibidea:**

	0	1	2	3	...	83	84
0	—	*25*	*68*			*134*	*12.3*
1	*25*	—	*105.5*				*26.5*
2	*68*	*105.5*	—				
3				—			
...					—		
83	*134*					—	
84	*12.3*	*26.5*					—

2. Urrutien dauden bi herriak eman. **Adibidea**:

```
Urrutien dauden bi herriak 0 eta 83 dira.
```

3. Herri batetik urrutien dagoen herria eman. **Adibidea**:

```
Eman herri zenbakia: 18
18. herritik urrutien dagoen herria 7.na da.
```

4. Herririk zentrikoena eman. Hau da, gainontzeko herri guztietara joateko, batez beste, nondik egingo diren kilometro gutxien. **Adibidea**:

```
Herririk zentrikoena 27. herria da.
```

5. Postariak egin beharreko kilometro kopurua kalkulatu. Horretarako, lehenik postaria non aurkitzen den esango zaigu eta ondoren -1 zenbakiaz amaitzen den herri sekuentzia bat emango digute. Postariak emandako herriak emandako ordenan bisitatu behar dituela suposatuz kalkulatu zenbat kilometro egin behar dituen. **Adibidea**

```
Eman postaria zein herritan dagoen: 25
Eman postaria pasatu beharreko herriak: 12 34 2 11 -1
Postariak egin beharreko kilometro kopurua 356 da.
```

PROBLEMAREN ANALISIA

Enuntziatuko konstanteak

1. *UdalerriKop* = 85. Gipuzkoan, ariketaren arabera, dagoen herri kopurua.

Sarrerarako beharrezkoak diren aldagaien adierazpidea

2. *distantzia* deituko dugun matrize bat beharko dugu, *UdalerriKop* x *UdalerriKop* posizio dituena, non herrien artean distantziak, kilometrotan, gordeko diren.

distantzia	*0*	*1*	…	*UdalerriKop-1*
0				
1				
…				
UdalerriKop-1				

Emaitzetarako beharrezkoak diren aldagaien adierazpidea

3. *kmKopuru* deituko dugun bektorea beharko dugu, *UdalerriKop* posizio dituena, non herri batetik gainontzekoetara joateko egin beharreko kilometroen batura gordeko den. Horrela, ondoren herririk zentrikoena zein den lortzeko.

	0	*1*	…	*UdalerriKop-1*
kmKopuru				

Problema ebazteko jarraibideak

1. *distantzia* matrizeko diagonala 0 zenbakiaz hasieratu. Herri batetik herri horretara joateko distantzia zero delako.

2. *distantzia* matrizea datuz bete. Horretarako erabiltzaileari herri batetik gainontzekoetara dagoen distantzia eskatu beharko zaio, baina lehenagotik emandakoak berriro eskatu gabe. Hau da, 1 herritik 2. herrira dagoen distantzia eskatu bazaio erabiltzaileari, ondoren ez diogu eskatuko 2. herririk 1. herrira dagoen distantzia. Horregatik, 1. herritik 2. herrira dagoen distantzia irakurri ondoren *distantzia[1,2]* eta *distantzia[2,1]* posizioetan gorde beharko dugu. 1. herritik 3. herrira dagoena irakurri ondoren *distantzia[1,3]* eta *distantzia[3,1]* posizioetan gorde beharko da.

3. Erabiltzaileak sakatutako aukera 0ren desberdina den bitartean, egin:

   a. Menua pantailan idaztea eta erabiltzailearen aukera irakurtzea (*aukera*)

   b. Erabiltzailearen aukera 1 bada, herrien arteko distantzien taula pantailaratu behar da.

Distantzia matrizeko lerro guztietarako, zutabe guztiak begiratu. Lerroa eta zutabea berdinak direnean gidoi bat idatzi beharko da eta bestela matrizeko balioa idatzi pantailan.

c. Erabiltzailearen aukera 2 bada, beraien artean urrutien dauden bi herriak lortu.

   Matrizeko balio maximoaren posizioa, hau da lerroa eta zutabea, lortu. Lerroa eta zutabea adierazten duten indizeek urrutien dauden bi herriak adieraziko dituzte.

d. Erabiltzailearen aukera 3 bada, erabiltzaileak emandako herritik urrutien dagoen herria kalkulatu.

   i. Jatorrizko herria eskatu erabiltzaileari (*JH*)

   ii. Bilatu, *JH* lerroan edo zutabean, maximoaren posizioa. Honela urrutien dagoen herria lortuko dugularik.

e. Erabiltzailearen aukera 4 bada, herririk zentrikoena aurkitu.

   i. *kmKopuru* bektorea datuz bate. Herri bakoitzerako, gainontzeko herrietara joateko kilometroak batuz lortuko delarik posizio bakoitzerako balioa.

   ii. *kmKopuru* bektoreko minimoaren posizioa lortu, herririk zentrikoena gainontzeko herrietara joateko kilometro gutxien egin behar direnetik izango baita.

f. Erabiltzailearen aukera 5 bada, postariak egiten dituen kilometroak kalkulatu.

   i. Postaria aurkitzen den herria irakurri (*dago*)

   ii. *kmGuztira* aldagaia 0ra hasieratu eta joan behar duen herria irakurri (*doa*).

   iii. Joan beharreko herria (*doa*) -1en desberdina den bitartean, zegoen herritik (*dago*) joan beharreko herrira (*doa*) dauden kilometroak gehitu *kmGuztira* aldagaira eta *doa* herria *dago* bihurtu eta joan beharreko herri berria irakurri.

C KODEKETA

```
#include <stdio.h>
#include <stdlib.h>
#define UdalerriKop 85
main(){
int hS, hB, h1, h2, jH, d, banaketa, aukera, zentroa;
int kmGuztira, dago, doa;
int distantzia[UdalerriKop][UdalerriKop];
int kmKopuru[UdalerriKop];

// distantzia herri batetik bertara 0 da
for (hS=0; hS<UdalerriKop; hS++) distantzia[hS][hS]=0;

//Udalerri desberdinen arteko distantzien jasotzea
for (hS=0; hS<UdalerriKop; hS++)  {
  printf("Eman %d herritik ondoko herrietarako "
         "distantziak (km-tan):\n", hS);
  for (hB=hS+1;hB<UdalerriKop;hB++) {
    printf("\t%d. herrira: ", hB);
    scanf("%d", &d);
    distantzia[hS][hB]=d;
    distantzia[hB][hS]=d;
  }
}

do {
  printf("\nMENUA\n");
  printf("1.-Udalerrien arteko distantzien "
         "erakusketa\n");
  printf("2.-Urrutien dauden bi herriak\n");
  printf("3.-Herri bat emanik, honek urrutien duena\n");
  printf("4.-Herri zentrikoena\n");
  printf("5.-Herri-ibilbide batek dituen km kopurua\n");
  printf("0.-Amaitu\n");
  printf("Sakatu aukera bat 0-5 tartean: ");
  scanf("%d",&aukera);
```

```
switch (aukera) {
case 0: printf("Programa bukatzera doa...\n");
        break;
case 1: for (hS=0; hS<UdalerriKop; hS++)
        {  printf("%d herritik ", hS);
           for (hB=0; hB<UdalerriKop; hB++)
           {  printf("%4d", distantzia[hS][hB]);}
              printf("\n");
        } //for
        break;
case 2: banaketa=0;
        for (hS=0; hS<UdalerriKop; hS++)
        {  for (hB=hS; hB<UdalerriKop;hB++)
           {  if (distantzia[hS][hB]>banaketa)
              { banaketa=distantzia[hS][hB];
                h1=hS;   h2=hB; }
            }
         }
         printf("Urrutien dauden herri bikotea %d"
                " eta %d dira.\n", h1, h2);
        break;
case 3: printf("Eman abiapuntu udalerria: ");
        scanf("%d", &jH);
        banaketa=0;
        for (hB=0; hB<UdalerriKop; hB++)
        { if (distantzia[jH][hB]>banaketa)
          { banaketa= distantzia[jH][hB];
            h2=hB; }
        }
        printf("%d herritik urrutien dago udalerria"
               " da: %d\n", jH, h2);
        break;
case 4: d=37569;
        //Oso balio handia minimoarekin geratzeko gero
        for (hS=0; hS<UdalerriKop; hS++)
```

```
          {  kmKopuru[hS]=0;
             for (hB=0; hB<UdalerriKop; hB++)
             { kmKopuru[hS]=kmKopuru[hS]+
                 distantzia[hS][hB]; }
             // ez da behar UdalerriKop-ekin
             // zatitzea kmKopuru[hS]
             if (d>kmKopuru[hS])
             { d=kmKopuru[hS]; zentroa=hS; }
          }
          printf("Udalerri zentrikoena da: %d\n",
                  zentroa);
          break;
  case 5: printf("Eman postaria zein herritan dagoen: ");
          scanf("%d",&dago);
          printf("Eman postaria pasatu beharreko herriak"
                 " (-1 amaitzeko): ");
          scanf("%d", &doa);
          kmGuztira=0;
          while (doa!=-1)
          {  kmGuztira= kmGuztira + distantzia[dago][doa];
             dago=doa;    //Pasa da hurrengo herrira, hau
                          // abiapuntu bihurtuz
             scanf("%d",&doa);   //Hurrengo jo-muga
           }
           printf("Emandako ibilbideak dituen km kopurua"
                  " da: %d\n", kmGuztira);
           break;
  default: printf("Aukera okerra sakatu duzu.\n");
  } //switch
} while (aukera!=0);
printf("\n\n");
system("PAUSE");
}
```

OLINPIADAK

Nazioarteko Batzorde Olinpiarrak programa informatiko bat eskatu digu. Programa honek, bukatu berri diren Beijingo olinpiadatako eta 2004ko Atenaseko olinpiadatako partaidetza datuekin zenbait kalkulu egin beharko ditu.

Beijing 2008ko Olinpiada, Txina

Pekingo komiteak 203 herrialdetako kirolariak 37 diziplina olinpikoetan parte hartu dutela esan digu. Egia da, herrialde guztiek ez dutela errepresentazioa izan diziplina guztietan. Ondorioz, Pekingo komiteak informazioa herrialdeka ordenatuta emango digu eta herrialde bakoitzeko parte hartu duten diziplinen informazioa emango digu bakarrik. Egin beharreko programak sarrerako datuak adibidean adierazten den bezala jaso beharko ditu.

Adibidea: Pekineko datu-sarrera: (beltzez erabiltzaileak sakatutakoak)

```
Herrialde bakoitzerako sartu kirol bakoitzean parte hartu duten
kirolariak:
Herrialdea 0:
    Kirola, partaide kopurua: 3 100
           (3 diziplina zenbakia da eta 100 partaide kopurua)
    Kirola, partaide kopurua: 6 23
    Kirola, partaide kopurua: -1 1
Herrialdea 1:
    Kirola, partaide kopurua: 1 13
    Kirola, partaide kopurua: 6 24
    Kirola, partaide kopurua: -1 0
...
Herrialdea 202:
    Kirola, partaide kopurua: 12 1
    Kirola, partaide kopurua: 34 2
    Kirola, partaide kopurua: 57 19
    Kirola, partaide kopurua: -1 3
```

Herrialde bateko datu-sarrera amaitzeko -1 bat sakatuko da diziplina zenbaki bezala, berdin izango zaigularik zein balio datorren partaide kopuruan.

Atenas 2004ko olinpiadak, Grezia

Atenaseko komiteak esan digu 2004. urtean ere 203 herrialdeek hartu zutela parte (Beijing 2008ko olinpiadetan parte hartu duten berberek). Aipatu, herrialde bakoitzetik zenbat partaide egon ziren diziplina bakoitzean erregistratuak ez duten arren herrialde bakoitzean zenbat partaide egon ziren totalean jasota dute. Bestalde posible izango dute ere herrialdeak Pekingo komiteak emandako ordena berean ematea. Gauzak honela, Atenaseko informazioa adibidean azaltzen den moduan eskatuko zaio erabiltzaileari.

Adibidea: Atenaseko datu-sarrera: (beltzez erabiltzaileak sakatutakoak)

```
Atenas 2004ko datu-sarrera:
    0 herrialdeko partaide kopurua:240
    1 herrialdeko partaide kopurua:2402
    ...
    202 herrialdeko partaide kopurua:3500
```

Behin olinpiadetako azken partaidetza datuak eskatu eta gorde ondoren, komite olinpikoak gure programak ondorengo kalkuluak egin behar dituela esan digu.

Pekin 2008ko Olinpiadei dagokionez:

1. Adierazi, mundu mailan, zein kirol diziplinak izan duen partaide gutxien eta hauek zenbat izan diren. Adibidea

   ```
   Partaide gutxien izan duen diziplina Olinpikoa 25garrena
   izan da 2 partaiderekin.
   ```

2. Erabiltzaileak emandako herrialde bakoitzarentzako, partaide gehien izan dituen kirol diziplina kalkulatuko da. Herrialde sarrera amaituko da herrialde bezala erabiltzaileak -1 zenbakia sakatzen duenean. Honez gain, emandako herrialde kodea zuzena dela konprobatuko da, hau da [0..202] tartean dagoela. Tartean ez badago errore mezua emango du programak eta hurrengo herrialdea eskatuko da. Adibidea:

   ```
   Herrialdea: 2
          Kirolari gehien dituen diziplina: 19
   Herrialdea: 302
      Herrialde kode okerra. Idatzi 0 eta 202 tarteko zenbaki
      bat edo -1 amaitzeko.
   Herrialdea: 5
      Kirolari gehien dituen diziplina: 1
   Herrialdea: -1
   ```

Atenas 2004ko olinpiadekin alderatuta:

3. Atenaseko ediziotik kirolari kopurua handitu duten herrialdeen zerrenda eman. Adibidea

   ```
   3   4   14 dira partaide kopurua handitu duten herrialdeak.
   Guztira, 3 herrialde dira.
   ```

   Beinging'2008n ez badago herrialderik bere partaidetza handitu duena Atenas 2004ekiko ondorengo mezua idatzi beharko da.

   ```
   Ez dago herrialderik bere partaidetza handitu duenik 2004ko
   edizioarekiko.
   ```

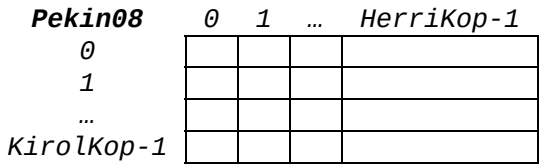
PROBLEMAREN ANALISIA

Enuntziatuko konstanteak

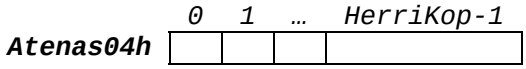
1. *KirolKop* = 37. Pekin eta Atenaseko olinpiadetan egon ziren kirol diziplina desberdinen kopurua.
2. *HerriKop* = 203. Pekinen eta Atenasen parte hartu zuren herrialdeen kopurua.

Sarrerarako beharrezkoak diren aldagaien adierazpidea

3. *Pekin08* deituko dugun matrize bat beharko dugu, *KirolKop* x *HerriKop* posizio dituena, non kirol diziplina bakoitzeko herrialde bakoitzak eraman dituen kirolari kopurua jasoko duen

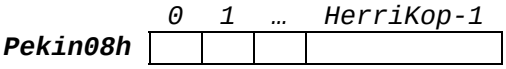


4. *Atenas04h* deituko dugun bektore bat beharko dugu *HerriKop* gelaxkekin. Bertan herrialde bakoitzak Atenasera eraman zituen kirolari kopurua jasoko dugu.

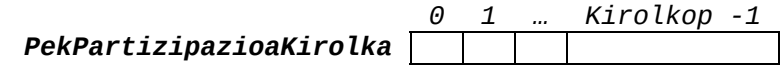


Emaitzetarako beharrezkoak diren aldagaien adierazpidea

5. *Pekin08h* deituko dugun bektore bat beharko dugu *HerriKop* gelaxkekin. Bertan, herrialde bakoitzak kirol diziplina guztiak kontuan izanik Pekinera eraman dituen kirolari kopurua jasoko dugu.



6. *PekPartizipazioaKirolka* deituko dugun bektore bat beharko dugu *KirolKop* posizioekin, non kirol bakoitzeko herrialde guztiak kontuan izanik orotara kirol horretan parte hartu duten kirolarien kopurua jasoko duen.



Problema ebazteko jarraibideak

1. *Pekin08* matrizea 0-z hasieratuko dugu, zenbait diziplinetan herrialdeek errepresentaziorik ez izatea gerta bailiteke.
2. *Pekin08* matrizea datuz bete. Horretarako erabiltzaileari banaka-banaka ordena gorakorrean herrialdeak erakutsiko zaizkio eta *he* herrialde bakoitzeko:

   a. kirol diziplina (*ki*) eta kirol honetan parte-hartzaileen (*pa*) kopurua osokoen parea eskatu

   b. Kirol diziplina(*ki*) -1 balioaren desberdina den bitartean,

      i. *Pekin08* matrizean *he* herrialdeak *ki* kirol diziplinan *pa* partaide izan dituela erregistratu.

      ii. Beste (*ki,pa*) balio parea lortu.

3. *Atenas04h* bektorea datuz bete. Horretarako erabiltzaileari banaka-banaka ordena gorakorrean herrialdeak erakutsiko zaizkio eta *he* herrialde bakoitzeko, *pa* partaide kopurua eskatu zaio *Atenas04h[he]* gelaxkan gordez.

4. *ki* kirol diziplina bakoitzeko:

   a. *batura* akumuladorea 0-z hasieratu.

   b. Herrialde bakoitzak *ki* diziplinan izan dituen partaideak *batura* aldagaian metatu.

   c. *batura* aldagaiak metatutako balioa *PekPartizipazioaKirolka* bektoreko *ki* kirolean metatu.

5. *PekPartizipazioaKirolka* bektoreko minimoa zein kirolan (posizioan) dagoen erabaki.

6. Erabiltzaileari *he* herrialde bat eskatu.

7. -1 ez den bitartean lortutako herrialdea,

   a. Herrialde kodea zuzena bada ($0 \leq he \leq 203$)

      i. *he* herrialdeak Pekinera zein diziplinetan eraman zituen kirolari gehiago erabaki. Horretarako *Pekin08* matrizeko *he* zutabeko kirol diziplina guztiak aztertu behar dira eta bertako balio maximoaren posizioa lortu; hots *Pekin08[0..36][he]* balioak.

   b. Herrialde kodea okerra bada ($he<0$ edo $203<he$)

      i. Herrialde kodea okerra dela adierazi.

   c. Beste *he* herrialde bat eskatu.

8. *he* herrialde bakoitzeko

   a. Pekinen izan dituen kirolari kopurua (*Pekin08h[he]*) Atenasera joan zirenak baino handiagoa balitz (*Pekin08h[he]*), *he* herrialdeak kirol ordezkari gehiago eraman dituela pantailaratu.

C KODEKETA

```
#include <stdio.h>
#include <stdlib.h>
#define KirolKop 37
#define HerriKop 203
main() {
int Pekin08[KirolKop][HerriKop];
int Pekin08h[HerriKop], Atenas04h[HerriKop];
int PeKPartizipazioaKirolka[KirolKop];
int he, ki, pa, partaidetzaMin, kArrakasta, batura, orora;

/* Hasieraketa derrigorrezkoa da, hainbat diziplinetan herrialdeek
   errepresentaziorik ez dutelako izango*/
for (he=0; he<HerriKop; he++)
  for (ki=0; ki<KirolKop; ki++)  Pekin08[ki][he]=0;
/* Pekingo datuen erregistroa*/
printf("Herrialde bakoitzerako sartu kirol bakoitzean parte hartu "
       "duten kirolariak:\n");
for (he=0; he<HerriKop; he++) {
  printf(" Herrialdea %d: \n", he);
  printf("   Kirola, partaide kopurua: ");
  scanf("%d %d",&ki,&pa);
  while (ki!=-1) {
    Pekin08[ki][he]=pa;
    printf("   Kirola, partaide kopurua: ");
    scanf("%d %d",&ki,&pa);
  }
}
/* Atenaseko erregistroak ez du hasieraketarik behar, herrialde
   guztiek partaideak izan baitituzte*/
printf("Atenas 2004ko datu-sarrera:\n");
for (he=0; he<HerriKop; he++) {
  printf("   %d herrialdeko partaide kopurua: ", he);
  scanf("%d",&Atenas04h[he]);
  // edo    scanf("%d",&pa); Atenas04h[he]=pa;
}
```

```
/* Pekineko kalkuluak
   Partaidetza minimoko diziplina:
       (1) Partaidetza kopurua diziplinaka;
       (2) hauen arteko minimoa*/
for (ki=0; ki<KirolKop; ki++) {
  batura=0;
  for (he=0; he<HerriKop; he++){batura=batura + Pekin08[ki][he];}
  PeKPartizipazioaKirolka[ki] = batura;
}
partaidetzaMin=0;
for (ki=1; ki<KirolKop; ki++) {
  if (PeKPartizipazioaKirolka[ki] <
      PeKPartizipazioaKirolka[partaidetzaMin])
   { partaidetzaMin=ki; }
}
printf("Partaide gutxien izan duen diziplina Olinpikoa %d-a "
       " izan da %d partaideekin.\n\n", partaidetzaMin,
       PeKPartizipazioaKirolka[partaidetzaMin]);

/* Erabiltzaileak esandako herrialdeko diziplina arrakastatsuena
partaidetza aldetik */
printf("Herrialdeko diziplina arrakastatsuena mugatzen"
       " (-1 amaitzeko)\n");
printf("Herrialdea: "); scanf("%d",&he);
while (he!=-1) {
  if ((0<=he) && (he<= HerriKop)) /*Diziplina arrakastatsuena?*/
  {  kArrakasta=0;   /*Printzipioz 0.a arrakastatsuena*/
     for (ki=1; ki<KirolKop; ki++) {
       if (Pekin08[ki][he]>Pekin08[kArrakasta][he])
           { kArrakasta=ki;}
     }
     printf("Kirolari gehien dituen diziplina: %d da\n\n",
             kArrakasta);
  }
  else /*sarrera okerra*/
```

```
  { printf("Herrialde kode okerra. Idatzi 0 eta &d tarteko "
           "zenbaki bat edo -1 amaitzeko.\n", HerriKop-1);
  }
  /* zuzena edo okerra, herrialdea aztertu ostean beste herrialde
  kode bat lortu */
 printf("Herrialdea: ");
 scanf("%d", &he);
}
printf("\n\n");

/* Pekin vs Atenas*/
/*  (1) herrialde bakoitzeko partaidetzak Pekinen kalkulatu */
for (he=0; he<HerriKop; he++) {
  batura=0;
  for (ki=0; ki<KirolKop; ki++) {batura=batura + Pekin08[ki][he];}
  Pekin08h[he]= batura;
}
/* 2. euren partaidetza handitu duten herrialdeen zerrenda
   Atenasetik-Pekinera */
orora=0;
for (he=0; he<HerriKop; he++)
  if (Pekin08h[he] > Atenas04h[he]){ printf("%5d", he); orora++; }
if (orora>0)
{ printf(" dira partaide kopurua handitu duten herrialdeak. "
          " Guztira, %d herrialde dira\n", orora);
}
else
{ printf("Ez dago herrialderik bere partaidetza handitu duenik "
         " 2004ko edizioarekiko.\n");
}
printf("\n\n");
system("PAUSE");
}
```

VENTAS S. A.

Ventas S. A. enpresak bere salmenta zonarik onena zein den aztertu nahi du. Horretarako, bere 10 saltzaileek lan egiten duten 20 zonen salmenta datuak irakurri eta analizatzeko programa informatiko bat eskatu digu.

Enpresak prezio desberdina duten 5 produktu saltzen ditu. Produktu hauen prezioa ez da aldatuko denbora luzez, horregatik datu hauek bektore konstante batean erazagutzeko aukera hobetsi da. Prezio hauekin ondoren salmenten zenbatekoa kalkulatu ahalko da. Produktuen prezioak honakoak dira:

	0	*1*	*2*	*3*	*4*
`ProdPrezioak`	20.5	50	15	5.25	150

Datu-sarreran erabiltzaileak saltzaile eta zona bakoitzeko ondorengo datuak sartuko ditu: saldu dituen produktuak eta bakoitzetik saldutako unitateak adibidean ikusten den moduan.

```
Sartu saltzaile zenbakia eta zona zenbakia: 2   4
  Sartu saldutako produktu zenbakia eta unitate kopurua: 2   10
  Sartu saldutako produktu zenbakia eta unitate kopurua: 1   5
  Sartu saldutako produktu zenbakia eta unitate kopurua: 3   20
  Sartu saldutako produktu zenbakia eta unitate kopurua: -1  -1
Sartu saltzaile zenbakia eta zona zenbakia: 1   3
  Sartu saldutako produktu zenbakia eta unitate kopurua: 5   50
  Sartu saldutako produktu zenbakia eta unitate kopurua: 3   10
  Sartu saldutako produktu zenbakia eta unitate kopurua: -1  -1
...
Sartu saltzaile zenbakia eta zona zenbakia: -1   -1
```

Datu-sarrera amaituko da saltzaile zenbaki bezala -1 bat ematen digutenean, berdin zaigularik zein balio ematen duten zona bezala. Saltzailez eta zonaz aldatuko da produktu zenbaki bezala -1 bat ematen digutenean, berdin izango zaigularik emandako unitate kopurua. Azterlan hau egiteko, saltzaile eta zona bakoitzeko interesatzen zaiguna da salmenten zenbateko totala. Kontutan izan, gerta litekeela saltzaile batek ez egitea inongo salmentarik zonaren batean.

Programak ondorengoa egin beharko du:

1. Pantailan idatzi saltzaile bakoitzak zona bakoitzean egindako salmenten zenbateko totala adibidean azaltzen den moduan. Adibidea:

```
Saltzailea / Zona0    Zona1    Zona2 ..... Zona19
    0        200.50   150.00   567.00 .....7584.00
    1        560.25   486.00   968.00 ..... 238.00
    ...
    9        458.75   158.00   445.00 ..... 868.00
```

2. Zona bat emanda, esan zein izan den zona horretako saltzailerik onena. Adibidea:

```
Eman zona: 3
3 zonako saltzailerik onena 2. saltzailea da.
```

3. Adierazi zein saltzailek zein zonatan lortu duen salmenta zenbateko handiena. Adibidea:

```
Salmenta zenbateko handiena 1 saltzaileak lortu du 19. zonan.
```

4. Adierazi zona bakoitzerako, zein saltzailek ez duen lortu salmentarik egitea eta zenbat diren totalean. Adibidea:

```
Zona 0: 3  8  9
     3 saltzaileek ez dute salmentarik egitea lortu.
Zona 1: Saltzaile guztiek salmentaren bat egin dute.

Zona 19: 3  4  5  7  8  9
     6 saltzaileek ez dute salmentarik egitea lortu.
```

PROBLEMAREN ANALISIA

Enuntziatuko konstanteak

1. *SalKop* = 10. Ventas enpresak dituen saltzaileen kopurua.
2. *Zonakop* = 20. Enpresako produktuak saltzen diren zona kopurua.
3. *ProduktuKop* = 5. Enpresak saltzen dituen produktu kopurua.

Sarrerarako beharrezkoak diren aldagaien adierazpidea

4. *ventas* deituko dugun matrize bat beharko dugu, *SalKop* x *ZonaKop* posizio dituena, non saltzaile bakoitzak zona bakoitzean guztira saldutakoaren diru kopurua jasoko duen.

ventas	0	1	…	ZonaKop-1
0				
1				
…				
SalKop-1				

5. *prodPrezioak* deituko dugun bektore bat beharko dugu enpresak saltzen dituen produktu adina posizioekin, posizio bakoitzean produktuaren prezioa metatuko da.

	0	1	…	ProduktuKop-1
prodPrezioak	20.5	50		150

Emaitzetarako beharrezkoak diren aldagaien adierazpidea

Problema ebazteko jarraibideak

1. *prodePrezioak* bektorea produktuen prezioekin hasieratu behar da
2. *ventas* matrizea 0-z hasieratuko dugu, saltzaile batek zona batean saldutako produktu guztien salmenta osoaren zenbatekoa metatu behar baita.
3. Saltzaile eta zona (*s, z*) pare bat irakurri
4. Irakurritako saltzaile kodea -1 ez den bitartean (*s!=-1*), *s* saltzaileak egin dituen produktuen salmenta *z* zonan erregistratu. Horretarako,
   a. Saldu duen produktu baten kodea, *p*, eta unitate kopurua lortu, *u*.
   b. Irakurritako produktu kodea -1 ez den bitartean (*p!=-1*),
      i. (*p,u*) salmentaren kostua, ventas matrizeko *s* saltzaileari *z* zonan koordenatu parean inkrementatu.
      ii. Beste (*p,u*) berri bat lortu; hots, hurrengo *p* produktu kode bat eta produktu horren saldutako *u* unitate kopurua.
   c. Beste (*s,z*) bikote berria lortu; hots, beste *s* saltzaile *z* zona bateko salmentak erregistratzeko.

5. Pantailaratu nahi den matrizearen burukoa idatzi enuntziatuak azaldutako moduan
6. *s* saltzaile bakoitzeko
    a. Zona guztietan egin dituen salmentak erakutsi; hots, ventas matrizeko *s* lerroko *ZonaKop* posizioak
7. Erabiltzaileari zona bat eskatu. Demagun *z* zona:
    a. *z* zona horretako saltzailerik onena zein izan den mugatu; hots, *ventas* matrizeko *z* zutabea finko mantenduz, zein lerro koordenatuak jasotzen duen balio maximoa erabaki.
8. *ventas* matrizeko posizio guztien arteko balio maximoa jasotzen duten koordenatu parea erabaki behar da: zein saltzailek zein zonatan salmenta zenbateko maximoa egin duen. Horretarako:

    Hasiera batean, saltzailerik onena 0 izan dela eta salmenta onena 0 zonan egin duela suposatuko dugu, horretarako *smax=0* eta *zmax=0* egiten dugu.

    Saltzaileka (*s*) eta honetan zonaka (*z*), adibidez, behin eta berriro egingo da:

    a. Koordenatu pare berrian balio handiagoa jasotzen bada (*smax, zmax*) koordenatuetan baino: *ventas[s][z]<ventas[smax][zmax]*
    b. *smax* eta *zmax* koordenatuak eguneratzen dira uneko aztergai diren *s* eta *z* koordenatuekin.
9. *z* zona bakoitzeko
    a. zonako *kont* kontagailua 0-ra hasieratzen da
    b. *z* zona aztergai dela pantailaratzen dugu
    c. *s* saltzaileak banaka aztertzen dira
        i. *s* saltzaileak *z* zonan salmentarik egin ez badu (*ventas[s][z]==0*), *s* saltzailearen kodea pantailaratzen da eta *kont* kontagailuaren balio unitate batean inkrementatzen da
    d. *kont* kontagailuak 0 balio badu, saltzaile guztiek salmentaren bat egin dutela pantailaratu; bestela, *kont* kontagailu adina saltzaileek ez dutela salmentarik pantailaratu.

C KODEKETA

```
#include <stdio.h>
#include <stdlib.h>
#define SalKop 10
#define ZonaKop 20
#define ProduktuKop 5

main() {
float ventas[SalKop][ZonaKop];
float prodPrezioak[PropduktuKop] = {20.5, 50, 15, 5.25, 150};
float batu ;
int s, z, p, u, smax, zmax, kont ;
for(s=0 ; s<SalKop ; s++) {    // akaso saltzaile orok ez du
  for(z=0 ; z<ZonaKop ; z++) // zona guztietan salmentarik egiten
    {ventas[s][z]=0;}      // hortaz, matrizea hasieratu 0-z
}
/* Datu-Sarrera  */
printf("Sartu saltzaile eta zona zenbakiak: ");
scanf("%d %d", &s, &z);
while (s!=-1) {
  printf(" Sartu saldutako produktu kopurua eta unitateak: ");
  scanf("%d %d", &p, &u);
  batu=0;
  while (p!=-1) {      // errepikatzeka p-ren menpeko
    batu=batu + u * prodPrezioak[p];
    printf(" Sartu saldutako produktu kopurua eta unitateak: ");
    scanf("%d %d", &p, &u);
  }
  ventas[s][z]= ventas[s][z]+ batu;
  printf("Sartu saltzaile eta zona zenbakiak: ");
  scanf("%d %d", &s, &z);
 }

/* Pantailaratzea */
printf("Saltzailea / ");
for(z=0 ; z<ZonaKop; z++) { printf("Zona%d\t", z); }
```

```
printf("\n\n");

for(s=0 ; s<SalKop ; s++) {
  printf("    %d\t",s);
  for(z=0 ; z<ZonaKop ; z++) { printf("\t%5.2f", ventas[s][z]); }
  printf("\n");
}

/* Saltzaile onena  */
printf("\nEman zona: "); scanf("%d",&z);
smax=0;       // 0 saltzailea kontsideratzen dugu oraingoz onena
for(s=0 ; s<SalKop ; s++)
  if (ventas[smax][z]<ventas[s][z]) {smax=s;}
                        // uneko s saltzailea hobea da

printf("%d zonako saltzailerik onena %d saltzailea da.\n\n",
       z,smax);

/* Salmenta maximoa nork-non  */
smax=0;                 // Defektuz, ventas[0][0] gelaxkako balioa
zmax=0;                 // kontsideratzen dugu balio maximoa duela
for (s=0 ; s<SalKop ; s++)
  for(z=0 ; z<ZonaKop ; z++)
    { if (ventas[smax][zmax]<ventas[s][z])
                         // koordenatu pare honek balio hobea
      { smax=s; zmax=z;}  // duenez, gorde indizeak aldatu aurretik
    }
printf("Salmenta zenbateko handiena %d saltzaileak lortu du %d"
       " zonan.\n\n", smax, zmax);

/* Salmentarik ez  */
for(z=0 ; z<ZonaKop ; z++) {
  kont=0;
  printf("\nZona %d: ", z);
  for (s=0 ; s<SalKop ; s++)
  { if (ventas[s][z]==0)  { printf("  %d", s); kont++; }
```

```
  }
  if (kont!=0)
     {printf("\n%d saltzaileek ez dute salmentarik egin.", kont);}
  else {printf("\nSaltzaile guztiek salmentaren bat egin dute.");}
}
printf("\n\n");
system("PAUSE");
}
```

HERIOZKO ISTRIPUAK

Azkeneko oporraldian trafiko istripu kopurua oso altua izan da. Hori dela eta, 50 foru-aldundiek euren hildakoen istripuen datuak konpartitzea erabaki dute zenbait ondorio estatistiko ateratzeko.

Kontuan izango dituzten 4 heriozko istripu arrazoiak honako hauek dira: abiadura, alkohola, drogak eta bestelakoak.

Teleoperadore batek probintzi desberdinetako arduradunek hornitutako datuen erregistroa egingo du. Datuen sarrera hirukoteen bidez egingo da (probintzia, arrazoia, hildako-kopurua). Zauritutakoak hil litezkeenez, probintzia eta arrazoi bera duen hirukotea behin baino gehiagotan sartu beharko da; kasu honetan, dagoeneko metatuta dagoen balioari hildako-kopuru berria gehitu beharko zaio. Heriozko istripuen datu sarrera amaitzeko -1 ipini hirukoteko balioren batean. **Adibidez:** (lodiz teleoperadoraren datuak)

```
Heriozko istripuetan gertatutako hildakoen kopuruaren datu-
sarrera (hirukoteko bat -1 amaitzeko)
   Probintzia   arrazoia  hildakoak:  0  1  2
   Probintzia   arrazoia  hildakoak:  0  0  11
   Probintzia   arrazoia  hildakoak:  3  0  5
   Probintzia   arrazoia  hildakoak:  0  0  2
   …
   Probintzia   arrazoia  hildakoak:  0  -1  0
```

Hildakoen erregistroa egin ostean, aztertutako istripuentzat suhiltzailek eta erizainek burutu dituzten kilometroen erregistroa egingo da probintziaka. **Adibidez**: (lodiz teleoperadorearen datuak)

```
Orora ibilitako km-en erregistroa probintziaka:
   0 probintzia:  340
   1 probintzia:  124
   2 probintzia:  678
   …
   49 probintzia:  210
```

Metatutako datuekin ondoren eskatzen diren emaitzak kalkulatu beharko ditu zure C programak:

1. Kalkulatu heriozko istripuetan hildakoen kopurua probintziaka eta orotara zenbat hildako egon diren.
2. Hildako gehien izan dituen probintzia eta laguntza prestatzeko km gehien egin dituen probintzia bera den ala ez erabaki. **Adibidez**

```
Hildako gehien duen probintzia 35 da.
Laguntza prestatzeko km gehien egin duen probintzia 2 da.
Probintziak desberdinak dira.
```

3. Probintziaka kalkulatu hildako bakoitzeko korritu diren km-en bataz bestekoa. **Adibidez**

```
0 probintzia:  bataz bestean 23.2 km hildako bakoitzeko.
1 probintzia: bataz bestean 12.7 km hildako bakoitzeko.
...
```

4. Aztertutako istripu arrazoi bat eskatu erabiltzaileari eta arrazoi horren ondorioz gertatuko hildakoen kopurua zein izan den kalkulatu. Prozesua errepikatu sakatutako arrazoi kodea zuzena den bitartean. **Adibidez**

```
Istripu arrazoi bat eman:  1
  57 hildako metatu dira 1 arrazoiaren ondorioz

Istripu arrazoi bat eman:  0
  89 hildako metatu dira 0 arrazoiaren ondorioz

Istripu arrazoi bat eman:  -7
```

PROBLEMAREN ANALISIA

Enuntziatuko konstanteak

1. *ProbKop* = 50. Azterketan parte hartuko duten probintzia kopurua.
2. *Arrazoiak* = 4. Azterketak kontuan izango dituen heriozko istripuen arrazoi moten kopurua.

Sarrerarako beharrezkoak diren aldagaien adierazpidea

3. *heriozkoIstripuak* deituko dugun matrize bat beharko dugu, *ProbKop* x *Arrazoiak* posizio dituena, non probintzia bakoitzeko heriozko istripu arrazoi mota bakoitzeko gertatu diren hildakoen kopurua jasoko duen.

heriozkoIstripuak	0	1	…	Arrazoiak-1
0				
1				
…				
ProbKop-1				

4. *laguntzekoKm* deituko dugun bektore bat beharko dugu *ProbKop* posizioekin. Bertan probintzia bakoitzeko kontsideratutako heriozko istripuetako eriei laguntzeko korritu diren kilometro kopurua jasotzeko dugu.

	0	1	…	ProbKop-1
laguntzekoKm				

Emaitzetarako beharrezkoak diren aldagaien adierazpidea

5. *hildakoakP* deituko dugun bektore bat beharko dugu *ProbKop* gelaxkekin. Bertan, probintzia bakoitzeko hildakoen kopurua jasoko da aztertutako heriozko-istripuetan.

	0	1	…	HerriKop -1
hildakoakP				

6. *hildakoakA* deituko dugun bektore bat beharko dugu *Arrazoiak* gelaxkekin. Bertan, heriozko-istripuetako arrazoia bakoitzeko hildakoen kopurua jasoko da aztertutako istripuetan.

	0	1	…	Arrazoiak -1
hiladakoakA				

Problema ebazteko jarraibideak

1. *heriozkoIstripuak* matrizea 0-z hasieratzen da, probintzia eta arrazoi berdinak ematerakoan hildakoen kopurua gehitu egin behar baitzaio aurrez jasota dagoen hildakoen kopuruari.
2. Probintzia kodea, istripu arrazoi kodea eta hildakoen kopurua, (*pro, arr, hil*) hirukotea, eskatu

3. Hirukoteko balioren bat -1 ez den bitartean
    a. *heriozkoIstripuak* matrizeko (*pro, arr*) koordenatuan dagoen balioari *hil* balioa metatu
    b. Beste (*pro, arr, hil*) hirukote bat eskatu
4. Probintziaka desplazatutako kilometroak eskatu eta *laguntzekoKm* bektorean erregistratu.
5. Probintzia guztietako hildakoen kopurua metatzeko *guztira* aldagaia 0-ra hasieratu.
6. *pro* probintzia bakoitzeko:
    a. Kontsideratutako arrazoien ondorioz gertatu diren hildako guztien kopurua kalkulatu *hildakoakP[pro]* metatuz
    b. *guztira* aldagaiari metatu kalkulatu berri den probintziako hildakoen kopurua; hots, *hildakoakP[pro]*
7. *hildakoakP* bektorea aztertuz, hildako gehien duen probintzia kalkulatu *hMaxP* aldagaian.
8. *laguntzekoKm* bektorea aztertuz, hildakoak izan dituzten istripuetara laguntza eskaintzeko zein probintziatan korritu diren kilometro gehien kalkulatu eta *kmMaxP* aldagaian gorde.
9. konparatu *hMaxP* eta *kmMaxP* aldagaiak, erabakitzeko hildako gehien dituen probintzia eta hauek laguntzeko kilometro gehien egindako probintzia biak berdinak diren ala ez erabakitzeko.
10. probintziaka kalkulatu, *hildakoakP* eta *laguntzekoKm* bektoreak erabiliz, bataz bestean hildako bakoitzeko zenbat km korritu dituzten suhiltzaileek eta erizainek.
11. Arrazoika kalkulatu (*arr*)
    a. Probintzia guztietan aztergai den *arr* arrazoiak zenbat hildako eragin dituen *hildakoakA* bektorean erregistratuz
12. *arr* heriozko istripuetako arrazoi bat eskatu
13. *arr* baliozkoa den bitartean ($0 \leq arr < 4$), errepikatu:
    a. *hildakoakA* bektoretik *arr* arrazoiak eragin dituen hildakoak pantailaratu
    b. beste *arr* arrazoi bat eskatu

C KODEKETA

```
#include <stdio.h>
#include <stdlib.h>

#define ProbKop 50
#define Arrazoiak 4

main () {
int heriozkoIstripuak[ProbKop][Arrazoiak];
int laguntzekoKm[ProbKop], hildakoakP[ProbKop];
int hildakoakA[Arrazoiak];
int pro, arr, hil, guztira, hMaxP,kmMaxP;
float bb;

// Matrizea 0-z hasieratu behar da inkrementuak egiteko
for (pro=0;pro<ProbKop; pro++)
  for (arr=0; arr<Arrazoiak; arr++)
    heriozkoIstripuak[pro][arr]=0;

// Istripuetan hildakoen erregistroa
printf("Heriozko istripuetan gertatutako hildakoen kopuruaren\n"
       " datu-sarrera (hirukoteko bat -1 amaitzeko)\n");
printf("   Probintzia   arrazoia  hildakoak:  ");
scanf("%d %d %d", &pro, &arr, &hil);
while (pro!=-1 && arr!=-1 && hil!=-1) {
  heriozkoIstripuak[pro][arr]=heriozkoIstripuak[pro][arr]+hil;
  printf("   Probintzia   arrazoia  hildakoak:  ");
  scanf("%d %d %d", &pro, &arr, &hil);
}

// Laguntza prestatzeko korritutako km-ak probintziaka
for (pro=0;pro<ProbKop; pro++) {
  printf("%d probintzia: ", pro);
  scanf("%d", &laguntzekoKm[pro]);
}
```

```
// 1. Hildakoak probintziaka eta guztira
guztira=0;
for (pro=0;pro<ProbKop; pro++) {
  hildakoakP[pro]=0;
  for (arr=0; arr<Arrazoiak; arr++)
    hildakoakP[pro]=hildakoakP[pro]+ heriozkoIstripuak[pro][arr];
  printf("%d probintzian %d hildako egon dira.\n", pro,
         hildakoakP[pro]);
  guztira=guztira+hildakoakP[pro];
}
printf("Orora hildakoak %d izan dira.\n\n", guztira);

// 2. Kointziditzen dute bi probintziek?
// Hildako gehien
hMaxP=0;
for (pro=1;pro<ProbKop; pro++)
  if (hildakoakP[hMaxP] < hildakoakP[pro]) {hMaxP=pro;}
printf("Hildako gehien duen probintzia %d da.\n", hMaxP);

// Km korritu gehien
kmMaxP=0;
for (pro=1;pro<ProbKop; pro++)
  if (laguntzekoKm[kmMaxP] < laguntzekoKm[pro]) {kmMaxP=pro;}
printf("Laguntza prestatzeko km gehien egin duen probintzia"
       " %d da.\n", kmMaxP);

if (hMaxP==kmMaxP) {printf("Probintziak kointziditzen dute.\n\n");}
else {printf("Probintziak desberdinak  dira.\n\n");}

// 3. Probintziaka hildako bakoitzeko korritutako km-en media
for (pro=0;pro<ProbKop; pro++) {
  bb=(1.0*hildakoakP[pro])/laguntzekoKm[pro];
                        // zatidura erreala lortzeko
  printf("%d probintzia:  bataz bestean %.1f km hildako"
         " bakoitzeko.\n", pro, bb);
}
```

```
printf("\n\n");

// Hildakoak jasotako arrazoiaren ondorioz
// Arrazoi bakoitzeko zenbat egon diren kalkulatu
for (arr=0; arr<Arrazoiak; arr++) {
  hildakoakA[arr]=0;
  for (pro=0;pro<ProbKop; pro++)
     hildakoakA[arr]=hildakoakA[arr]+
                     heriozkoIstripuak[pro][arr];
}
// Arrazoiak eskatu eta goian kalkulatutako kopuru erakutsi
printf("Istripu arrazoi bat eman: ");
scanf("%d", &arr);
while (0<=arr && arr<Arrazoiak) {
  printf("%d hildako metatu dira %d arrazoiaren ondorioz.\n\n",
  hildakoakA[arr], arr);
  printf("Istripu arrazoi bat eman: ");
  scanf("%d", &arr);
}

printf("\n\n");
system("PAUSE");
}
```

EUSKADI IRRATIA

"*Euskadi Irratia*"-tik deitu dute programa ezberdinetan parte hartzen duten kolaboratzaileen informazioa gestionatzen lagunduko dien programa informatikoa egiteko. Azken urteetako denboraldietako datuak aztertu dituzte eta konturatu dira gehienez 100 kolaboratzaile eta gehienez 50 programa desberdin dituztela denboraldiko. Jakina da ere, kolaboratzaile bera programa desberdinetan hartzen duela parte eta gerta litekeela baten bat inongo programatan parte ez hartzea (osasun arazoengatik, azkenik zabaldu ez den atalen bat egitea espero zutelako, ...).

"Euskadi Irratia"-ko programazio arduradunak, aurreko datuez gain, programaren interfazea nolakoa izatea nahi duen ere esan digu. Jarraian datorren menua azaltzea nahiko luke:

```
A. Programazioa sartu.
B. Programa gehitu.
C. Kolaboratzaileak gehitu.
D. Kolaboratzaileen partaidetza sartu programa desberdinetan.
E. Kolaboratzaile bakoitzak kobratuko duena kalkulatu eta
   guztira "Euskadi Irratia"-k ordaindu beharrekoa.
F. Kolaboratzaileetan diru gehien gastatzen duen programa
   kalkulatu.
G. Kolaboratzailerik izan ez duten programak zerrendatu eta
   zenbat diren esan.
H. Bukatu.
```

Menuaren aukera bakoitzeko programak ondokoa egin beharko du:

A. Programazioa sartu.

Aukera hau sakatzean denboraldiz aldatu nahi dela adieraziko da; hau da, ez digu genuen informazioak balioko eta informazio berria emango digute (neguko denboralditik udarako denboraldira pasatu direlako adibidez). Datuak adibidean agertzen den moduan etorriko dira (letra lodiz erabiltzaileak emandako datuak):

```
Denboraldi honetako programen zerrenda eman (amaitzeko Bukatu
sakatu):
    Programa  0: Goiz kronika
    Programa  1: Faktoria
    Programa  2: Aperitifa
    Programa  3: Goizak Gaur
    …
    Programa 38: Bukatu
```

B. Programa gehitu.

Aukera honek programa berri bat gehituko dio programazioari ez bagara programa posibleen kopuru maximora iritsi. Programen maximora iritsi ezkero ondorengo mezua idatziko da:

```
Programa posibleen zerrenda beteta dago, informatika
zerbitzuarekin kontaktuan jarri.
```

Bestelako kasuan programa berri bat gehituko da:

```
38. programa sartu: Esaidazu
```

C. Kolaboratzaileak gehitu.

Aukera hau sakatzean kolaboratzaileak gehituko dira beti ere kolaboratzaileen maximoa gainditu gabe. Maximora iristen garen kasuetan honelako errore mezu bat emango digu:

```
Kolaboratzaile posibleen zerrenda beteta dago, informatika
zerbitzuarekin kontaktuan jarri.
```

Kolaboratzaileak sartu daitezkeenean komunikazioa honelakoa izango da (letra lodiz erabiltzaileak emandako datuak):

```
Kolaboratzaileak sartu (amaitzeko Bukatu sakatu):
    Kolaboratzailea : Iñigo Segurola
    Minutuko kobratzen duena(eurotan): 5
    Kolaboratzailea : Andoni Egaña
    Minutuko kobratzen duena(eurotan): 3.75
    Kolaboratzailea : Jesús Torquemada
    Minutuko kobratzen duena(eurotan): 6.5
    …
    Kolaboratzailea : Bukatu
```

D. Kolaboratzaileen partaidetza sartu programa desberdinetan.

Aukera hau sakatzean programa desberdinetan kolaboratzaileek izandako partaidetza berriak emango dira minututan. Programa bakoitzarentzat kolaboratzaile zerrenda eskatuko da bere partaidetza denborarekin, adibidean ikusten den moduan (letra lodiz erabiltzaileak emandako datuak):

```
Goiz kronika programa:
    Eman kolaboratzailea eta denbora (minututan): 2 150
    Eman kolaboratzailea eta denbora (minututan): 1 120
    Eman kolaboratzailea eta denbora (minututan): 25 150
    25. kolaboratzailea ez da existitzen.
    Eman kolaboratzailea eta denbora (minututan): 2 150
    ...
    Eman kolaboratzailea eta denbora (minututan): -1 0
Faktoria programa:
    Eman kolaboratzailea eta denbora (minututan): 2 150

    ...
    Eman kolaboratzailea eta denbora (minututan): -1 0
...
```

E. Kolaboratzaile bakoitzak kobratuko duena kalkulatu eta guztira "*Euskadi Irratia*"-k ordaindu beharrekoa.

Kolaboratzaile bakoitzari ordaindu beharrekoa kalkulatu orain arte izandako partaidetza desberdinak kontutan hartuz eta kalkulatu ere,

guztira irratiak kolaboratzaileei ordaindu beharrekoa. Kolaboratzaile batek orain arte ez badu parte hartzerik izan ez da zerrendan agertuko.

```
Andoni Egaña 120 minututan parte hartu du: 450 euro
Jesús Torquemada 450 minututan parte hartu du: 2925 euro
“Euskadi Irratiak” kolaboratzaileetan gastatuko duena: 3375 euro
```

F. Kolaboratzaileetan diru gehien gastatzen duen programa kalkulatu.

Orain arte ditugun datuekin, kalkulatu zein programak gastatu duen diru gehien kolaboratzaileetan eta zenbat den esan.

```
Kolaboratzaileetan diru gehien gastatzen duen programa Goiz
Kronika da 2400 eurorekin.
```

G. Kolaboratzailerik izan ez duten programak zerrendatu eta zenbat diren esan.

Adierazi orain arte zein programatan ez duen kolaboratzailerik parte hartu eta zenbat kasutan gertatu den hau esan.

```
Ez du kolaboratzailerik parte hartu ondoko programatan:
    Goizak Gaur
    Klasikoak EITBn
Guztira 2 programa dira.
```

H. Bukatu.

Programa amaitu egingo da aukera hau sakatzean.

PROBLEMAREN ANALISIA

Enuntziatuko konstanteak

1. *KolaboMax* = 100. Irratiak gehienez izan ditzakeen kolaboratzaile kopuru maximoa
2. *ProgMax* = 50. Irratiak denboraldi bakoitzean izan ditzakeen programen kopuru maximoa.
3. *Luzera* = 70. Kolaboratzaile eta programa izen bakoitzak gehienez izan ditzakeen karaktere kopuru maximoa

Sarrerarako beharrezkoak diren aldagaien adierazpidea

4. Programen izenak gordetzeko karaktere matrize bat (*progIzenak* deitua) beharko dugu, *ProgMax* lerro izango ditu eta lerro bakoitzak *Luzera* zutabe izango ditu.

progIzenak	*0*	*1*	*...*	*Luzera-1*
0				
1				
...				
ProgMax-1				

5. Kolaboratzaileen izenak gordetzeko karaktere matrize bat (*kolaboIzenak*) beharko dugu, *KolaboMax* lerro eta lerro bakoitzak *Luzera* zutabe izango ditu.

kolaboIzenak	*0*	*1*	*...*	*Luzera-1*
0				
1				
...				
KolaboMax-1				

6. Kolaboratzaile bakoitzak minutuko kobratuko duena gordetzeko *KolaboMax* posiziotako bektore erreala beharko dugu, *kolaIrabMin* deitua.

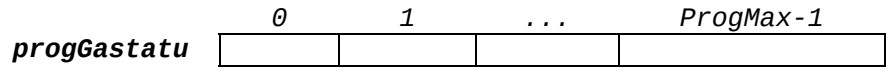
	0	*1*	*...*	*KolaboMax-1*
kolaIrabMin				

7. Matrize bat (*ProgMax* x *KolaboMax*) beharko dugu kolaboratzaile bakoitzak parte hartu duen programa bakoitzean zenbat hitz egin duen gordetzeko.

partaidetza	*0*	*1*	*...*	*KolaboMax-1*
0				
1				
...				
ProgMax-1				

Emaitzetarako beharrezkoak diren aldagaien adierazpidea

8. Programa bakoitza kolaboratzaileetan gastatu duen totala gordetzeko *ProgMax* posizio dituen bektore erreala beharko da.



Problema ebazteko jarraibideak

1. Irratiak dituen kolaboratzaile kopurua 0ra hasieratu (*kolaboKop*).
2. Errepikatu
   a. Menua pantailan idatzi
   b. Aukera irakurri (*aukera*)
   c. Irakurritako *aukera* 'A' bada, programazio berria sartu. Honetarako:
      i. Programen kontadorea 0ra hasieratu (*progKop*)
      ii. *partaidetza* matrizea 0ra hasieratu
      iii. Programa izena irakurri (*izen*)
      iv. Programen kontadorea *ProgMax* baino txikiagoa eta irakurritako izena "Bukatu" ez den bitartean, irakurritako programa *progIzenak* matrizean kopiatu, programen kontadorea batean handitu eta hurrengo programa izena irakurri.
   d. Irakurritako *aukera* 'B' bada, programa gehitu. Honetarako:
      i. Programen kontadorea irratiak izan ditzakeen programa kopuru maximora ez bada iritsi
         (1) orduan programa kontadorea batean handitu, programa izena irakurri eta *progIzenak* matrizean gorde
         (2) bestela, errore abisua eman
   e. Irakurritako *aukera* 'C' bada, kolaboratzaileak gehitu. Honetarako:
      i. Kolaboratzaileen zerrenda beteta badago
         (1) orduan, errore abisua eman
         (2) bestela, kolaboratzailea irakurri. Irakurritako kolaboratzaile izena "Bukatu"ren desberdina den bitartean eta kolaboratzaile zerrendan lekua dagoen bitartean, irakurritako kolaboratzailea *kolaboIzenak* matrizean kopiatu, minutuko kobratzen duena irakurri, kolaboratzaile kopurua batean handitu eta hurrengo kolaboratzaile izena irakurri.
   f. Irakurritako *aukera* 'D' bada, kolaboratzaileen partaidetza berria irakurri. Honetarako:

i. Programa bakoitzerako (*p*)

(1) Kolaboratzaile eta denbora irakurri

(2) Kolaboratzailea -1en desberdina den bitartean egin

(a) Kolaboratzailea existitzen bada irakurritako minutuak gehitu orain arteko denborari, bestela errore mezua eman

(b) Kolaboratzaile eta denbora berria irakurri

g. Irakurritako *aukera* 'E' bada, kolaboratzaile bakoitzari ordaindu beharrekoa kalkulatu eta guztira irratiak kolaboratzaileetan gastatuko duena kalkulatu. Honetarako:

i. Irratiak kolaboratzaileengatik ordaintzen duena 0ra hasieratu

ii. Kolaboratzaile bakoitzerako,

(1) Programa guztiak kontutan hartuta parte hartutako minutuak kalkulatu

(2) Hitz egindako minutuak bidertu minutuko kobratzen duenarekin eta 0 baino handiagoa bada pantailan idatzi

(3) Irratiak guztira ordaindu behar duenari gehitu kolaboratzaile honi ordaindu beharrekoa.

iii. Irratiak guztira ordaindu beharrekoa pantailan idatzi.

h. Irakurritako *aukera* 'F' bada, kolaboratzaileetan diru gehien gastatu duen programa kalkulatu. Honetarako:

i. Programa bakoitzak kolaboratzaileetan gastatu duena kalkulatu behar da eta *progGastatu* bektorean gorde.

Programa bakoitzerako

(1) programak kolaboratzaileetan gastatu duena 0ra hasieratu

(2) kolaboratzaile bakoitzerako, programa horretan hitz egindakoa bidertu minutuko kobratzen duenari eta emaitza batu programak kolaboratzaileetan gastatu duenari gehitu.

ii. *progGastatu* bektoreko maximoaren posizioa kalkulatu kolaboratzaileetan diru gehien gastatu duen programa zein den jakiteko.

i. Irakurritako *aukera* 'G' bada, kolaboratzailerik izan ez duten programak zeintzuk diren esan eta guztira zenbat izan diren ere. Honetarako:

i. Kolaboratzailerik ez duten programen kontadorea (*þKoþ*) 0ra hasieratu

ii. Programa bakoitzerako kolaboratzaileetan gastatu dena 0 bada, kolaboratzailerik ez dela egon adieraziko du, hortaz *þkoþ* kontadorea batean handitu eta programaren izena idatzi.

iii. Kolaboratzailerik ez duten programen kontadorea (*þKoþ*) pantailan idatzi.

j. Irakurritako *aukera* 'H' bada "*Programa amaitzera doa* …" mezua idatzi.

k. Irakurritako *aukera* aurrekoren bat ez bada aukera okerra dela esaten duen mezua idatzi

Irakurritako *aukera* 'H' ez den bitartean

C KODEKETA

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
/* Euskadi Irratiaren Programazioa */
/* Konstanteen definizioa*/
#define KolaboMax  100
#define ProgMax     50
#define Luzera      70
main(){
/*Programa desberdinen izena gordetzeko erabiliko dugun matrizea */
char progIzenak[ProgMax][Luzera];
/*Kolaboratzaileen izenak gordetzeko erabiliko dugun matrizea */
char kolaboIzenak[KolaboMax][Luzera];
/*Programak kolaboratzaileetan gastatu dutena gordetzeko bektorea
*/
/*Kolaboratzaileek minutuko kobratzen duena gordetzeko bektorea*/
float progGastatu[ProgMax], kolaIrabMin[KolaboMax];
/*Kolaboratzaile bakoitzak parte hartu duen programa bakoitzean
zenbat minutu hitz egin duen gordeko duen matrizea*/
int partaidetza[ProgMax][KolaboMax];
int p, k, min, progKop, kolaboKop, pmax, pKop, kKop;
char aukera, izen[Luzera];
float irabaziak, guztIrratia;

kolaboKop=0;
/*kolaboratzaileen kontadorea 0ra hasieratu*/
do {
  /*Menua idatzi*/
  printf("A) Programazioa sartu.\n");
  printf("B) Programa gehitu.\n");
  printf("C) Kolaboratzaileak gehitu.\n");
  printf("D) Kolaboratzaileen partaidetza sartu programa "
         "desberdinetan\n");
  printf("E) Kolaboratzaile bakoitzak kobratuko duena kalkulatu"
         " eta guztira Euskadi Irratiak ordaindu beharrekoa\n");
```

```
printf("F) Kolaboratzaileetan diru gehien gastatzen duen"
       " programa kalkulatu\n");
printf("G) Kolaboratzailerik izan ez duten programak"
       " zerrendatu eta zenbat diren esan.\n");
printf("H) Bukatu\n");
printf("\n");
/*Aukera irakurri*/
printf("Eman zure aukera:");
fflush(stdin);
scanf("%c", &aukera);
switch (aukera) {
  case 'A':
     /*Programazio berria sartu.*/
     /*Lehenik partaidetza matrizea 0ra hasieratu behar da eta
       programen kontadorea 0ra jarri*/
     progKop=0;
     for (p=0; p<ProgMax; p=p+1) {
       for (k=0; k<KolaboMax; k=k+1) { partaidetza[p][k]=0;}
     }
     /*Programazio berria irakurri*/
     printf("Denboraldiko programen zerrenda eman");
     printf("amaitzeko Bukatu idatzi):\n");
     printf("Programa %d: ",progKop);
     fflush(stdin);
     gets(izen);
     while (strcmp(izen, "Bukatu")!=0 && (progKop<ProgMax)) {
       strcpy(progIzenak[progKop], izen);
       progKop=progKop+1;
       printf("Programa %d: ",progKop);
       fflush(stdin);
       gets(izen);
      }
     break;
  case 'B':  /*Programa berria gehitu.*/
     if (progKop==ProgMax) {
     /*programa posibleen zerrenda beteta dago*/
```

```
      printf("Programa posibleen zerrenda beteta dago, jarri"
             " kontaktuan informatika zerbitzuarekin\n");
      }
      else { /*programa gehitu daiteke*/
        printf("Programa %d: ",progKop);
        fflush(stdin);
        gets(progIzenak[progKop]);
        progKop=progKop+1;
      }
      break;
    case 'C': /*Kolaboratzaileak gehitu. Lehenengo kolaboratzaileen
              zerrenda beteta dagoen konprobatu*/
      if (kolaboKop==KolaboMax) {
        printf("Kolaboratzaile posibleen zerrenda beteta dago,"
               " informatika zerbitzuarekin kontaktuan jarri\n");
      }
      else { printf("Kolaboratzaileak sartu (amaitzeko Bukatu "
                    " idatzi):\n");
        printf("Eman kolaboratzailea:");
        fflush(stdin);
        gets(izen);
        while (strcmp(izen,"Bukatu")!=0 && kolaboKop<KolaboMax) {
          strcpy(kolaboIzenak[kolaboKop], izen);
          printf("Minutuko kobratzen duena (eurotan):");
          scanf("%f", &kolaIrabMin[kolaboKop]);
          kolaboKop=kolaboKop+1;
          printf("Eman kolaboratzailea:");
          fflush(stdin);
          gets(izen);
        }
      }
      break;
    case 'D': /*Kolaboratzaileen partaidetza sartu programa
                desberdinetan.*/
      for (p=0; p<progKop; p=p+1) {
         /*Irakurri den programa bakoitzerako izandako
```

```
        kolaboratzaileen informazioa irakurri.*/
        printf("Programa %s:\n", progIzenak[p]);
        printf("\tEman kolaboratzailea eta denbora "
               " (minututan):");
        scanf("%d %d", &k, &min);
        while (k!=-1) {
          /* Kolaboratzailea existitzen da? */
          if (k<kolaboKop && k>=0) {
              /*kolaboratzailea existitzen bada minutuak gehitu*/
              partaidetza[p][k]=partaidetza[p][k]+min;
          }
          else {
            printf("%d kolaboratzailea ez da existitzen\n", k);
            }
          printf("\tEman kolaboratzailea eta denbora "
                 " (minututan):");
          scanf("%d %d", &k, &min);
          }
        }
        break;
    case 'E': /*Kolaboratzaile bakoitzak kobratuko duena kalkulatu
             eta guztira irratiak ordaindu beharrekoa.*/
        guztIrratia=0;
        for (k=0; k<kolaboKop; k=k+1) {
          /*Kolaboratzaileka hitz egindako minutuak kalkulatu*/
          min=0;
          for (p=0; p<progKop; p=p+1) {
            min=min+partaidetza[p][k];
        }
        /*hitz egindako minutuak minutuko irabazten duenarekin
        bidertu*/
        irabaziak=min*kolaIrabMin[k];
        guztIrratia=guztIrratia+irabaziak;
        /*Kolaboratzaileak irabazirik badu pantailan idatzi*/
        if (irabaziak>0) {
```

```
          printf("%s kolaboratzaileak %d minutu hartu du parte"
                 " %8.2f euro irabaziz.\n", kolaboIzenak[k],
                   min, irabaziak);
          }
        }
       /*Irratiak guztira kolaboratzaileetan gastatutakoa idatzi*/
       printf("Euskadi Irratiak guztira kolaboratzaileetan "
              "%8.2f euro gastatu ditu", guztIrratia);
    break;
  case 'F': /*Zein programak gastatu du diru gehien
            kolaboratzaileetan.*/
    /*Programa bakoitzak gastatu duena kalkulatu*/
    for (p=0; p<progKop; p=p+1) {
      /*Programa bakoitzak kolaboratzaileetan gastatu duena
      kalkulatu eta progGastatu bektorean gorde*/
      progGastatu[p]=0;
      for (k=0; k<kolaboKop; k=k+1) {
        progGastatu[p]=progGastatu[p]+
            (partaidetza[p][k]*kolaIrabMin[k]);
      }
    }
    /*progGastatu bektoreko maximoaren posizioa kalkulatu*/
    pmax=0;
    for (p=1; p<progKop; p=p+1) {
      if (progGastatu[pmax]<progGastatu[p]) {pmax=p;}
    }
    printf("Kolaboratzaileetan diru gehien gastatu duen "
           "programa %s da %8.2f eurorekin\n",
           progIzenak[pmax], progGastatu[pmax]);
    break;
  case 'G': /*Zein programatan ez du inongo kolaboratzailerik
            parte hartu eta zenbat dira.*/
    printf("Ez du kolaboratzailerik parte hartu ondorengo"
           " programetan:\n");
    pKop=0;
    for (p=0; p<progKop; p=p+1) {
```

```
          /*Programak kolaboratzaileetan gastatu duena 0 bada, ez du
            kolaboratzailerik izan */
          if (progGastatu[p] == 0) {
            printf("%s \n", progIzenak[p]);
            pKop=pKop+1;
          }
        }
        printf("Guztira %d programa dira.\n", pKop);
        break;
    case 'H': /*Programa amaiera*/
        printf("Programa amaitzera doa ...\n");
        break;
    default: /*aukera okerra*/
           printf("Aukera okerra!!!\n");
  }
} while (aukera != 'H');

printf("\n\n");
system("PAUSE");
}
```

SAN SEBASTIANAK

San Sebastian eguneko haurren danborradaren zenbait datu ezagutzeko Donostiako udalak programa bat egitea eskatu digu. Programa honek parte hartzen duten konpainia desberdinen informazioa eskatuko digu (konpainiaren izena, sorrera-urtea eta geroztik urte bakoitzean ateratako postuak). Konpainia guztiek beren sorrera-urtetik hasita urte guztietan parte hartu dute desfilean.

Udalak esan digu gaur egun desfilean 50 konpainiek parte hartzen dutela eta 1980. urtetik hasita 2006. urtera arteko datuak dauzkala.

Hasieran, programak konpainia guztien izenak eta sorrera-urteak eskatu beharko dizkio erabiltzaileari, jarraian azaltzen den moduan (adibideetan **letra lodia** erabili da erabiltzaileak sartutako datuak adierazteko):

```
Haur konpainia desberdinen datuen sarrera:
        Izena: Mundaiz
        Urtea: 1980
        Izena: Ibai
        Urtea: 1984
        ...
```

Ondoren, konpainia bakoitzarentzat ateratako urte bakoitzerako zein postutan atera den eskatuko dio erabiltzaileari. Kontuan har ezazu, konpainia bakoitzarentzat datuak sorrera-urtetik aurrera eskatu behar dizkiola erabiltzaileari (adibideetan **letra lodia** erabili da erabiltzaileak sartutako datuak adierazteko):

```
Mundaiz konpainiaren postuak eman:
        1980 urtea: 3
        1981 urtea: 7
        ...
        2006 urtea: 1
Ibai konpainiaren postuak eman:
        1984 urtea: 13
        1985 urtea: 21
        ...
        2006 urtea: 2
.............
```

Datu hauek guztiak irakurri ondoren programak menu baten bitartez erabiltzaileari honako aukera hauek eskaini behar dizkio, erabiltzaileak "irten" sakatu arte. Erabiltzaileak menuko aukeren artean ez dagoen zenbaki bat sakatuz gero, programak "Emandako aukera okerra da." idatzi eta berriro ere menua aurkeztu beharko du.

Hau da programak idatzi beharreko menua:

```
Menua:
        1.-Informazioa ikusi
        2.-Lehen postuan atera ez diren konpainien izenak eta zenbat
           diren
```

```
3.-Erabiltzaileak emandako urteetarako kalkulatu zenbat
   konpainia atera diren
4.-Esan zenbat konpainia atera dira bere sorrera-urtean
   azken postuan
5.-Irten
Eman zure aukera: 2
```

Aukera bakoitza sakatzean ondokoa egin beharko du programak:

1. Informazioa ikusi. Ondorengo adibidean azaltzen den moduan, taula-formatuan idatzi konpainia bakoitza urte bakoitzean zein postutan atera den. Konpainia bat urte batean atera ez bada gidoia idatzi beharko da. Adibidea:

	1980	*1981*	*1982*	*1983*	*1984*	*....*	*2006*
Mundaiz	*3*	*7*	*1*	*3*	*5*	*....*	*1*
Ibai	*-*	*-*	*-*	*-*	*13*	*....*	*2*

.....

2. Lehen postuan atera ez diren konpainien izenak eta zenbat diren. Desfilean inoiz lehen postuan atera ez diren konpainien izenak eman. Amaieran zenbat izan diren adierazi. Adibidea:

```
Lehen postuan inoiz atera ez diren konpainiak Ekintza, Aitor
dira. Guztira 2 konpainia direlarik.
```

3. Erabiltzaileak emandako urteetarako kalkulatu zenbat konpainia atera diren. Erabiltzaileari urte sekuentzia bat eskatu eta emandako urte bakoitzerako programak zenbat konpainia atera diren erantzungo dio. Urte segida erabiltzaileak –1 urtea sartzean amaituko da. Adibidea:

```
Emandako urte bakoitzerako ateratako konpainia kopurua emango
dizut (amaitzeko –1 sakatu):
    Eman urtea: 1997
    Ateratako konpainiak 23 dira
    Eman urtea: 2001
    Ateratako konpainiak 45 dira
    Eman urtea: 1991
    Ateratako konpainiak 16 dira
    Eman urtea: -1
```

4. Esan zenbat konpainia atera dira bere sorrera-urtean azken postuan. Kontatu zenbat konpainia atera diren bere sorrera-urtean azken postuan.

5. Irten. Programa amaituko da.

Oharra:

A) Urte bat array-ean daukan posizioarekin erlazionatzeko, dagokion urteari hasiera-urtea kenduta kalkulatu daiteke. Adib: 1991- 1980 = 11 beraz, 1991. urtea 11. posizioan egongo da eta 1980 urtea 0. posizioan.

PROBLEMAREN ANALISIA

Enuntziatuko konstanteak

1. *KonpainiaKop* = 50. Konpainia kopurua adierazten du
2. *HasUrtea* = 1980. Haur konpainien desfilea hasi zen urtea
3. *BukUrtea* = 2006. Haur konpainien desfilea egin den azken urtea
4. *UrteKop* = *HasUrtea-BukUrtea*+1
5. *Luzera* = 40. Konpainia bakoitzaren izenaren luzera maximoa

Sarrerarako beharrezkoak diren aldagaien adierazpidea

6. Konpainia bakoitzaren izena gordetzeko karaktere matrize bat behar dugu. Lerro bakoitzean konpainia bakoitzaren izena gordeko delarik.

izenak	*0*	*1*	*...*	*Luzera-1*
0				
1				
...				
KonpainiaKop-1				

7. Konpainia bakoitzaren sorrera urtea gordetzeko bektorea. Posizio kopurua dauden konpainia kopuruaren berdina izan behar du.

	0	*1*	*...*	*KonpainiaKop-1*
SorreraUrtea				

8. Konpainia bakoitza urte bakoitzean zein postutan atera zen gordetzen duen matrizea. *KonpainiaKop* lerro izango ditu eta lerro bakoitzean *BukUrtea-HasUrtea*+1 zutabe egongo dira

postuak	*0*	*1*	*...*	*UrteKop-1*
0				
1				
...				
KonpainiaKop-1				

Emaitzetarako beharrezkoak diren aldagaien adierazpidea

Problema ebazteko jarraibideak

1. *postuak* matrizea 0ra hasieratu
2. Datuak irakurri:
   a. Konpainia bakoitzaren izena (*izenak* taulan gorde) eta sorrera urtea (*sorreraUrtea* taulan gorde)
   b. Konpainia bakoitzarentzat bere sorrera urtetik hasita ateratako postuak irakurri (*postuak* matrizean gorde datuak)
3. Errepikatu

a. Menua idatzi

b. Aukera irakurri

c. Aukera bakoitza tratatu:

    i. Informazioa ikusi

        (1) Urteak idatzi lerro batean (*HasUrtea*-tik hasi *BukUrtea* arte)

        (2) Konpainia bakoitzarentzat

            (a) Konpainia izana idatzi

            (b) Urte bakoitzarentzat

                (i) Baldin konpainia hori urte horretan atera bada (hau da *postuak* matrizean 0ren desberdina den balioa badu) orduan matrizeko balioa idatzi, bestela gidoia idatzi.

            (c) Lerro jauzia idatzi

    ii. Emandako urteetarako ateratako konpainia kopurua kalkulatu

        (1) Urtea irakurri (*u*)

        (2) Bitartean *u* desberdin –1 izan

            (a) *kont*=0

            (b) Konpainia bakoitzarentzat

                (i) Baldin konpainia hori *u* urtean atera bada *kont* batean handitu

            (c) Idatzi *kont*-en balioa

            (d) Urtea irakurri(*u*)

    iii. Sorrera urtean azken postuan atera ziren konpainia kopurua kalkulatu

        (1) *azkenPostu*=0

        (2) Konpainia bakoitzarentzat (*kAzken*)

            (a) Lortu konpainiaren sorrera urtea (*u*)

            (b) *azkenKon*=0 /**u* urtean azken posizioan atera den konpainia gordeko du*/

            (c) Konpainia bakoitzarentzat (*k*)

                (i) Baldin *k* konpainia *u* urtean *azkenKon* konpainia baino atzerago atera bazen *azkenKon* bezala gorde *k* konpainia

(d) Baldin *kAzken* konpainia *azkenKon*-en berdina bada (adierazi nahi du, sorrera urtean azken postuan atera zela) orduan *azkenPostu* kontadorea batean handitu.

(3) Idatzi *azkenPostu* aldagaiak duen balioa

iv. programa amaitzera doa idatzi

v. bestela: emandako aukera okerra da idatzi

emandako aukera 4ren desberdina den bitartean

C KODEKETA

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>

#define HasUrtea 1980
#define BukUrtea 2006
#define KonpainiaKop 50
#define UrteKop BukUrtea-HasUrtea+1
#define Luzera 40

main(){
int postuak[KonpainiaKop][UrteKop];
int sorreraUrtea[KonpainiaKop];
char izenak[KonpainiaKop][Luzera];
int p, u, k, kont, aukera, azkenKon, kAzken, azkenPostu;

/* postuak matrizea 0-ra hasieratu */
for (k=0; k<KonpainiaKop; k=k+1) {
  for(u=0; u<UrteKop;u=u+1) { postuak[k][u]=0; }
}

/* Konpainia izenak eta sorrera urteak irakurri */
printf("Haur konpainia desberdinen datuen sarrera:\n");
for (k=0; k<KonpainiaKop; k=k+1) {
  printf("Izena: ");  fflush(stdin);  gets(izenak[k]);
  printf("Urtea: ");
  scanf("%d", &sorreraUrtea[k]);
}

/* Konpainia bakoitza urte bakoitzean ateratako postua irakurri*/
for (k=0; k<KonpainiaKop; k=k+1) {
  printf("%s konpainiaren postuak eman: \n", izenak[k]);
  for(u=sorreraUrtea[k]; u<=BukUrtea; u=u+1) {
    printf("%d urtea: ", u);
    scanf("%d", &postuak[k][u-HasUrtea]);
```

```
  }
}

do {
  printf("Menua:\n");  /* Menua idatzi */
  printf("\t1.- Informazioa ikusi\n");
  printf("\t2.- Erabiltzaileak emandako urteetarako kalkulatu"
         " zenbat konpainia atera diren\n");
  printf("\t3.- Esan zenbat konpainia atera dira bere sorrera-"
         " urtean azken postuan\n");
  printf("\t4.- Irten\n");
  printf("\nEman zure aukera: ");
  /* Aukera irakurri */
  scanf("%d", &aukera);
  /* Aukera tratatu*/
  switch (aukera) {
    case 1: /* Pantailaratu matrizeko datuak */
         /* Urteak idatzi */
         printf("\t\t");
         for (u=HasUrtea; u<=BukUrtea; u=u+1)
           { printf("%d\t", u); }
         printf("\n");
         /* Konpainia bakoitzaren datuak idatzi*/
         for (k=0; k<KonpainiaKop; k=k+1) {
           printf("%-30s", izenak[k]);
           for(u=0; u<UrteKop; u=u+1) {
             if (postuak[k][u]==0) {printf("-\t");}
             else {printf("%d\t",postuak[k][u]);}
           }
           printf("\n");
         }
         break;
    case 2: /* Emandako urte bakoitzerako zenbat konpainia atera
               diren kalkulatu*/
        printf("Emandako urte bakoitzerako ateratako konpainia"
               " kopurua emango dizut (amaitzeko -1 sakatu)\n");
        printf("Eman urtea: ");
```

```
        scanf("%d", &u);
        while (u != -1) {
          kont=0;
          for (k=0; k<KonpainiaKop; k=k+1) {
            if (postuak[k][u-HasUrtea]!=0) { kont=kont+1; }
          }
          printf("%d urtean %d konpainia atera dira\n\n", u, kont);
          printf("Eman urtea: ");
          scanf("%d", &u);
        }
        break;
    case 3: /* Sorrera urtean azken postuan atera diren konpainia
               kopurua kalkulatu*/
        azkenPostu=0;
        for (kAzken=0; kAzken<KonpainiaKop; kAzken=kAzken+1){
          u = sorreraUrtea[kAzken]-HasUrtea;
          /* u urtean azken postuan atera zen konpainia kalkulatu*/
          azkenKon=0;
          for (k=0; k<KonpainiaKop; k=k+1) {
            if (postuak[k][u]>postuak[azkenKon][u]) {azkenKon = k;}
          }
        if (azkenKon == kAzken) { azkenPostu=azkenPostu+1; }
        }  // for
        printf("Sorrera urtean azken postuan atera diren konpinia"
               " kopurua %d da\n", azkenPostu);
        break;
    case 4:  printf("\n\nAmaitzera noa ...\n\n");  break;
    default: printf("\n\nEmandako aukera okerra da.\n\n");
  }
} while (aukera!=4);

printf("\n\n");
system("PAUSE");
}
```