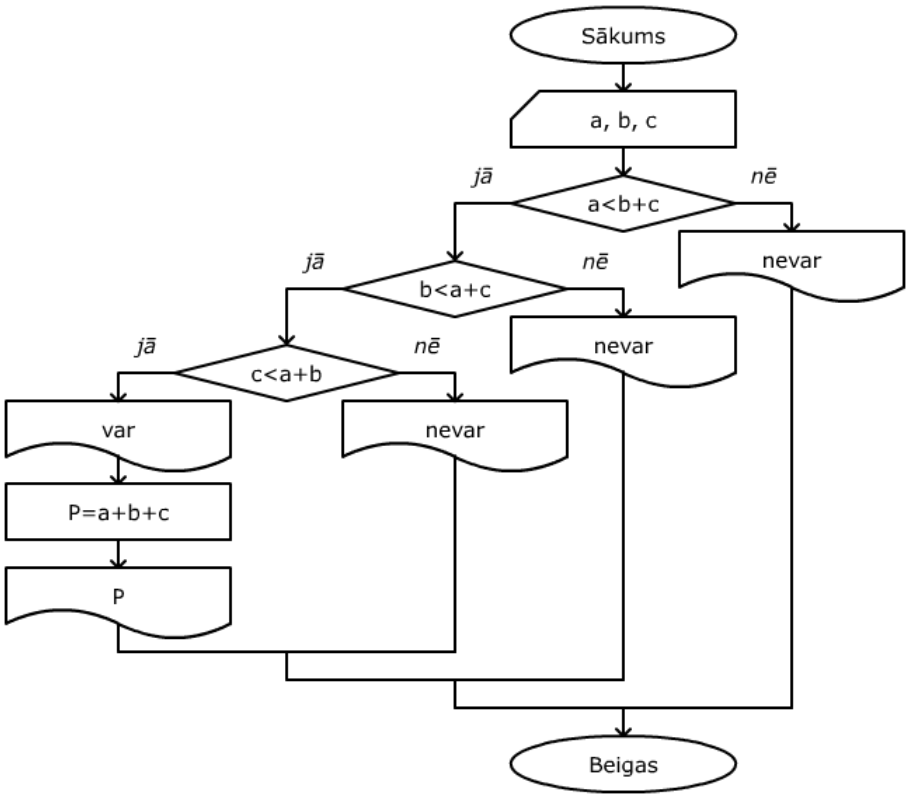
2012. gada Liepājas Universitātes
Datorzinātņu olimpiādes
studentu programmēšanas konkursa uzdevumi

1. uzdevums [*uzd1*] (5 punkti)
Sastādīt programmu pēc dotās algoritma blokshēmas. Ja iespējams, optimizē programmas kodu. Ar vārdiem raksturo, kādu uzdevumu risina šis algoritms.



2. uzdevums [*uzd2*] (8 punkti)
Zinātnieki pētīja baktēriju "A" un baktēriju "B" savstarpējo iedarbību. Viņi konstatēja, ka

- katru stundu viena baktērija "A" iznīcina divas baktērijas "B",
- no katras baktērijas "A" stundas laikā izveidojas vēl divas "A" tipa baktērijas;
- katra baktērija "B" vairojas vienu reizi stundā daloties uz pusēm.

Lai pārliecinātos par faktu pareizību, zinātnieki traukā ievietoja N "A" tipa baktērijas un M "B" tipa baktērijas. Zināms, ka zinātnieki traukā ievietoja vismaz 1 "A" un vismaz 1 "B" tipa baktēriju, pie tam kopīgais baktēriju skaits traukā nebija mazāks par 10 un nepārsniedza 100 baktērijas.
Uzzīmēt blokshēmu, kas ļauj noteikt, pēc cik stundām traukā būs tikai viena veida baktērijas.
Sastādīt programmu, kas drukā ekrānā katra tipa baktēriju skaitu ik pa stundai, līdz traukā būs palikušas tikai viena veida baktērijas. Baktēriju skaitu - N un M vērtības ievada lietotājs, pie tam programmai jāziņo par kļūdu, ja tiek ievadīts nekorekts baktēriju skaits.

3. uzdevums [*uzd3*] (7 punkti)
Reinim ir N klucīši, viņš vēlas no tiem uzbūvēt piramīdu ar tukšu vidu, tā, ka augšējais stāvs sastāv no viena klucīša, otrais stāvs no augšas sastāv no četriem klucīšiem, trešais stāvs no augšas sastāv astoņiem klucīšiem, ceturtais stāvs no divpadsmit klucīšiem, utt.
Sastādīt programmu, kas aprēķina, cik stāvi būs piramīdai un cik klucīšu paliks pāri.
Teksta datnē *uzd3.in* dots viens naturāls skaitlis - klucīšu skaits, zināms, ka 0<N<32000.
Programmas darbības rezultātu izvadīt teksta datnē *uzd3.out*. Pirmajā rindiņā izvadīt naturālu skaitli - piramīdas stāvu skaitu, bet otrajā rindiņā pāri palikušo klucīšu skaitu.

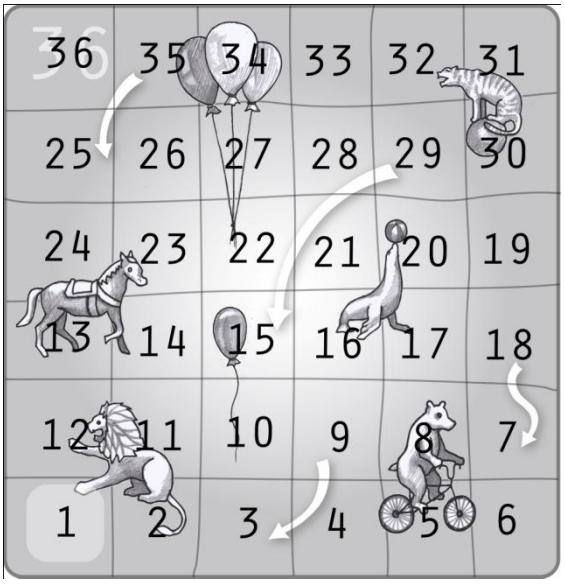
4. uzdevums [*uzd4*] (10 punkti)
Sastādīt programmu, kas izdrukā ekrānā skaitļu rindu un tās mediānu.
Skaitļu rindas elementu skaitu ievada lietotājs, elementu skaits nedrīkst pārsniegt 30 un elementu vērtības ir nejauši skaitļi, kas pieder intervālam [10;90].
Piezīme: Mediāna ir vidējais skaitlis sakārtotā skaitļu rindā, ja skaitļu rindā ir nepāra elementu skaits, bet divu vidējo skaitļu vidējais aritmētiskais, ja skaitļu rindā ir pāra elementu skaits.

5. uzdevums [*uzd5*] (8 punkti)
Sastādi programmu, kas realizē doto algoritmu.

1.solis. Lietotājs ievada N vērtību, 0<n<100.
2.solis. Aizpilda masīva A[N] elementu vērtības ar nejaušiem skaitļiem.
3.solis. Izvada masīva A[N] elementus ekrānā.
4.solis. Sākot no masīva pirmā elementa caurskata masīva elementu vērtības un atrod mazāko.
5.solis. Atrasto mazāko vērtību apmaina vietām ar masīva pirmo vērtību.
6.solis. Sākot ar otro masīva elementu caurskata masīvu vērtības un atrod mazāko.
7.solis. Atrasto mazāko vērtību apmaina vietām ar masīva otro vērtību.
8.solis. Sākot ar trešo masīva elementu caurskata masīvu vērtības un atrod mazāko.
9.solis. Atrasto mazāko vērtību apmaina vietām ar masīva trešo vērtību.
...
2N.solis. Sākot ar priekšpēdējo elementu caurskata masīva vērtības un atrod mazāko.
2N+1.solis. Atrasto mazāko vērtību apmaina vietām ar masīva priekšpēdējo vērtību.
2N+2.solis. Izvada masīva A[N] elementus ekrānā.

6. uzdevums [*uzd6*] (10 punkti)
Niķis un Stiķis spēlēja "Cirku". Kad Stiķis uzvarēja, Niķis sāka strīdu, apgalvojot, ka Stiķis ir šmaucies. Par laimi Stiķis jau bija paredzējis, ka Niķis varētu uzsākt strīdu, tādēļ visu spēles laiku bija pierakstījis, cik punktus katru reizi uzmet viņš un cik Niķis.
Palīdzi atrisināt strīdu, sastādi programmu, kas izdrukā ekrānā tabulu, kas attēlo, uz kura lauciņa pēc katra gājiena atradās Niķa un Stiķa spēles kauliņi, un kura nosaka kurš patiesībā uzvarēja un uz kura lauciņa atrodas zaudētāja kauliņš spēles beigās.
Zināms, ka spēli sāka Niķis un katru reizi, kad tika uzmests 1 vai 6, spēlētājam bija papildus gājiens, kā arī, ja spēlētāja kauliņš apstājas uz 2. lauciņa, spēles kauliņš jāpārvieto uz 12. spēles lauciņu,

no 5. – jāpāriet uz 8.,
no 9. – jāpāriet uz 3.,
no 10. – jāpāriet uz 15.,
no 13. – jāpāriet uz 23.,
no 16. – jāpāriet uz 20.,
no 18. – jāpāriet uz 7.,
no 22. – jāpāriet uz 34.,
no 29. – jāpāriet uz 15.,
no 30. – jāpāriet uz 32.,
no 35. – jāpāriet uz 25.



Spēles sākumā kauliņi atrodas blakus spēles laukumam, un uzvar tas, kurš pirmais nokļūst uz 36. – pēdējā spēles lauciņa.
Niķa uzmestie punkti: 5; 4; 2; 5; 2; 1; 1; 2; 1; 1; 2.
Stiķa uzmestie punkti: 3; 4; 6; 2; 5; 3; 5; 4.

7. uzdevums [*uzd7*] (7 punkti)
Sastādīt programmu, kas izdrukā ekrānā naturālu skaitļu virknes elementus, ja zināms tās pirmais elements N un fakts, ka skaitļu virknes katrs pāra elements tiek iegūts iepriekšējo virknes elementu pareizinot ar 3, bet katrs nepāra elements, sākot ar trešo virknes elementu, tiek iegūts no iepriekšējā virknes elementa "nodzēšot" pēdējo ciparu. N ievada lietotājs, tā vērtība nedrīkst būt mazāka par 1000 un lielāka par 10000.
Piemēram: 1000; 3000; 300; 900; 90; 270; 27; 81; 8; 24; 2; 6.

8. uzdevums [*uzd8*] (7 punkti)
Alise saņēma SMS ar tekstu "skeintsaP vet tilut sisenta ilutsev." un izlasījusi to, neko nesaprata. Pēc brīža pie durvīm piezvanīja. Pastnieks bija atnesis viņai adresētu vēstuli, bet arī tās teksts bija nesaprotams. Meitene pārdomu pilna staigāja uz priekšu un atpakaļ pa istabu līdz ieraudzīja savu atspulgu spogulī un iesmējās, "Nu, protams, ...".
Sastādi programmu, kas palīdz Alisei izlasīt vēstules tekstu, kas ierakstīts teksta datnē *uzd8.in*. Rezultātu saglabā teksta datnē *uzd8.out*.

9. uzdevums [*uzd9*] (5 punkti)
Sastādīt programmu, kas izvada ekrānā NxN tabulas vērtības, kuras vērtības veidojas pēc piemērā redzamā principa. N ievada lietotājs, pie tam N vērtība var būt tikai nepāra skaitlis intervālā [1;21].
Piemēram, ja N=7, tabulas vērtības ir sekojošas:

1	*	*	*	*	*	**1**
1	**2**	*	*	*	**2**	**1**
1	**2**	**3**	*	**3**	**2**	**1**
1	**2**	**3**	**4**	**3**	**2**	**1**
1	**2**	**3**	*	**3**	**2**	**1**
1	**2**	*	*	*	**2**	**1**
1	*	*	*	*	*	**1**

10. uzdevums [*uzd10*] (8 punkti)
Maks un Morics plānoja kārtējo nedarbu. Lai neviens nevarētu uzzināt par viņu plāniem, visus pierakstus ko viņu veic, viņi šifrē. Teksta šifrēšanai viņi izmanto programmu.
Ar vārdiem apraksti, kādu šifrēšanas algoritmu izmanto viņu sastādītā šifrēšanas programma.
Pārveido šifrēšanas programmu par atšifrēšanas programmu. Atšifrējot datnē *uzd10.in* esošo tekstu uzzināsi, ko Maks un Morics plāno. Atšifrēto tekstu saglabā teksta datnē *uzd10.out*.

Šifrēšanas programmas C++ kods

```
#include <fstream.h>
#include <string.h>
#include <math.h>
void main(){
  ifstream dati1;
  ofstream dati2;
  dati1.open("teksts1.in",ios::in);
  dati2.open("teksts2.out",ios::out);
  char a[100][256];
  char s;
  int g[100][2];
  char x[15][15];
  int i=0,j=0,k,z,n,p;
  dati1.get(s);
  a[i][j]=s;
  g[i][0]=1;
  j=1;
  while(!dati1.eof()){
     dati1.get(s);
     if(int(s)!=10 && !dati1.eof()){
          g[i][0]++; a[i][j]=s; j++;
     }else{
          i++; j=0; g[i][0]=0;
     }
  }
  n=i;
  for(i=0; i<n; i++){
      p=1;
      while(pow(p,2)<g[i][0]) p++;
      g[i][1]=p;
  }
  for(k=0; k<n; k++){
    z=0;
    for(i=0; i<g[k][1]; i++){
        for(j=0; j<g[k][1]; j++){
           x[i][j]=a[k][z];
           z++;
       }
    }
    for(i=0; i<g[k][1]; i++){
        for(j=0; j<g[k][1]; j++){
           dati2<<x[j][i];
       }
    }
    dati2<<"\n";
  }
  dati1.close();
  dati2.close();
}
```