Liepājas Universitātes

Datorzinātņu olimpiādes 2015

skolēnu programmēšanas konkursa uzdevumi

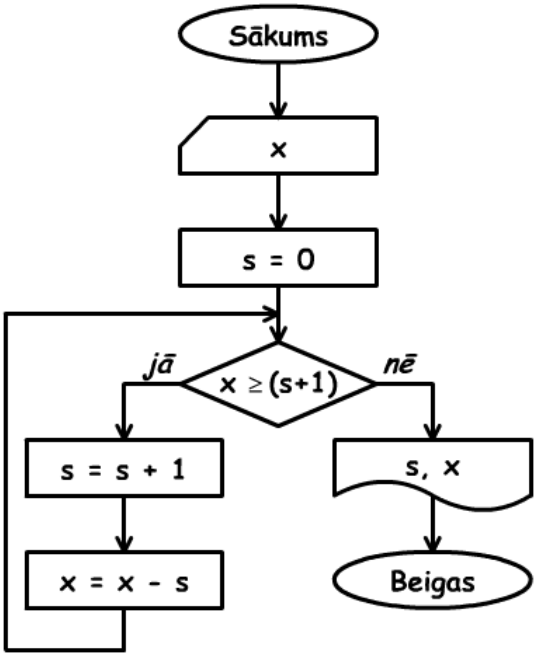
1. uzdevums [*uzd1*] (5 punkti)

Zināms, ka pēc dotās blokshēmas vai algoritma strukturētā apraksta var noteikt cik stāvu torni var uzbūvēt un cik klucīšu paliks pāri, ja to būvē, kā attēlots zīmējumā A. (x – klucīšu skaits, s – stāvu skaits)

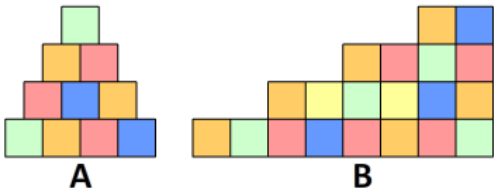
Sākums

1. **solis.** Izvēlas mainīgā *x* vērtību;
2. **solis.** Mainīgajam s piešķir sākuma vērtību 0;
3. **solis.** Ja *x* ≥ (s+1), tad pāriet uz 4. soli, citādi uz 6. soli;
4. **solis.** *s* vērtību palielina par 1;
5. **solis.** *x* vērtību samazina par *s*;
6. **solis.** Izvada *s* un *x* vērtības

Beigas



Sastādi programmu, kas ļautu noteikt cik stāvu torni var uzbūvēt un cik klucīšu paliks pāri, ja to būvē, kā attēlots zīmējumā *B*.



2. uzdevums [*uzd2*] (5 punkti)

Dots algoritma apraksts strukturētā teksta veidā. Uzzīmē dotā algoritma blokshēmu.

Sākums

1. **solis** izvēlas mainīgo n un m vērtības
2. **solis** mainīgajam k piešķir sākuma vērtību 0
3. **solis** ja *n* nav vienāds ar *m*, tad pāriet uz 4. soli, citādi uz 8. soli
4. **solis** ja *n* ir lielāks nekā *m*, tad pāriet uz 5. soli, citādi uz 6. soli
5. **solis** *n* vērtību samazina par *m*, pāriet uz 7. soli
6. **solis** *m* vērtību samazina par *n*, pāriet uz 7. soli
7. **solis** *k* vērtību palielina par 1, pāriet uz 3. soli
8. **solis** *k* vērtību palielina par 1
9. **solis** izvada *k* vērtību

Beigas

3. uzdevums [*uzd3*] (10 punkti)

Sastādīt programmu, kas lietotāja ievadīto heksadecimālo skaitli, kura garums nepārsniedz 256 simbolus, izvada ekrānā kā bināro skaitli.

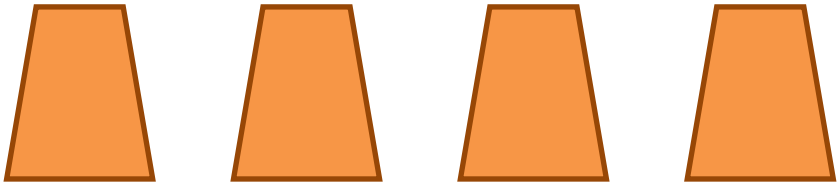
Piemēram,

- ja lietotājs ievada **F1**, programma izdrukā **1111 0001**;
- ja lietotājs ievada **ABC**, programma izdrukā **1010 1011 1100**.

4. uzdevums [*uzd3*] (10 punkti)
Sastādi programmu, kas ekrānā izvada dotās tabulas vērtības.

0	0	0	0	**1**	**1**	0	0	0	0
0	0	0	**1**	**1**	**1**	**1**	0	0	0
0	0	**1**	**1**	**1**	**1**	**1**	**1**	0	0
0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0
1	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
1	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0
0	0	**1**	**1**	**1**	**1**	**1**	**1**	0	0
0	0	0	**1**	**1**	**1**	**1**	0	0	0
0	0	0	0	**1**	**1**	0	0	0	0

5. uzdevums [*uzd4*] (10 punkti)
Rihardam ir četras necaurspīdīgas, plastmasa glāzes. Glāzes viņš novietoja uz galda kā attēlots zīmējumā un zem pirmās glāzes novietoja bumbiņu.



1. 2. 3. 4.

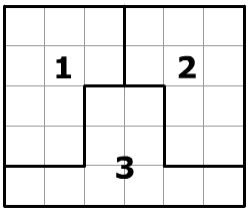
Pēc tam kad Rihards bija pārvietojis glāzes pēc sekojošā algoritma, bumbiņa atradās zem ceturtajā pozīcijā novietotās glāzes.

1) pirmajā pozīcijā novietoto glāzi apmainīja vietām ar trešajā pozīcijā novietoto glāzi,
2) otrajā pozīcijā novietoto glāzi apmainīja vietām ar ceturtajā pozīcijā novietoto glāzi,
3) otrajā pozīcijā novietoto glāzi apmainīja vietām ar trešajā pozīcijā novietoto glāzi,
4) pirmajā pozīcijā novietoto glāzi apmainīja vietām ar ceturtajā pozīcijā novietoto glāzi,
5) otrajā pozīcijā novietoto glāzi apmainīja vietām ar ceturtajā pozīcijā novietoto glāzi.

Sastādīt programmu, kas nosaka, zem kuras pozīcijas glāzes atradīsies bumbiņa, ja programmas lietotājs sākotnēji var ievadīt zem kuras glāzes novietot bumbiņu un cik reizes pielietot Riharda izdomāto sajaukšanas algoritmu.

6. uzdevums [*uzd6*] (15 punkti)
Jebkuram naturālam skaitlim var aprēķināt ciparu summu. Ja iegūtā summa nav viencipara skaitlis, arī tai var aprēķināt ciparu summu. Šādā veidā iegūtu viencipara skaitli sauksim par supersummu.
Dana, Alise un Roberts nolēma uzspēlēt spēli "Superskaitlis". Par spēles uzvarētāju kļūs tas, kurš izdomās piecciparu skaitli, kurā nav divu vienādu ciparu un skaitļa supersumma būs vislielākā.
Sastādīt programmu, kas prasa ievadīt trīs piecciparu skaitļus (Danas, Alises un Roberta) un ekrānā izvada uzvarētāja vārdu un tā iedomāto skaitli un skaitļa supersumma.
Piemēram, ja Dana ievada skaitli 85674, Alise – 56789, bet Roberts – 94347, tad programma paziņo, ka "Uzvarētāja Alise ar skaitļa 56789 supersummu 8".

7. uzdevums [*uzd7*] (10 punkti)



Kārlis topošo arhitektu konkursam veido maketu. Tā kā materiāls, kuru viņš plāno izmantot maketa sienu izgatavošanai nav lēts, viņam iepriekš jāveic plānošanas un aprēķina darbi.
Kārlis uz rūtiņu papīra uzzīmējis ēkas starpsienu plānojumu. Zināms, ka vienas rūtiņas izmērs dzīvē būs 10 cm x 10 cm, bet sienu augstums būs 20 cm.

Sastādi programmu, kas palīdz Kārlim aprēķināt (izdrukā failā *uzd7.out*), cik kvadrātcentimetrus materiāla nepieciešams iegādāties, lai izgatavotu iecerēto maketu, ja faila *uzd7.in* pirmajā rindā doti divi veseli skaitļi *N* un *M* ($1 \le N, M \le 100$) - ēkas izmērs rūtiņās, bet katrā no sekojošajām faila N rindiņām doti M skaitļi, kas apzīmē katras rūtiņas piederību telpai plānā.

Piemēram,
ievaddati *uzd7.in*

```
6 5
1 1 1 2 2 2
1 1 1 2 2 2
1 1 3 3 2 2
1 1 3 3 2 2
3 3 3 3 3 3
```

izvaddati *uzd7.out*

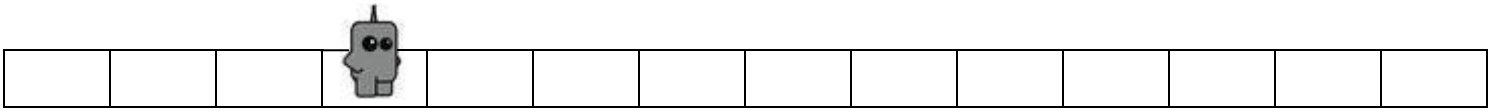
```
6800
```

8. uzdevums [*uzd8*] (15 punkti)

Orientēšanās sacensībās piedalījās desmit komandas, katru komandu pārstāvēja pieci dalībnieki. Failā *uzd8.in* doti sacensību dalībnieku rezultāti formā: N H M S, kur N – komandas numurs, kuru pārstāv dalībnieks, H – stundas, M – minūtes, S – sekundes, ko sacensības dalībnieks pavadīja trasē. Sastādīt programmu, kas failā *uzd8.out* izdrukā sacensību rezultātus formā: N Hk Mk Sk (kur N – komandas numurs, Hk – stundas, Mk – minūtes, Sk – sekundes, ko konkrētās komandas dalībnieki kopā pavadījuši trasē), sakārtotus augošā secībā pēc komandu dalībnieku trasē pavadītā kopīgā laika.

9. uzdevums [*uzd9*] (10 punkti)

Robots spēj pārvietoties pa galīgu lenti, kura sadalīta rūtiņā. Rūtiņas sanumurētas no -100 līdz 100. Robots sākumā atrodas rūtiņā 0. Robots spēj izpildīt tikai divas komandas, paiet trīs soļus uz priekšu (P3) un paiet divus soļus atpakaļ (A2). Piemēram, izpildot programmu "P3 A2 A2 P3 P3 A2 A2 P3 P3", robots apstāsies rūtiņā 7.



Sastādīt programmu, kas nolasa failu *uzd9.in*, kura vienīgajā rindā dota robota izpildāmā programma, komandas atdalītas tieši ar vienu atstarpi, un izdrukā failā *uzd9.out* vienu skaitli – rūtiņas numuru, kurā atrodas robots pēc programmas izpildes.

Piemēram,
ievaddati *uzd9.in*

```
P3 A2 A2 P3 P3 A2 A2 P3 P3
```

izvaddati *uzd9.out*

```
7
```

10. uzdevums [*uzd10*] (10 punkti)

Harijam skolas kursā "Programmēšanas pamati" jāapgūst tēma "Datu šifrēšana". Skolotājs viņam ir iedevis programmas piemēru, kas realizē datu šifrēšanu, izmantojot Džona Falkonera (John Falconer) 1685. gadā publicēto "Divu burtu alfabētu".

Piemēram,

ja šifrējamais teksts ir: **LIELS NAMS**,

tad šifrētā simbolu virknes ir: **ABABAABBBAAABAAABABABAAAB ABBAAAAAAAABABBBAAAB**.

Palīdzi Harijam uzrakstīt programmu, kas atšifrē tekstu un ieraksta to failā *uzd10.out*, kas iepriekš ticis nošifrēts izmantojot "Divu burtu alfabētu" un dots teksta failā *uzd10.in*.

"Divu burtu alfabēts" algoritma realizācija (Pascal kods)

```
program DivuBurtuAlfabets;
var dati1,dati2:text;
    s:char;
begin
assign(dati1,'uzd10.in');
reset(dati1);
assign(dati2,'uzd10.out');
rewrite(dati2);
while (not EOF(dati1)) do begin
   read(dati1,s);
        case s of
                'A': write(dati2,'AAAAA');
        'B': write(dati2,'AAAAB');
        'C': write(dati2,'AAABA');
        'D': write(dati2,'AAABB');
        'E': write(dati2,'AABAA');
        'F': write(dati2,'AABAB');
        'G': write(dati2,'AABBA');
        'H': write(dati2,'AABBB');
        'I': write(dati2,'ABBBA');
        'K': write(dati2,'ABAAB');
        'L': write(dati2,'ABABA');
        'M': write(dati2,'ABABB');
        'N': write(dati2,'ABBAA');
        'O': write(dati2,'ABBAB');
        'P': write(dati2,'ABBBA');
        'R': write(dati2,'BAAAA');
        'S': write(dati2,'BAAAB');
        'T': write(dati2,'BAABA');
        'V': write(dati2,'BAABB');
        'Z': write(dati2,'BABBB');
                else write(dati2,s);
        end;
end;
close(dati1);
close(dati2);
end.
```

"Divu burtu alfabēta" algoritma realizācija (C++ kods)

```
#include <fstream>
using namespace std;
int main(){
  ifstream dati1;
  ofstream dati2;
  dati1.open("uzd10.in",ios::in);
  dati2.open("uzd10.out",ios::out);
  char s;
  dati1.get(s);
  while(!dati1.eof()){
     switch(s){
        case 'A': dati2<<"AAAAA"; break;
        case 'B': dati2<<"AAAAB"; break;
        case 'C': dati2<<"AAABA"; break;
        case 'D': dati2<<"AAABB"; break;
        case 'E': dati2<<"AABAA"; break;
        case 'F': dati2<<"AABAB"; break;
        case 'G': dati2<<"AABBA"; break;
        case 'H': dati2<<"AABBB"; break;
        case 'I': dati2<<"ABBBA"; break;
        case 'K': dati2<<"ABAAB"; break;
        case 'L': dati2<<"ABABA"; break;
        case 'M': dati2<<"ABABB"; break;
        case 'N': dati2<<"ABBAA"; break;
        case 'O': dati2<<"ABBAB"; break;
        case 'P': dati2<<"ABBBA"; break;
        case 'R': dati2<<"BAAAA"; break;
        case 'S': dati2<<"BAAAB"; break;
        case 'T': dati2<<"BAABA"; break;
        case 'V': dati2<<"BAABB"; break;
        case 'Z': dati2<<"BABBB"; break;
        default: dati2<<s;
     }
     dati1.get(s);
  }
  dati1.close();
  dati2.close();
}
```

"Divu burtu alfabēta" algoritma realizācija
(Java kods)

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;

public class DivuBurtuAlfabets {
	public static void main(String[] args) {
		try{
			char s;
			int x;
			BufferedReader dati1=new BufferedReader(new FileReader("uzd10.in"));
			BufferedWriter dati2=new BufferedWriter(new FileWriter("uzd10.out"));
			x=dati1.read();
			s=(char)x;
			while(x!=-1){
				switch(s){
					case 'A': dati2.write("AAAAA"); break;
					case 'B': dati2.write("AAAAB"); break;
					case 'C': dati2.write("AAABA"); break;
					case 'D': dati2.write("AAABB"); break;
					case 'E': dati2.write("AABAA"); break;
					case 'F': dati2.write("AABAB"); break;
					case 'G': dati2.write("AABBA"); break;
					case 'H': dati2.write("AABBB"); break;
					case 'I': dati2.write("ABBBA"); break;
					case 'K': dati2.write("ABAAB"); break;
					case 'L': dati2.write("ABABA"); break;
					case 'M': dati2.write("ABABB"); break;
					case 'N': dati2.write("ABBAA"); break;
					case 'O': dati2.write("ABBAB"); break;
					case 'P': dati2.write("ABBBA"); break;
					case 'R': dati2.write("BAAAA"); break;
					case 'S': dati2.write("BAAAB"); break;
					case 'T': dati2.write("BAABA"); break;
					case 'V': dati2.write("BAABB"); break;
					case 'Z': dati2.write("BABBB"); break;
					default: dati2.write(s);
				}
				x=dati1.read();
				s=(char)x;
			}
			dati1.close();
			dati2.close();
		} catch (IOException e){}
	}
}
```