Liepājas Universitātes
Datorzinātņu olimpiādes 2019
programmēšanas konkursa uzdevumi studentiem

1. uzdevums [uzd1] (5 punkti)
Sastādi programmu, kas nosaka, vai caur taisnstūra formas caurumu ar izmēriem **N**x**M** var izbāzt ķieģeli ar izmēriem **A**x**B**x**C** un cilindru, kura augstums ir **H** un pamata rādiuss ir **R**? Visi izmēri doti centimetros. Visu mainīgo vērtības ir naturāli skaitļi robežās no 1 līdz 1000, kurus ievada lietotājs. Programmai jānodrošina, ka lietotājs nevar ievadīt nekorektas ievadvērtības un ekrānā jāizdrukā divi teikumi – "VAR/NEVAR iebāzt ķieģeli caurumā", "VAR/NEVAR iebāzt cilindru caurumā".

2. uzdevums [uzd2] (5 punkti; vai 7 punkti, ja uzdevums realizēts, neizmantojot ciklus)
Sastādi programmu, kas nosaka dotās naturālo skaitļu virknes N-tā elementa vērtību.

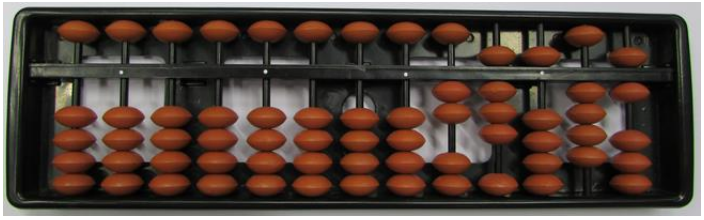
1, 2, 3, 6, 5, 10, 7, 14 …

N – naturāls skaitlis robežās no 1 līdz 10000, kuru ievada lietotājs. Programmai jānodrošina, ka lietotājs nevar ievadīt nekorektu N vērtību.

3. uzdevums [uzd3] (10 punkti)
Atrodi sakarības starp skaitļiem dotās n x n tabulas rūtiņās. Sastādi programmu, kas, izmantojot atrastās sakarības, izvada dotos skaitļus ekrānā tabulas formā (bez atdalošajām līnijām). Mainīgā n vērtību ievada lietotājs. Lieluma n vērtība var būt tikai vesels skaitlis intervālā [3;20]. Piemēram, ja n = 9, tad tabula tiek aizpildīta ar šādām vērtībām:

1	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0
1	2	1	0	0	0	0	0	0
1	3	3	1	0	0	0	0	0
1	4	6	4	1	0	0	0	0
1	5	10	10	5	1	0	0	0
1	6	15	20	15	6	1	0	0
1	7	21	35	35	21	7	1	0
1	8	28	56	70	56	28	8	1

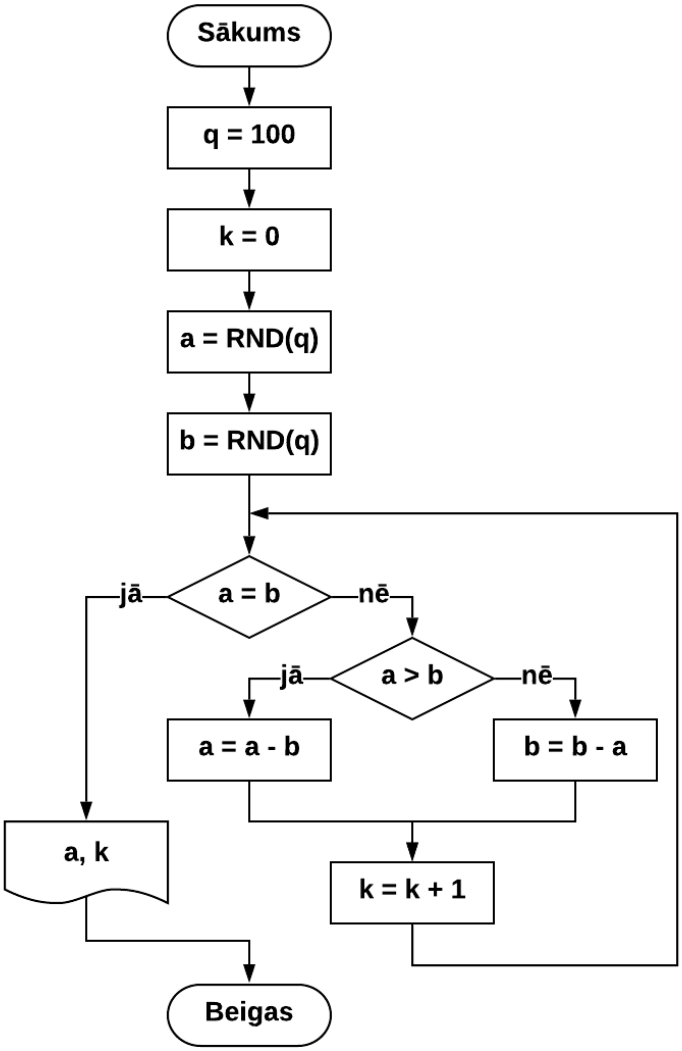
4. uzdevums [uzd4] (10 punkti)
Attēlā redzams Japāņu izcelsmes abaks – sorobans, uz kura atlikts skaitlis 28546. Sastādi programmu, kas lietotāja ievadīto naturālo skaitli N attēlo kā uz sorobana. N – naturāls skaitlis, par kuru zināms, ka tā garums nepārsniedz 30 ciparus.



N=28546

```
0 | | 0 |
| 0 0 | 0
---------
0 0 | 0 0
0 0 0 0 |
| 0 0 0 0
0 | 0 0 0
0 0 0 | 0
```

5. uzdevums [uzd5] (5 punkti)
Sastādi programmu pēc dotās algoritma blokshēmas. Ar RND(q) blokshēmā apzīmēta funkcija, kuras vērtība ir nejaušs vesels skaitlis robežās [1;q].



6. uzdevums [uzd6] (10 punkti)
Aleksis uz lielā brāļa rakstāmgalda ieraudzīja atvērtu pierakstu kladi un pamanīja tajā algoritma aprakstu, kurš paredzēts ciparu virknes apstrādei. Viņš nolēma to pielietot ciparu virknei 112233445566778899 un rezultātā ieguva 36.

Sākums
1. solis – Izvēlas ciparu virkni.
2. solis – No ciparu virknes izsvītro visus tos ciparus, kuri atrodas pāra pozīcijās.
3. solis – Atlikušās ciparu virknes beigās pieraksta izsvītroto ciparu summu.
4. solis – No ciparu virknes izsvītro visus tos ciparus, kuri atrodas nepāra pozīcijās.
5. solis – Atlikušās ciparu virknes beigās pieraksta izsvītroto ciparu summu.
6. solis – Ja ciparu virknē atlikuši vairāk kā divi cipari, atgriezies pie 2. soļa, citādi pāriet uz 7. soli.
7. solis – Rezultāts: pēdējie divi ciparu virknes cipari.

Beigas

Sastādi programmu, kas pielieto doto algoritmu no datnes *uzd6.in* nolasītai ciparu virknei (ciparu virknes garums nepārsniedz 250 ciparus) un rezultātu izdrukā datnē *uzd6.out*. Piezīme, ja, izsvītrojot ciparus, atlikušo ciparu virkne sākas ar 0, nulle ciparu virknes sākumā jāatmet.

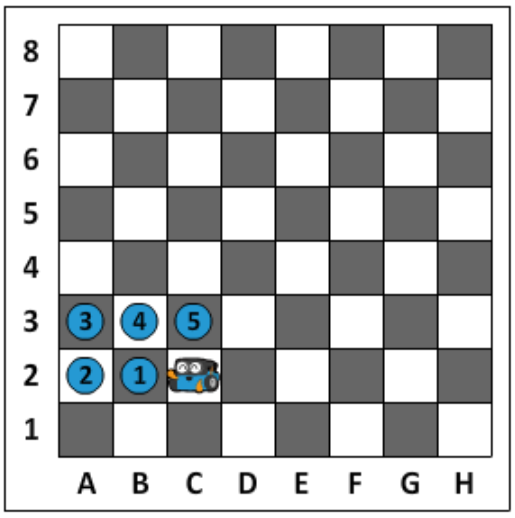
Piemēram,

ievaddati *uzd6.in*	izvaddati *uzd6.out*
11020234567	49

7. uzdevums [uzd7] (15 punkti)

Robots pārvietojas pa šaha lauciņa rūtiņām un atstāj aiz sevis bumbiņas, kas numurētas ar naturāliem skaitļiem, sākot ar 1. Robots pārvietojas pēc šādiem noteikumiem:

- sākotnēji robots tiek novietots uz tukša šaha dēlīša brīvi izvēlētā rūtiņā virzienā ar skatu pret lauciņu apzīmējošo burtu;
- ja pa labi no robota esošajā rūtiņā nav bumbiņas, robots pagriežas un pārvietojas uz to;
- ja pa labi no robota esošajā rūtiņā jau atrodas bumbiņa, robots pārvietojas vienu rūtiņu uz priekšu;
- robots beidz pārvietoties, ja nevar vairs izdarīt gājienu vai visās šaha lauciņa rūtiņās izvietotas bumbiņas.



Sastādi programmu, kas nolasa datnē *uzd7.in* doto šaha lauciņa rūtiņas koordināti, no kuras sāk pārvietoties robots, un izdrukā datnes *uzd7.out* 1. rindiņā, lauciņa koordinātes, kurā robots apstāsies, un 2. rindiņā, kāds būs pēdējās robota atstātās bumbiņas kārtas numurs.

Piemēram,

ievaddati *uzd7.in*	izvaddati *uzd7.out*
B2	A1 8

8. uzdevums [uzd8] (10 punkti, ja programmas realizācijā netiek izmantots masīvs;
-5 punkti, ja programmas realizācijā izmantots masīvs)

Dažos sporta veidos sportistu sniegumu vērtē vairāki tiesneši, pēc tam no visiem tiesnešu vērtējumiem tiek atmests augstākais un zemākais vērtējums, bet no atlikušajiem vērtējumiem tiek aprēķināts vidējais vērtējums. Ja augstāko vai zemāko vērtējumu ielikuši vairāki tiesneši, tad atmesti tiek visi augstākie vai zemākie vērtējumi.

Sastādi programmu, kas, neizmantojot masīvu, nolasa datnē *uzd8.in* dotos tiesnešu vērtējumus (ne vairāk kā 100 veseli skaitļi, katrs vērtējums dots savā rindā) un aprēķina (ko?), un izdrukā datnē *uzd8.out* aprēķināto vidējo vērtējumu. (Vidējā vērtējuma precizitāte atkarīga no izvēlētās programmēšanas valodas.)

Piemēram,

ievaddati *uzd8.in*	izvaddati *uzd8.out*
5 7 3 2 4 7 6	5.4

9. uzdevums [uzd9] (15 punkti, ja programmas realizācijā netiek izmantots 2-dimensiju masīvs;
-5 punkti, ja programmas realizācijā izmantots 2-dimensiju masīvs)

Aleksi jau atkal ieinteresēja vecākā brāļa skolas pierakstu klade. Tajā bija rakstīts:

Viženera šifrs

Šis šifrs ir viena no senākajām (1586.g.) un pazīstamākajām daudzalfabētu šifrēšanas metodēm. Viženera šifra pamatā ir kvadrātveida tabula, kuras pirmā rinda ir alfabēts, otrā – tas pats alfabēts, tikai nobīdīts par vienu pozīciju, trešā – par 2 pozīcijām utt.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	**O**	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	**G**	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Šifrēšana notiek sekojoši: izvēlas patvaļīga garuma atslēgu, piemēram, vārdu OLA, un pieraksta to zem šifrējamā teksta SKOLAS SOMA.

Šifrēšana notiek pa vienam burtam. Ņem šifrējamā teksta vienu burtu un atrod to tabulas kreisajā kolonnā, tad ņem to atslēgas burtu, kas ir pierakstīts zem dotā teksta burta, un atrod to tabulas augšējā rindiņā. Kriptogrammas burts ir nolasāms fiksētās kolonnas un rindiņas krustojumā (līdzīgi kā reizrēķina tabulā).

Šifrējot tekstu "SKOLAS SOMA" ar atslēgu "OLA", iegūst kriptogrammu "GVOZLS GZMO". No šī piemēra ir redzams, ka viens un tas pats burts A dotajā tekstā var tikts aizstāts ar dažādiem burtiem kriptogrammā L un O. Tāpat divi atšķirīgi burti O un A var tikt aizstāti ar vienu un to pašu burtu O.

Sastādi programmu, kas nolasa datnē *uzd9.in* doto atslēgu (dota datnes 1. rindā) un tekstu (teksts sākot no datnes 2. rindas) un neizmantojot 2-dimensiju masīvu nošifrē to izmantojot Viženera šifru. Kriptogrammu izdrukā datnē *uzd9.out*. Gan atslēga, gan teksts pierakstīti ar latīņu alfabēta lielajiem burtiem.

Piemēram,

ievaddati *uzd9.in*	izvaddati *uzd9.out*
OLA	GVOZLS GZMO
SKOLAS SOMA	

10. uzdevums [uzd10] (10 punkti)
Kurts skolā apgūst mācību priekšmetu "Programmēšanas pamati". Klasē, strādājot kopā ar skolotāju, Kurts ir uzrakstījis programmu, kas aprēķina divu lielu veselu skaitļu summu.
Pārveido Kurta uzrakstīto programmu, tā, lai tā nolasa no faila uzd10.in divus lielus veselus skaitļus, katrs skaitlis dots savā rindā, un aprēķina šo skaitļu starpību. Aprēķināto skaitļu starpību jāizvada failā uzd10.out.

C++ kods divu skaitļu summas aprēķinam

```
#include <fstream>
using namespace std;
int main(){
	ifstream dati1;
	ofstream dati2;
	dati1.open("uzd10.in",ios::in);
	dati2.open("uzd10.out",ios::out);
	string sk1, sk2;
	int n1, n2, n3;
	int i;
	dati1>>sk1;
	dati1>>sk2;
	n1=sk1.length();
	n2=sk2.length();
	if(n1>n2) n3=n1; else n3=n2;
	int x1[n3+1]={0};
	int x2[n3+1]={0};
	int x3[n3+1]={0};
	for(i=0; i<n1; i++) x1[i]=sk1[n1-i-1]-'0';
	for(i=0; i<n2; i++) x2[i]=sk2[n2-i-1]-'0';
	for(i=0; i<n3; i++){
		x3[i+1]=(x1[i]+x2[i]+x3[i])/10;
		x3[i]=(x1[i]+x2[i]+x3[i])%10;
	}
	if(x3[n3]!=0) dati2<<x3[i];
	for(i=n3-1; i>=0; i--) dati2<<x3[i];
	dati1.close();
	dati2.close();
}
```

Python3 kods divu skaitļu summas aprēķinam

```
dati1 = open("uzd10.in", "r")
dati2 = open("uzd10.out", "w")
n1=0; n2=0; n3=0
i=0
sk1 = dati1.readline()
sk2 = dati1.readline()
n1=len(sk1)-1
n2=len(sk2)
if n1>n2: n3=n1
else: n3=n2
x1 = [0] * (n3+1)
x2 = [0] * (n3+1)
x3 = [0] * (n3+1)
for i in range(n1): x1[i]=int(sk1[n1-i-1])
for i in range(n2): x2[i]=int(sk2[n2-i-1])
for i in range(n3):
      x3[i+1]=int((x1[i]+x2[i]+x3[i])/10)
      x3[i]=(x1[i]+x2[i]+x3[i])%10
if x3[n3] != 0: dati2.write(str(int(x3[i])))
for i in range(n3-1, -1, -1): dati2.write(str(int(x3[i])))
dati1.close()
dati2.close()
```

Java kods divu skaitļu summas aprēķinam

```
import java.io.*;
public class Sum {
	public static void main(String[] args) {
	try{
		BufferedReader dati1=new BufferedReader(new FileReader("uzd10.in"));
		BufferedWriter dati2=new BufferedWriter(new FileWriter("uzd10.out"));
		String sk1, sk2;
		int n1, n2, n3;
		int i;
		sk1=dati1.readLine();
		sk2=dati1.readLine();
		n1=sk1.length();
		n2=sk2.length();
		n3=Math.max(n1,n2);
		int[] x1=new int[n3+1];
		int[] x2=new int[n3+1];
		int[] x3=new int[n3+1];
		for(i=0; i<n1; i++) x1[i]=sk1.charAt(n1-i-1)-'0';
		for(i=0; i<n2; i++) x2[i]=sk2.charAt(n2-i-1)-'0';
		for(i=0; i<n3; i++){
			x3[i+1]=(x1[i]+x2[i]+x3[i])/10;
			x3[i]=(x1[i]+x2[i]+x3[i])%10;
		}
		if(x3[n3]!=0) dati2.write(x3[i]+"");
		for(i=n3-1; i>=0; i--) dati2.write(x3[i]+"");
		dati1.close();
		dati2.close();
	} catch (IOException e){System.out.print("Nevar atrast failu!");}
	}
}
```

PHP kods divu skaitļu summas aprēķinam

```
<?php
	$dati1=fopen('uzd10.in','r');
	$dati2=fopen('uzd10.out','w');
	$sk1=fgets($dati1);
	$sk2=fgets($dati1);
	$sk1=trim($sk1);
	$sk2=trim($sk2);
	$sk1=strrev($sk1);
	$sk2=strrev($sk2);
	$n1=strlen($sk1);
	$n2=strlen($sk2);
	if($n1>$n2) $n3=$n1; else $n3=$n2;
	$x1=array_fill(0,$n3+1,"0");
	$x2=array_fill(0,$n3+1,"0");
	$x3=array_fill(0,$n3+1,"0");
	for($i=0; $i<$n1; $i++) $x1[$i]=$sk1[$i];
	for($i=0; $i<$n2; $i++) $x2[$i]=$sk2[$i];
	for($i=0; $i<$n3; $i++) {
		$x3[$i+1]=(int)((intval($x1[$i])+intval($x2[$i])+intval($x3[$i]))/10);
		$x3[$i]=(intval($x1[$i])+intval($x2[$i])+intval($x3[$i]))%10;
	}
	if(intval($x3[$n3])!=0) fwrite($dati2,$x3[$n3]);
	for($i=$n3-1; $i>=0; $i--) fwrite($dati2,$x3[$i]);
	fclose($dati1);
	fclose($dati2);
?>
```