



**ДРЖАВНО  
ТАКМИЧЕЊЕ ИЗ  
РАЧУНАРСТВА ЗА  
УЧЕНИКЕ ОСНОВНИХ  
ШКОЛА**

**БИЛТЕН**

-----  
**ДРЖАВНА КОМИСИЈА  
ЗА ТАКМИЧЕЊЕ  
УЧЕНИКА ОСНОВНИХ  
ШКОЛА**

2017/18

**15.04. 2018.**



ДМС

## Садржај:

- **Извештај Државне комисије  
након такмичења**
- **Задаци и решења**



# ДРУШТВО МАТЕМАТИЧАРА СРБИЈЕ

**Кнез Михаилава 35/IV  
11000 Београд  
Србија**

## Извештај Државне комисије након такмичења из рачунарства Београд – 15. април 2018 .

Државно такмичење из рачунарства за ученике V-VIII разреда основних школа у организацији Друштва математичара Србије (ДМС) је одржано 15. априла 2018. у Рачунарској гимназији у Београду. На такмичењу, ученици основних школа подељени су у две категорије. Прву категорију чине ученици V и VI разреда, а другу категорију ученици VII и VIII разреда.

Иако је по Правилнику (члан 6.3 Правилника о такмичењима ученика основних и средњих школа из информатике у организацији Министарства просвете Републике Србије и Друштва математичара Србије) Комисија обавезна да на такмичење позове 120 ученика, у циљу популаризације програмирања, одлучено је да тај број повећамо скоро два пута (члан 10.2 и 10.4).

Државно такмичење је окупило 202 најуспешнија ученика са окружних такмичења (у школској 2017/18 години)

Најуспешнији такмичари на Државном такмичењу (освајачи 1. награде):

1. категорија

Презиме и име	Разред	Школа	Град	Поени (макс.100)
Гојковић Филип	6	Павле Савић	Београд	100
Николић Јован	6	Кадинача	Лозница	100
Бојковић Филип	6	Бранко Радичевић	Београд	97
Милићев Јован	6	Уједињене нације	Београд	95
Ристић Тадеј	6	Бановић Страхиња	Београд	93

## 2. категорија

Презиме и име	Разред	Школа	Град	Поени (макс.100)
Бенгин Јован	8	Математичка гимназија	Београд	100
Вукелић Матеја	7	Математичка гимназија	Београд	100
Тољић Вељко	8	Вожд Карађорђе	Лесковац	98
Пурић Милош	8	Математичка гимназија	Београд	88
Зелић Немања	8	Математичка гимназија	Београд	77
Ђоковић Лука	8	Краљ Петар	Ниш	75
Ђорђевић Алекса	8	Стефан Немања	Ниш	73

Такмичење је трајало 150 минута. Такмичари су решавали по четири проблемска задатка и кодирани их у програмским језицима C, C++, C#, Pascal, Python, Java на Windows платформи. Коришћена су следећа развојна окружења: Dev-C++, Code::Blocks, MS Visual Studio, Free Pascal, IDLE, Eclipse

Аутоматска евалуација решења је обављена на [retija.org](http://retija.org) (систему за управљање такмичењем).

Најбољих 45 такмичара је изборило учешће на 12. Српској информатичкој олимпијади на којој ће бити изабрана репрезентација која ће ове године заступати Републику Србију на међународним информатичким такмичењима и Европској јуниорској олимпијади из програмирања у Инополису у Русији од 26. јула до 3. августа.

## Државно такмичење ученика основних школа из рачунарства 2018. године 5. и 6. разред

1. [pločice] Правоугаону терасу димензија  $d \times s$  центиметара квадратних треба поплочати коришћењем плочица квадратног облика странице  $p$  центиметара, које се постављају тако да су им странице паралелне страницама терасе. Написати програм којим се одређује колико се плочица мора сећи ради поплочавања, као и површину дела терасе који заузимају сечене плочице. Од сваке сечене плочице користи се један део, а други одбацује.

Улаз:

$p$  - страница плочице у см ( $10 \leq p \leq 50$ )

$d$  - дужина просторије у см ( $200 \leq d \leq 10000$ )

$s$  - ширина просторије у см ( $200 \leq s \leq 10000$ )

Израз: број плочица које се морају сећи и површина коју покривају сечене плочице.

Пример:

Улаз	Израз
20	29
310	5700
270	

### Решење:

```
#include <iostream>
using namespace std;

int main() {
    // dimenzija plocice i duzina i sirina prostorije
    int p, d, s;
    cin >> p >> d >> s;

    int dfloor = d / p, dceil = (d + p - 1) / p;
    int sfloor = s / p, sceil = (s + p - 1) / p;
    // dceil * sceil - minimalni broj plocica koje pokrivaju oblast
    // dfloor * sfloor - maksimalni broj plocica koje su sadrzane u oblasti
    int brojSecenihPlocica = (dceil * sceil) - (dfloor * sfloor);
    int povrsinaPokrivenaSecenimPlocicama = (d*s) - (dfloor*p*sfloor*p);
    cout << brojSecenihPlocica << endl;
    cout << povrsinaPokrivenaSecenimPlocicama << endl;
}
```

2. [kupovina] Познате су све цене предмета који се продају у једној продавници. Купац има на располагању одређени износ динара и жели да купи што скупле предмете. Редом узима предмете почев од најскупљег, док има новца. Ако нема новца за најскупљи, узима најскупљи за који има новца. *Напомена:* ова стратегија не гарантује да ће предмети које купи бити укупно највеће могуће вредности (нпр. ако има 5 динара и ако су цене предмета 4, 3 и 2 динара, он ће купити предмет само предмет од 4 динара, а могао би да купи предмете од 3 и 2 динара).

Напиши програм који одређује колико новца купцу преостаје након куповине на описани начин

У првој линији стандардног улаза налази се износ новца (цео број) који има купац, у другој број предмета,  $N$  (природни број мањи од 50000), а затим се у наредној линији уносе редом цене

предмета, раздвојене размацама. На стандарни излаз исписати преостали износ новца, након куповине на описани начин.

### Примери

Улаз	Израз	Улаз	Израз
1250	10	10000	0
5		6	
1010 357 725 1125 115		3010 3005 5725 1265 2075 385	

### Решење:

```
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

int main() {
    // učitavamo iznos kojim kupac raspolaze
    int iznos;
    cin >> iznos;
    // učitavamo cene svih predmeta
    int n;
    cin >> n;
    vector<int> cene(n);
    for (int i = 0; i < n; i++)
        cin >> cene[i]

    // sortiramo cene nerastuce
    sort(cene.begin(), cene.end(), greater<int>());

    // kupujemo najvrednije predmete koje mozemo kupiti sve dok imamo novca
    for (int i = 0; iznos > 0 && i < cene.size(); i++)
        if (iznos >= cene[i])
            iznos -= predmeti[i].cena;

    // ispisujemo preostali iznos novca nakon kupovine
    cout << iznos << endl;

    return 0;
}
```

3. **[utovar]** У транспортни брод преносе се редом пакети задатих маса колицима дате носивости (она је већа или једнака маси сваког пакета). При томе пакете увек преносимо у целости. Када неки пакет не може стати у колица због тренутног прекорачења носивости колица, та колица превозимо до брода, и почињемо пуњење нових колица. Написати програм који приказује редом, за свака колица, број пакета и укупну масу пакета пренетих њима.

У првој линији стандардног улаза налази се носивост колица  $N$  (природан број  $10 \leq N \leq 500$ ). У свакој наредној линија налази се маса пакета  $m_i$  (природан број  $1 \leq m_i \leq N$ ) који треба пренети. Улаз се завршава линијом у којој се налази број 0 (њих има највише 50000). На стандардном излазу се прво, за свака колица, у по једној линији, налази број пакета на колицима и укупна маса пакета стављених на њих (раздвојени размаком).

## Пример

Улаз

30  
5  
10  
8  
12  
3  
5  
15  
13  
5  
15  
2  
4  
22  
9  
0

Излаз

3 23  
3 20  
2 28  
4 26  
1 22  
1 9

## Решење:

```
#include <iostream>
using namespace std;

int main() {
    // nosivost paketa
    int Nosivost;
    cin >> Nosivost;

    // podaci o tekucem stanju na kolicima
    int masaNaKolicima = 0, brojPaketaNaKolicima = 0;

    // indikator da li se stiglo do kraja
    bool kraj = false;
    while(!kraj) {
        // ucitavamo paket
        int masaPaketa;
        cin >> masaPaketa;
        // proveravamo da li je u pitanju oznaka kraja
        if (masaPaketa == 0)
            kraj = true;

        // ako se tekuca kolica vise ne mogu popuniti (stiglo se do kraja ili
        // ucitani paket ne staje na njih)
```

```

if (kraj || masaNaKolicima + masaPaketa > Nosivost) {
    // ispisujemo broj i masu paketa na njima
    cout << brojPaketaNaKolicima << " " << masaNaKolicima << endl;
    // ako nismo dosli do kraja, zapocinjemo nova kolica
    if (!kraj) {
        brojPaketaNaKolicima = 0;
        masaNaKolicima = 0;
    }
}
// ako nismo dosli do kraja, ucitani predmet postavljamo
// na kolica (stara, ako staje na njih, tj. nova ako ne staje)
if (!kraj) {
    masaNaKolicima += masaPaketa;
    brojPaketaNaKolicima++;
}
}
return 0;
}

```

4. **[knjige]** Јованка жели да пребаци књиге са једне на другу полицу. На првој полици се налази  $N$  књига и позната је ширина сваке од њих у милиметрима. Она рукама хвата одређени низ књига са прве полице које се налазе једна до друге и све заједно их одједном пребациује на другу полицу, при чему жели да пребаци књиге тако да јој попуне другу полицу од ивице до ивице.

У првој линији стандардног улаза налази се природни број  $z$  (такав да је  $1 \leq z \leq 10^6$ ) који представља ширину друге полице. У другој се налази број књига на првој полици  $n$  ( $2 \leq n \leq 5 \cdot 10^5$ ), а у трећој ширине књига на првој полици (позитивни природни бројеви мањи од 100), раздвојени размаком. Исписати све редне бројеве књига од којих Јованка може да започне пребацивање тако да друга полица буде попуњена од ивице до ивице (књиге на првој полици се броје од нуле), поређане растуће.

*Пример*

*Улаз:*

125  
10  
35 40 25 50 50 50 25 35 15 35  
50+25+35+15

*Изаз:*

2  
4  
5

*Објашњење:*

Ако крене књиге број 2, пренеће 25+50+50  
Ако крене од књиге број 4, пренеће 50+50+25  
Ако крене од књиге број 5, пренеће

**Решење:**

```

#include <iostream>

using namespace std;

int main() {
    // ubrzavamo ucitavanje
    ios_base::sync_with_stdio(false);

    // ucitavamo trazeni zbir
    int trazeniZbir;
    cin >> trazeniZbir;

    // ucitavamo elemente niza
    int n;

```



```

cin >> n;
int a[50000];
for (int i = 0; i < n; i++)
    cin >> a[i];

// broj segmenata traženog zbira
int brojSegmenata = 0;

// granice segmenta
int i = 0, j = 0;
// zbir segmenta
int zbir = a[0];
while (true) {
    if (zbir < traženiZbir) {
        // prelazimo na segment [i, j+1]
        j++;
        // ako takav segment ne postoji, završili smo pretragu
        if (j >= n)
            break;
        // izračunavamo zbir segmenta[i, j+1] na osnovu zbira [i, j]
        zbir += a[j];
    } else {
        // ako je zbir jednak traženom uvećavamo brojac pronađenih
        if (zbir == traženiZbir)
            brojSegmenata++;
        // prelazimo na interval [i+1, j]
        // izračunavamo zbir intervala [i+1, j] na osnovu zbira [i, j]
        zbir -= a[i];
        i++;
    }
}

cout << brojSegmenata << endl;

return 0;
}

```

## Државно такмичење 2018. године 7. и 8. разред

### 1. [куровина]

Решење: Погледати задатак за 5. и 6. разред

2. [бинарни низови] Написати програм којим се приказују сви бинарни низови (низови који садрже само нуле и јединице) дужине  $n$  који не садрже две узастопне јединице. Низове приказати у лексикографском поретку (тако да су бинарни бројеви формиран од тих низова поређани растуће). Прва линија стандардног улаза садржи природан број  $n$  ( $n \leq 25$ ). На стандардном излазу приказати тражене бинарне низове у лексикографском поретку, сваки низ у посебној линији.

*Пример*

<i>Улаз</i>	<i>Излаз</i>
3	000
	001
	010
	100
	101

**Решење:**

```
#include <iostream>
#include <vector>

using namespace std;

void prikazi(const vector<bool>& s){
    for (bool b : s)
        cout << (b ? "1" : "0");
    cout << endl;
}

void binarniNizovi_(vector<bool>& s, int i) {
    if (i == s.size()) {
        prikazi(s);
        return;
    }

    s[i] = false;
    binarniNizovi_(s, i + 1);
    if (i == 0 || s[i-1] == false) {
        s[i] = true;
        binarniNizovi_(s, i + 1);
    }
}

void binarniNizovi(int n) {
    vector<bool> s(n);
    binarniNizovi_(s, 0);
}

int main() {
```

```

int n;
cin >> n;
binarniNizovi(n);
return 0;
}

```

3. **[akcije]** Вредност акција једне компаније варирала је из дана у дан током једног дужег временског периода. Напиши програм који одређује колико има низова узастопних дана у којима је укупан збир промена вредности акција једнак датом броју  $z$ . Са стандардног улаза се у првој линији уноси тај број  $z$  (цео број  $-10000$  и  $10000$ ), затим, у наредној линији број дана  $n$  у којима се пратила промена вредности акција ( $3 \leq n \leq 50000$ ) и затим у наредној линији промене за сваки од дана у односу на претходни дан (цели бројеви између  $-100$  и  $100$ ) раздвојени размацама. На стандардни излаз испиши број периода у којима је укупна промена једнака  $z$  (тј. броја непразних сегмената низа промена чији је збир једнак  $z$ ).

*Пример*

*Улаз*

```

11
10
1 2 3 5 1 -1 1 5 3 2

```

*Излаз*

```

7

```

*Објашњење:* следећи сегменти имају збир 11

```

1 2 3 5
1 2 3 5 1 -1
2 3 5 1
2 3 5 1 -1 1
5 1 -1 1 5
1 -1 1 5 3 2
1 5 3 2

```

**Решење:**

```

#include <iostream>
#include <map>
#include <vector>

using namespace std;

int main() {
    ios_base::sync_with_stdio(false); cin.tie(0);

    // učitavamo traženi zbir
    int traženiZbir;
    cin >> traženiZbir;

    // zbir prefiksa
    int zbirPrefiksa = 0;

```

```

// broj segmenata sa trazanim zbirom
int broj = 0;

// broj pojavljivanja svakog vidjenog zbira prefiksa
map<int, int> zbiroviPrefiksa;
// zbir pocetnog praznog prefiksa je 0 i on se za sada pojavio
// jednom
zbiroviPrefiksa[0] = 1;

// ucitavamo elemente niza niz
int n;
cin >> n;
for (int i = 0; i < n; i++) {
    int x;
    cin >> x;
    // prosirujemo prefiks tekucim elementom
    zbirPrefiksa += x;

    // trazimo broj pojavljivanja vrednosti zbirPrefiksa - trazeniZbir
    // i azuriramo broj pronadjenih segmenata
    auto it = zbiroviPrefiksa.find(zbirPrefiksa - trazeniZbir);
    if (it != zbiroviPrefiksa.end())
        broj += it->second;

    // povecavamo broj pojavljivanja trenutnog zbira
    zbiroviPrefiksa[zbirPrefiksa]++;
}

cout << broj << endl;

return 0;
}

```

4. [**domine**] Домине се слажу једна уз другу, тако што се поља на доминама постављеним једну уз другу морају поклапати. Домине обично имају само два поља, међутим, наше су домине специјалне и имају више различитих поља (означених словима). Ако све домине које слажемо имају исту шару, напиши програм који одређује како је домине могуће сложити тако да заузму што мање простора по дужини (свака наредна домина мора бити смакнута бар за једно поље). На пример, ако на доминама пише ababscabab, најмање простора заузимају ако се сложе на следећи начин:

```

ababscabab
  ababscabab
    ababscabab

```

*Улаз:* Први ред стандардног улаза садржи ниску малих слова енглеске абетеде које представљају шаре на доминама. Дужина ниске је између 2 и 100000 карактера. У наредном реду се налази цео број  $n$  ( $1 \leq n \leq 50000$ ) који представља број домина.

*Излаз:* На стандардни излаз исписати цео број који представља укупну дужину сложених домина.

Примери

Улаз:	Излаз:	Улаз:	Излаз:	Улаз:	Излаз:
ababababab	19	abc	15	aa	11
3		5		10	

### Решење:

```
#include <iostream>
#include <string>
#include <vector>

using namespace std;

int main() {
    string str;
    cin >> str;
    int n;
    cin >> n;

    vector<int> kmp(str.size() + 1);
    kmp[0] = -1;
    for (int i = 0; i < str.size(); i++) {
        int k = i;
        while (k > 0 && str[i] != str[kmp[k]])
            k = kmp[k];
        kmp[i + 1] = kmp[k] + 1;
    }

    cout << kmp[str.size()] + n * (str.size() - kmp[str.size()]) << endl;

    return 0;
}
```

