**20. siječnja 2023.**

2023 Natjecanje iz informatike

Školska razina 2023./ Osnovna škola (5. i 6. i 7. i 8. razred)

Primjena algoritama OŠ

# OPISI ALGORITAMA

Agencija za odgoj i obrazovanje
Education and Teacher Training Agency

HRVATSKI SAVEZ INFORMATIČARA

Ministarstvo znanosti, obrazovanja i sporta

## 5.1. Zadatak: Vikend — Autor: Nikola Dmitrović

Između subote koja je početak vikenda i šesti dan u tjednu te D-tog dana u tjednu je ukupno 6 – D – 1 dana.

*Programski kod (pisan u Python 3)*

```
D = int(input())
print(5 - D)
```

**Potrebno znanje:** naredba učitavanja i ispisa

**Kategorija:** ad hoc

## 5.2. Zadatak: Naši — Autor: Nikola Dmitrović

Utakmica je u produžetku završila rezultatom X:Y. To znači da je suparnička momčad tijekom cijele utakmice postigla Y golova. Kako je moguće da u produžetku nije dala niti jedan gol ili da ih je dala sve, tada je ukupan broj rezultata kojim je mogao završiti regularni dio Y + 1.

*Programski kod (pisan u Python 3)*

```
X = int(input())
Y = int(input())
print(Y + 1)
```

**Potrebno znanje:** naredba učitavanja i ispisa

**Kategorija:** ad hoc

## 5.3. Zadatak: Ruke — Autor: Nikola Dmitrović

Svaki od šest brojeva s ulaza označava redom ishode ogleda: A:B, A:C, A:D, B:C, B:D, C:D. Za svaki broj treba provjeriti je li jedan te ovisno o točnosti upita povećati broj pobjeda za jedan osobi iz ogleda, ovisno o kojem se ogledu radi. Kod provjere prvog broja treba ispisati odmah i oznaku pobjednika ogleda A:B.

*Programski kod (pisan u Python 3)*

```
O1 = int(input()); O2 = int(input()); O3 = int(input())
O4 = int(input()); O5 = int(input()); O6 = int(input())
A = B = C = D = 0
# A:B
if O1 == 1:
    print("A")
    A += 1
else:
    print("B")
    B += 1
```

```
# A:C
if O2 == 1:
    A += 1
else:
    C += 1
#A:D
if O3 == 1:
    A += 1
else:
    D += 1
# B:C
if O4 == 1:
    B += 1
else:
    C += 1
# B:D
if O5 == 1:
    B += 1
else:
    D += 1
# C:D
if O6 == 1:
    C += 1
else:
    D += 1
```

**Potrebno znanje:** naredba odlučivanja

**Kategorija:** ad hoc

## 6.1. Zadatak: Oni Autor:

Vidi zadatak 5.2.

## 6.2. Zadatak: Katovi Autor: Nikola Dmitrović

U početku nalazimo se na prvom katu (trenutno = 1) te smo napravili nula promjena (promjena = 0). Redom učitavamo katove po kojima se krećemo te za svaki provjerimo koliko je udaljen od kata na kojem se trenutno nalazimo te broj promjena povećamo za tu razliku. Trenutni kat promijenimo u kat na kojem se nalazimo.

*Programski kod (pisan u Python 3)*

```
N = int(input())
K = int(input())
trenutno = 1
promjena = 0
for i in range(N):
    kat = int(input())
    promjena += abs(trenutno - kat)
    trenutno = kat
print(promjena)
```

**Potrebno znanje:** naredba ponavljanja, naredba odlučivanja

**Kategorija:** ad hoc

## 6.3. Zadatak: Obaranje Autor: Nikola Dmitrović

U primjerima vrijednima 45 bodova vrijedilo je da je **N**=4. Rješenje tog dijela zadatka pogledajte pod 5.3.

Zadatak definira da su prvo snage odmjerile osoba „1“ redom sa svim ostalima više oznake, pa „2“ redom sa svim ostalima više oznake i na kraju K-ti po redu dvoboj između osoba „N-1“ i „N“. Za riješiti cijeli zadatak treba uočiti da je podatke moguće učitavati dvostrukom for petljom i redom analizirati. Za praćenje rezultata po osobama treba koristiti listu/niz na razini spremanja podataka.

*Programski kod (pisan u Python 3)*

```
N = int(input())
K = int(input())
bodovi = [0] * N
for i in range(N):
    for j in range(i+1,N):
        Oi = int(input())
        if Oi == 1:
            bodovi[i] += 1
        else:
            bodovi[j] += 1
print(bodovi)
```

**Potrebno znanje:** naredba ponavljanja

**Kategorija:** ad hoc

## 7.1. Zadatak: Ruke Autor:

Pogledaj zadatak 5.3.

## 7.2. Zadatak: Kaladont Autor: Patrick Pavić

Učitavamo riječ po riječ i pratimo trenutak kada nisu ispunjeni uvjeti iz teksta zadatka.

*Programski kod (pisan u Python 3)*

```
n = int(input())
lst = ""
for i in range(n):
    s = input()
    if lst != "" and s[:2] != lst[-2:]:
        print(["Roko", "Juraj"][i % 2])
        print(s)
        lst = ""
        break
    lst = s
if lst != "":
    print("Dominik")
```

**Potrebno znanje:** stringovi

**Kategorija:** ad hoc

## 7.3. Zadatak: Zabavljač Autor: Gabrijel Jambrošić

Budući da trebamo odrediti dobitak u najgorem slučaju među svim mogućim Patrikovim zamjenama, možemo gledati kao da Stjepan prvo odabire kutije, a zatim ih Patrik mijenja tako da Stjepan dobije najmanje novca.

Stjepanu je optimalno izabrati M kutija s najviše novca, a Patrik će tada mijenjati izabranu kutiju koja trenutno sadrži najviše novca s neizabranom kutijom koja trenutno sadrži najmanje novca jer će na taj način najviše smanjiti Stjepanov dobitak. Ako u nekom trenutku svaka izabrana kutija sadrži manje novca od svake neizabrane, tada Patrik (ako mu je ostalo zamjena) može mijenjati dvije izabrane ili dvije neizabrane kutije ne mijenjajući pritom rezultat.

*Programski kod (pisan u C++)*

```
#include <bits/stdc++.h>
using namespace std;
```

```
int main() {
    int n, m, k, suma = 0;
    cin >> n >> m >> k;
    k = min(k, min(m, n - m));
    vector <int> svi(n);
    for(int i = 0; i < n; i++) cin >> svi[i];
    sort(svi.begin(), svi.end());
    for(int i = n - m; i < n; i++) suma += svi[i];
    for(int i = 0; i < k; i++) suma += svi[i] - svi[n - 1 - i];
    cout << suma << endl;
    return 0;
}
```

**Potrebno znanje:** lista

**Kategorija:** ad hoc

## 8.1. Zadatak: Obaranje Autor:

Pogledaj zadatak 6.3.

## 8.2. Zadatak: Zabavljač Autor:

Pogledaj zadatak 7.3.

## 8.3. Zadatak: Komadići Autor: Stjepan Požgaj

Provjerom slučajeva tražimo odgovor na pitanje iz teksta zadatka.

*Programski kod (pisan u C++)*

```
N #include <iostream>
#include <cstdio>
#include <cassert>
#include <set>
#include <vector>
#include <string>
using namespace std;
#define TRACE(x) cerr << #x << " = " << x << endl
#define FOR(i, a, b) for(int i = (a); i < (b); i++)
#define REP(i, n) FOR(i, 0, n)
typedef long long int llint;
typedef pair<int, int> par;
#define X first
```

```
#define Y second


vector<string> ucitaj_pravokutnik() {
  int n, m;
  cin >> n >> m;
  vector<string> ret;
  REP(i, n) {
    string s;
    cin >> s;
    ret.push_back(s);
  }
  return ret;
}
pair<bool, par> provjeri(vector<vector<int>> mat) {
  auto f = make_pair(false, par(-1, -1));
  int mini_i = 1000, maksi_i = -1;
  int mini_j = 1000, maksi_j = -1;
  REP(i, (int) mat.size())
    REP(j, (int) mat[i].size()) {
      if(mat[i][j] > 1) return f;
      if(mat[i][j] == 0) continue;
      assert(mat[i][j] == 1);
      mini_i = min(mini_i, i);
      mini_j = min(mini_j, j);
      maksi_i = max(maksi_i, i);
      maksi_j = max(maksi_j, j);
    }
  FOR(i, mini_i, maksi_i + 1)
    FOR(j, mini_j, maksi_j + 1)
      if(mat[i][j] != 1)
        return f;
  return make_pair(true, par(maksi_i + 1 - mini_i, maksi_j + 1 - mini_j));
}
pair<bool, par> probaj(vector<string> prvi,
                       vector<string> drugi,
                       par pomak) {
```

```
  vector<vector<int>> mat(50, vector<int>(50));
  REP(i, (int) prvi.size())
    REP(j, (int) prvi[i].size())
      if(prvi[i][j] == '#')
        mat[15 + i][15 + j]++;
  REP(i, (int) drugi.size())
    REP(j, (int) drugi[i].size())
      if(drugi[i][j] == '#')
        mat[pomak.X + i][pomak.Y + j]++;
  return provjeri(mat);
}
pair<bool, par> spoji(vector<string> prvi,
                      vector<string> drugi) {
  // ovaj set samo koristim da provjerim je li rjesenje jedinstveno
  set<pair<bool, par>> s;
  REP(i, 30)
    REP(j, 30) {
      auto t = probaj(prvi, drugi, par(i, j));
      if(t.X)
        s.insert(t);
    }
  assert(s.size() == 0 || s.size() == 1);
  if(s.size() == 1)
    return *s.begin();
  return make_pair(false, par(-1, -1));
}

int main() {
  auto prvi = ucitaj_pravokutnik();
  auto drugi = ucitaj_pravokutnik();
  auto t = spoji(prvi, drugi);
  if(t.X) {
    cout << "DA" << endl;
    cout << t.Y.X << " " << t.Y.Y << endl;
  }
  else
```

```
    cout << "NE" << endl;
  return 0;
}
```

**Potrebno znanje:** dvodimenzionalna lista

**Kategorija:** ad hoc