INFORMATIČKI KLUB

FUTURA

```
#include<stdio.h>
int main()
{
    printf("Hello World!");
    return 0;
}
```

C

RADIONICE PROGRAMIRANJA

ZA SREDNJE ŠKOLE

RADIONICE PROGRAMIRANJA ZA SREDNJE ŠKOLE - 6. RADIONICA

Nikola Rabrenović i Krunoslav Žubrinić
Informatički klub FUTURA
Dubrovnik, 24. siječnja 2015.

Creative Commons

- **slobodno smijete:**
  - **dijeliti** — umnožavati, distribuirati i javnosti priopćavati djelo
  - **remiksirati** — prerađivati djelo
- **pod slijedećim uvjetima:**
  - **imenovanje**. Morate priznati i označiti autorstvo djela na način kako je specificirao autor ili davatelj licence (ali ne način koji bi sugerirao da Vi ili Vaše korištenje njegova djela imate njegovu izravnu podršku).
  - **nekomercijalno**. Ovo djelo ne smijete koristiti u komercijalne svrhe.
  - **dijeli pod istim uvjetima**. Ako ovo djelo izmijenite, preoblikujete ili stvarate koristeći ga, preradu možete distribuirati samo pod licencom koja je ista ili slična ovoj.

U slučaju daljnjeg korištenja ili distribuiranja morate drugima jasno dati do znanja licencne uvjete ovog djela. Najbolji način da to učinite je linkom na ovu internetsku stranicu.

Od svakog od gornjih uvjeta moguće je odstupiti, ako dobijete dopuštenje nositelja autorskog prava.

Ništa u ovoj licenci ne narušava ili ograničava autorova moralna prava.

Tekst licence preuzet je s http://creativecommons.org/.

Sadržaj

- Rješenje zadataka sa školskog natjecanja 2015.
  - Joker (1-2. razred)
  - Flash (1.-2. i 3.-4. razred)
  - Piton (1.-2. razred)

Zadatak: Joker

- Popularna igra na sreću sastoji se od izvlačenja 6 kuglica iz bubnja na kojima su napisani različiti prirodni brojevi između 1 i 45.
- Dobitni joker broj se formira iz izvučenih brojeva tako da se istim redoslijedom kojim su kuglice izvučene zapišu njihove zadnje znamenke.
- Na primjer, ako su redom izvučeni brojevi **12, 35, 1, 2, 23 i 39**, onda je joker broj **251239**.
- Dobitak listića računa se tako što se serijski broj napisan na listiću usporedi s izračunatim joker brojem tako da se prebroji koliko se zadnjih znamenki ova dva broja poklapa.

Serijski brojevi	Naziv dobitka
251239	I. vrsta
X51239	II. vrsta
XX1239	III. vrsta
XXX239	IV. vrsta
XXXX39	V. vrsta
XXXXXX	Nedobitan

Zadatak: Joker

ULAZNI PODACI:

- U prvom redu nalazi se točno šest različitih prirodnih brojeva manjih ili jednakih 45 koji predstavljaju brojeve na kuglicama izvučenim iz bubnja.
- U svakom od sljedeća tri reda ulaza nalazi se po jedan niz od šest znamenaka koji predstavlja serijski broj listića. Serijski broj uvijek ima šest znamenaka te može imati vodeće nule.

Serijski brojevi	Naziv dobitka
251239	I. vrsta
X51239	II. vrsta
XX1239	III. vrsta
XXX239	IV. vrsta
XXXX39	V. vrsta
XXXXXX	Nedobitan

IZLAZNI PODACI:

- U svaki od tri reda potrebno je ispisati naziv dobitka za odgovarajući serijski broj.

ulaz

```
12 35 1 2 23 39
151239
251229
251339
```

izlaz

```
II. vrsta
Nedobitan
V. vrsta
```

ulaz

```
5 45 35 25 15 1
555551
235551
555552
```

izlaz

```
I. vrsta
III. vrsta
Nedobitan
```

ulaz

```
2 17 33 12 39 44
000022
001194
232294
```

izlaz

```
Nedobitan
V. vrsta
IV. vrsta
```

Vrijeme

ionica

Rješenje zadatka: Joker

- Direktnom simulacijom postupka:
  - Odredite joker broj (zadnja znamenka dobitnih brojeva) i spremiti ga u niz char-ova.
    - **broj%10 + '0'** (+'0' treba za pretvorbu broja int->char)
  - Krenuvši od kraja uspoređivati i-tu znamenku kupljenog listića s i-tom znamenkom jokera
    - Za svaku uspješnu usporedbu povećaj brojač za 1.
    - Ovisno o vrijednosti brojača ispiši poruku.

Rješenje zadatka: Joker

```
Unesite 6 dobitnih brojeva: 2 17 33 12 39 44
Unesite 3 dobitna listiša: 000022 001194 232294
Nedobitan
V. vrsta
IV. vrsta
```

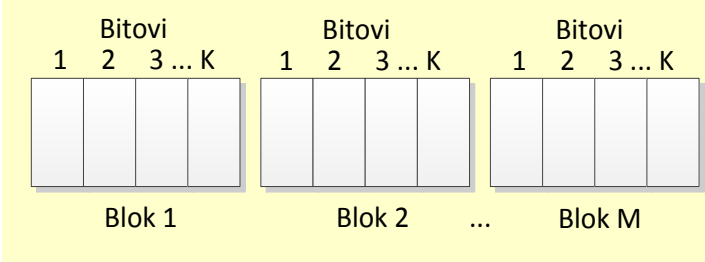
```c
#include <stdio.h>
#include <stdlib.h>
int main() {
   int niz[6], i, j, k, n = -1;
   char list[3][6 + 1], dob[6 + 1];
   printf("Unesite 6 dobitnih brojeva: ");
   scanf("%d %d %d %d %d %d", &niz[0], &niz[1], &niz[2], &niz[3], &niz[4], &niz[5]);
   printf("Unesite 3 dobitna listića: ");
   scanf("%s %s %s", list[0], list[1], list[2]);
   for (i = 0; i<6; i++) {
      dob[i] = '0'+(niz[i] % 10);
   }
   for (i = 0; i < 3; i++) {
      for (j = 5; j >= 0; j--){
         if (list[i][j] != dob[j]){
            n = 99;
            ispis(j);
            break;
         }
      }
      if (n == -1)
         ispis(-1);
   }
   system("pause");
   return 0;
}
```

```c
// funkcija za ispis
void ispis(int i){
   if (i > 3)  printf("Nedobitan\n");
   else if (i == 3) printf("V. vrsta\n");
   else if (i == 2) printf("IV. vrsta\n");
   else if (i == 1) printf("III. vrsta\n");
   else if (i == 0) printf("II. vrsta\n");
   else if (i == -1) printf("I. vrsta\n");
}
```

joker.c

Zadatak: Flash

- Flash memorija je vrsta elektroničke memorije koja ne gubi informacije nakon prekida napona. Mirko je nedavno patentirao i proizveo novu vrstu jeftine memorije koju je nazvao MOR. MOR je jeftina u proizvodnji, ali je složenija za rukovanje zbog određenih ograničenja.
- MOR memorija se sastoji od niza od M blokova, gdje se svaki blok sastoji od točno K bitova.
- Kod MOR memorije nije uvijek moguće postaviti pojedini bit na željenu vrijednost već su dopuštene samo dvije operacije:
  - Pojedini bit možemo postaviti na 0 te ova operacija traje 1 ms.
  - Sve bitove u pojedinom bloku možemo postaviti na 1 te ova operacija traje 100 ms.
- Napišite program koji će za zadano početno i traženo stanje memorije pronaći najmanje vrijeme potrebno da se memorija iz početnog stanja dovede u traženo.



Zadatak: Flash

ULAZNI PODACI:

- U prvom redu nalaze se dva prirodna broja, M i K (M, K ≤ 20, M*K ≤ 80) odvojena razmakom - broj blokova i broj znakova bloka.
- U drugom redu nalazi se niz od točno M*(K+1)-1 znakova - početno stanje.
- U trećem redu nalazi se niz od točno M*(K+1)-1 znakova - traženo stanje.
  - Početno i traženo stanje memorije su nizovi znakova koji se sastoje od M blokova međusobno odvojenih znakom '|', a svaki blok se sastoji od točno K znakova '0' ili '1' koje predstavljaju vrijednost određenog bita u bloku.

IZLAZNI PODACI:

- Najmanje moguće vrijeme u ms potrebno da se memorija postavi u traženo stanje.

ulaz

```
2 4
0110|1000
0000|0000
```

izlaz

```
3
```

ulaz

```
2 4
0110|1000
0000|0001
```

izlaz

```
105
```

ulaz

```
3 3
110|011|111
101|111|011
```

izlaz

```
202
```

Rješenje zadatka: Flash

- Obrađuje se blok po blok.
- Uspoređuju se znamenke na istim pozicijama u nizovima u drugom i trećem redu.
  - Ako su znamenke različite, vrši se promjena:
    - Ako je gore 0, a dolje 1:
      - Postavi sve bitove bloka u 1.
      - vrijeme = vrijeme + 100
- Prođi kroz sve bitove (neovisno o bloku)
  - Ako je gore 1, a dolje 0:
    - vrijeme = vrijeme + 1

Rješenje zadatka: Flash

```
Unesi 2 broja: 3 3
Unesi staro stanje i novo stanje: 110|011|111
101|111|011
202
```

```
int main() {
   int blok, cel, i, j, iznos = 0;
   char a[20 + 1], b[20 + 1];
   printf("Unesi 2 broja: ");
   scanf(" %d %d", &blok, &cel);
   printf("Unesi staro stanje i novo stanje: ");
   scanf(" %s %s", a, b);
   // prvo zamijeni blokove s 1 gdje je potrebno
   for (i = 0; i < blok*cel; i = i + cel) {
      if (zamjena_1(a, b, i, i + cel) == 1) {
         for (j = i; j < i + cel; j++)
             a[j] = '1';
         iznos = iznos + 100;
      }
   }
   // sada zabilježi sve zamjene 1->0
   for (i = 0; i < blok*cel; i++) {
      if ((a[i] == '1') && (b[i] == '0'))
         iznos = iznos + 1;
   }
   printf("%d", iznos);
   system("pause");
   return 0;
}
```

```
int zamjena_1(char niz[20 + 1], char niz2[20 + 1], int a, int b){
   int i;
   for (i = a; i <= b; i++){
      if ((niz[i] == '0') && (niz2[i] == '1'))
         return 1;
   }
   return 0;
}
```

Zadatak: Piton

- Piton je strukturirani programski jezik sintaksom sličan popularnom Pythonu koji ima samo dvije naredbe:
  - **Px** gdje je **x** malo slovo engleske abecede: ova naredba ispisuje slovo **x**.
  - **Fk** gdje je **k** prirodni broj manji ili jednak od 20, zapisan bez vodećih nula: ova naredba ponavlja točno **k** puta blok naredbi koji slijedi.
- Blokovi naredbi se specificiraju uvlačenjem naredbi, ali se koristi znak '.' (točka) umjesto uobičajenih razmaka.
  - Za nenegativni cijeli broj N definiramo N-uvučeni blok kao niz linija koji se sastoji od jedne ili više:
    - Naredbi P uvučene s N točaka.
    - Naredbi F uvučene s N točaka nakon koje slijedi točno jedan M-uvučeni blok gdje je M cijeli broj veći od N.
- Jedan blok može sadržavati proizvoljan broj P i F naredbi proizvoljnim redoslijedom.
- Nakon svake F naredbe slijedi blok koji mora biti više uvučen nego sama naredba.
- Prva linija ne sadrži točku).

Zadatak: Piton

Napišite program koji će, za zadani program u Pitonu, pronaći **koliko kojih malih slova** zadani program ispisuje.

ULAZNI PODACI:

- U prvom redu nalazi se broj (L ≤ 100) linija zadanog programa.
- Sljedećih L linija sadrži niz od najviše 80 znakova – programski kod programa.
  - Dozvoljeni znakovi su točke, mala slova engleske abecede, velika slova ‘F’ i ‘P’ te znamenke ‘0’ do ‘9’.

IZLAZNI PODACI:

- Potrebno je ispisati koliko se različitih malih slova pojavljuje u izlazu zadanog programa. Slova trebaju biti ispisana abecednim redosljedom.
  - Za svako slovo x koje se pojavljuje u izlazu potrebno je ispisati jedan red oblika '**x k**', gdje je k ukupan broj ispisivanja slova x u programu.

Zadatak: Piton

ulaz

```
5
Px
Py
F3
..Px
..Pz
```

izlaz

```
x 4
y 1
z 3
```

ulaz

```
9
F3
..Pa
..Pb
..F4
...Pc
..F2
.....Pd
.....Pe
..Pa
```

izlaz

```
a 6
b 3
c 12
d 6
e 6
```

ulaz

```
8
Pa
F3
.F12
..Px
..F20
...F2
....F3
.....Pa
```

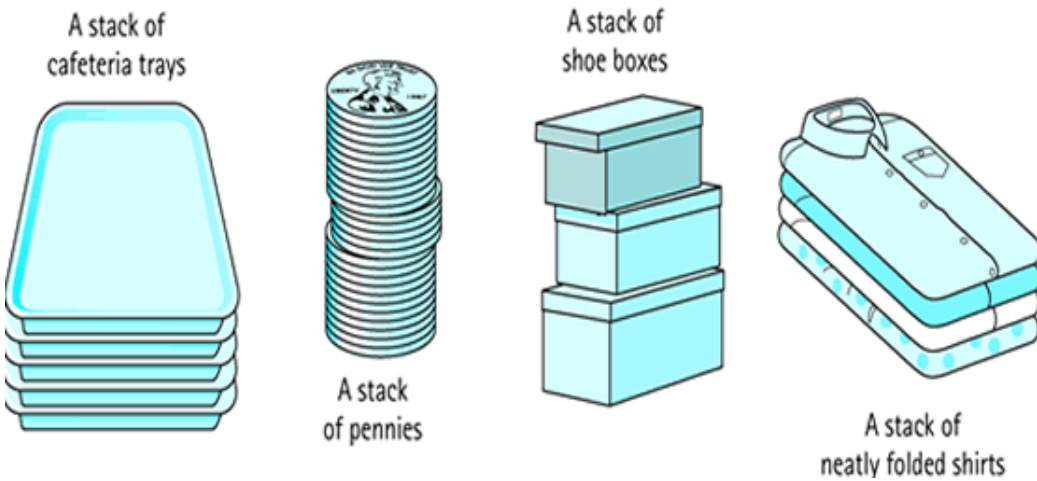
izlaz

```
a 4321
x 36
```

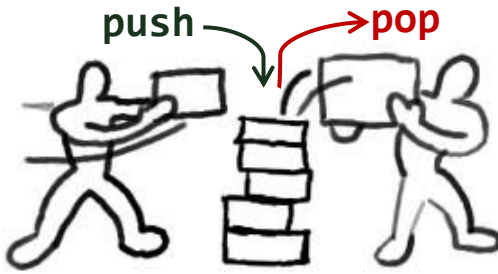
Vrijeme

Rješenje zadatka: Piton

- Problem provjere sintaksne ispravnosti računalnog programa.
- Za provjeru ispravnosti strukture računalnog programa uobičajeno se koristi struktura podataka **stog** (*stack*).



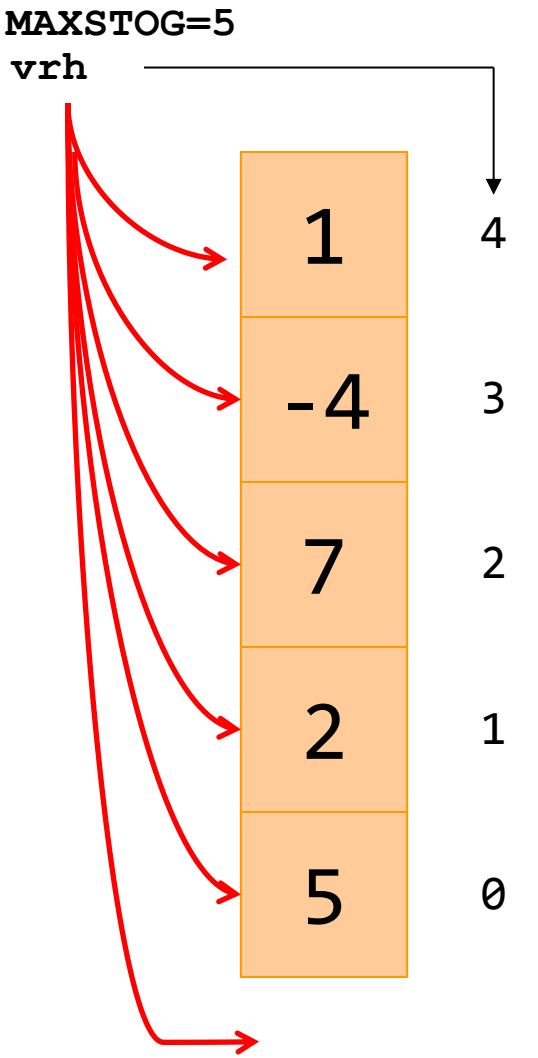
Stog



- Struktura podataka kod koje se posljednji pohranjeni podatak prvi uzima u obradu
- Potrebne operacije:
  - dodavanje (***push***) elemenata na vrh stoga (*top*)
  - brisanje (***pop***) elemenata s vrha stoga
  - inicijalizacija praznog stoga
- Može se realizirati poljem
  - u jednodimenzionalno polje se dodaju ili brišu pojedine stavke prema načelu *Last In First Out* (*LIFO*).
  - ažurira se vrijednost varijable koja predstavlja vrh stoga.
- Inicijalizacija praznog stoga:
  - postavljanje vrha na početnu vrijednost

```
vrh = -1;
```

Dodavanje elementa na stog

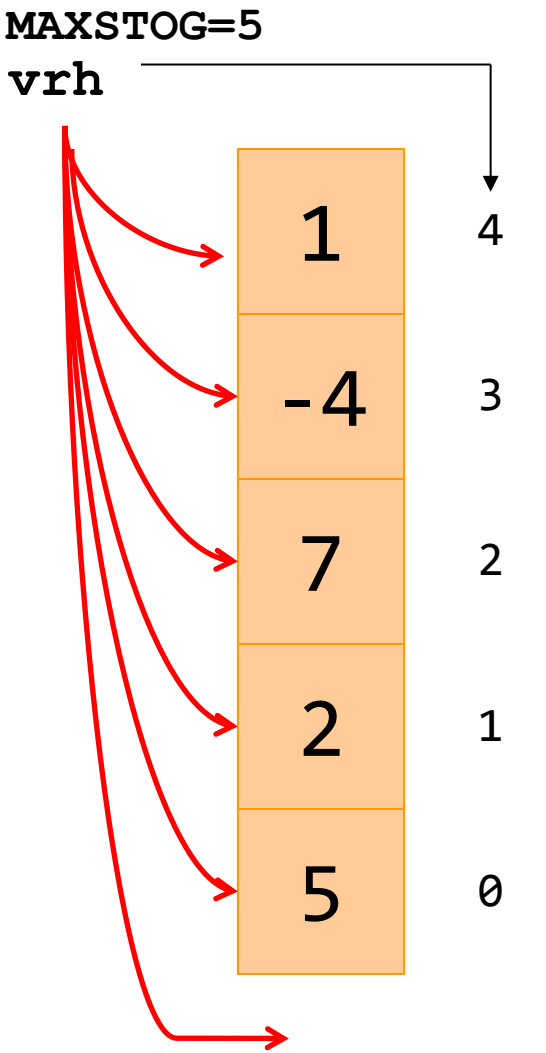



```
if (vrh < MAXSTOG-1) {
  vrh++;
  polje[vrh] = element;
}
```

```
vrh = -1;

dodaj(5);
dodaj(2);
dodaj(7);
dodaj(-4);
dodaj(1);
dodaj(9);
```

Skidanje elemenata sa stoga




```
if (vrh >= 0) {
   podatak = polje[vrh];
   vrh--;
}
```

```
skini();
skini();
skini();
skini();
skini();
skini();
```

Rješenje zadatka: Piton

- U jednom retku nakon uvlaka (točkica) može se naći slovo P ili F
- Moramo odrediti broj točkica da bismo:
  - Izravno pristupili slovu naredbe.
  - Znali na kojoj razini se naredba nalazi.
- Dakle, prvo prebrojimo točkice.
- Ako je naredba P, slovu iza naredbe pribrojimo broj ponavljanja.
- Ako je naredba F, izdvojimo broj ponavljanja iza naredbe.
  - Na stog stavimo dubinu i broj ponavljanja.
- Poseban slučaj je kada treba skinuti element sa stoga.
  - Ako je tekuća dubina manja od posljednje (na stogu), skida se, a broj ponavljanja se smanjuje.

Rješenje zadatka: Piton

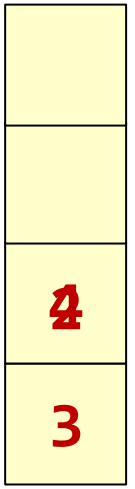
F3 -> blok se izvodi 3 puta (uvlaka 0+)

..Pa -> naredba se izvodi 3 puta

..Pb -> naredba se izvodi 3 puta

..F4 -> blok se izvodi 4 puta (uvlaka 2+)

..F2 -> blok se izvodi 2 puta (uvlaka 2+)



Rješenje zadatka: Piton

```
Unesi broj linija koda: 9
F3
..Pa
..Pb
..F4
...Pc
..F2
.....Pd
.....Pe
..Pa
a 6
b 3
c 12
d 6
e 6
```

```
int main() {
  int i, br, dub, vrh = -1, pon = 1;
  int slova[26] = { 0 };
  char n[80 + 1], brpon[3] = { '\0' };
  int stog[100][2];
  printf("Unesi broj linija koda: ");
  scanf(" %d", &br);
  for (i = 0; i < br; i++){
    scanf(" %s", n);
    dub = vrati_dubinu(n);
    while (vrh >= 0 && stog[vrh][0] >= dub) {
      pon = pon / stog[vrh][1];
      vrh--;
    }
    if (n[dub] == 'P'){
      slova[n[dub + 1] - 'a'] += pon;
    }
    else {// n[dub] == 'F'
      strncpy(brpon, n + dub + 1, strlen(n)-1);
      pon = pon * atoi(brpon);
      vrh++;
      stog[vrh][0] = dub;
      stog[vrh][1] = atoi(brpon);
    }
  }
  for (i = 0; i<27; i++) {
    if (slova[i]>0)
      printf("%c %d\n", i+'a', slova[i]);
  }
  system("pause");
  return 0;
}
```

```
int vrati_dubinu(char naredba[]){
   int i, br = 0;
   for (i = 0; naredba[i] != '\0'; i++){
      if (naredba[i] == '.')
         br++;
      else
         break;
   }
   return br;
}
```


piton.c