FUTURA

IV. RADIONICA - INFOKUP NATJECANJE

ALGORITMI ZA SREDNJE ŠKOLE

Tomo Sjekavica, Informatički klub FUTURA

Dubrovnik, 09. siječanj 2014.

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/.

Kockice 1/2

- Mirko je preko školskih praznika sa svojim prijateljima osmislio novu društvenu igru koja ih jako zabavlja.
- Pravila igre su jednostavna. Igra se igra tako da svaki igrač baca **pet kockica**, a na svakoj kockici su napisane vrijednosti od 1 do 6. Pobjednik igre je onaj igrač koji u kombinaciji od bačenih pet kockica ima **najviše istih kockica**, odnosno najviše kockica s **istom dobivenom vrijednošću**.

Kockice 2/2

Školsko natjecanje
Algoritmi – 2012
Srednja škola

- Mirku i njegovim prijateljima dosadilo je za svako bacanje brojati kockice, pa im treba program koji će to raditi za njih.
- Ulaz
  - U prvom i jedinom retku nalazi se **pet prirodnih brojeva k_i ($1 \leq k_i \leq 6$)**, koji predstavljaju vrijednost i-te bačene kockice.
- Izlaz
  - U prvom i jedinom retku potrebno je ispisati jedan prirodan broj koji označava najveći broj kockica s istom vrijednošću u bačenoj kombinaciji.

Vrijeme

Primjeri testnih podataka

ULAZ	ULAZ	ULAZ
1 2 2 6 5	5 6 4 1 3	6 5 5 5 5
IZLAZ	**IZLAZ**	**IZLAZ**
2	1	4

Kockice - rješenje

```
#include<stdio.h>
int main(){
    int i, kockica;
    int k[6] = {0}, max = 0;
    for(i=0; i<5; i++){
        scanf("%d", &kockica);
        k[kockica-1]++;
        if(k[kockica-1] > max)
            max = k[kockica-1];
    }
    printf("%d\n", max);
    system("PAUSE");
    return 0;
}
```

```
1 2 2 6 5
2
Press any key to continue . . .
```

```
5 6 4 1 3
1
Press any key to continue . . .
```

```
6 5 5 5 5
4
Press any key to continue . . .
```

kockice.c

WiFi 1/5

- Jedan poznati svjetski pružatelj Internet usluga (eng. *Internet service provider*) radi testiranje efikasnosti svoje mreže pod određenim opterećenjem. Da bi to napravili, odlučili su postaviti jedan usmjernik (eng. *router*) i na njega bežično povezati određen broj računala.
- Za svako računalo odredili su **koliko podataka treba poslati** usmjerniku. Podaci se šalju **u krugovima**, a količina podataka koje računala šalju svaki krug je **fiksna**. Redoslijed slanja podataka za računala definiran je određenim brojem **krugova** na sljedeći način.

WiFi 2/5

- Računala su sortirana u listu prioriteta kako su zadana na ulazu. Prvi na ulazu je i prvi u listi prioriteta i ima najveći prioritet, dok je zadnji na ulazu ujedno i zadnji na listi prioriteta i ima najmanji prioritet. Kada neko računalo pošalje sve svoje podatke usmjerniku, **briše se** iz te liste, a slanje po krugovima nastavljaju ostala računala.
- Nakon brisanja nekog računala iz liste, sva ostala računala s manjim prioritetom od izbrisanog računala se pomiču za jedno mjesto unaprijed u listi prioriteta. Ako je bilo 10 računala na listi prioriteta i 4. računalo je završilo sa slanjem podataka, potrebno je izbrisati 4. računalo, a računala 5-10 se pomiču za jedno mjesto u listi prioriteta, na način da računalo sa prioritetom 5 dobiva prioritet 4 i tako dalje.

WiFi 3/5

- Za svaki krug definirano je koja računala šalju podatke i to tako da je zadano **koliko prvih računala iz liste taj krug šalje podatke**. Ako kažemo da u nekom krugu podatke šalje prvih 5 računala, tada podatke šalje prvih 5 računala s liste prioriteta.
- Tim stručnjaka iz spomenute tvrtke zanima koliko krugova će trebati da sva računala pošalju sve podatke.
- **NAPOMENA:** Ukoliko nakon zadnjeg definiranog kruga slanja još uvijek ima računala s neposlanim podacima, **slanje ponovo kreće od prvog kruga** i tako cirkularno dok sva računala ne pošalju sve podatke. Ukoliko je za određeni krug definirano da podatke šalje više računala nego ih je preostalo u listi, podatke šalju samo računala koja su preostala u listi.

WiFi 4/5

- Ulaz
  - U prvom retku nalaze se dva prirodna broja **N** i **D** (**1 ≤ N ≤ 1000, 1 ≤ D ≤ 50**), koji predstavljaju ukupan broj računala spojenih na usmjernik (**N**) i količinu podataka koje pojedino računalo šalje u nekom krugu (**D**).
  - U drugom retku nalazi se N prirodnih brojeva P_i (**$1 \le P_i \le 1000$**), koji predstavljaju ukupnu količinu podataka koje i-to računalo treba poslati usmjerniku, pri čemu je prvo računalo na ulazu ujedno i prvo računalo u listi prioriteta za slanje, dok je posljednje računalo na ulazu ujedno i posljednje računalo u listi prioriteta.

WiFi 5/5

Školsko natjecanje
Algoritmi – 2012
Srednja škola

- Ulaz (nastavak)
  - U trećem retku nalazi se prirodan broj **K** (**1 ≤ K ≤ 1000**), koji predstavlja ukupan broj definiranih krugova slanja.
  - U četvrtom retku nalazi se **K** prirodnih brojeva R_j (**1 ≤ R_j ≤ N**), koji predstavljaju broj računala s početka liste prioriteta koji u j-tom krugu šalju.
- Izlaz
  - U prvom i jedinom retku potrebno je ispisati jedan prirodan broj koji označava **ukupan broj krugova** potrebnih da sva računala pošalju svoje podatke.

Vrijeme

Primjeri testnih podataka

ULAZ	ULAZ	ULAZ
3 1 8 5 5 2 3 2	4 10 45 30 42 10 4 2 1 3 3	6 6 70 10 4 32 40 21 4 3 5 1 2
IZLAZ	**IZLAZ**	**IZLAZ**
8	7	13

WiFi – rješenje 1/2

```
#include<stdio.h>
int main(){
    int i, j, n, d, k;
    int p[1000], r[1000];
    int prvih, krug = 0;

    scanf("%d %d", &n, &d);
    for(i=0; i<n; i++)
        scanf("%d", &p[i]);

    scanf("%d", &k);
    for(i=0; i<k; i++)
        scanf("%d", &r[i]);
```

wifi.c

WiFi – rješenje 2/2

```
for(i=1; i<=k; i++){
    prvih = r[i-1];
    for(j=0; j<n; j++){
        if(p[j] > 0){
            p[j] -= d;    prvih--;
        }
        if(prvih == 0)    break;
    }
    if(prvih == r[i-1])   break;
    krug++;
    i = i % k;
}
printf("%d\n", krug);
system("PAUSE");          return 0;
}
```

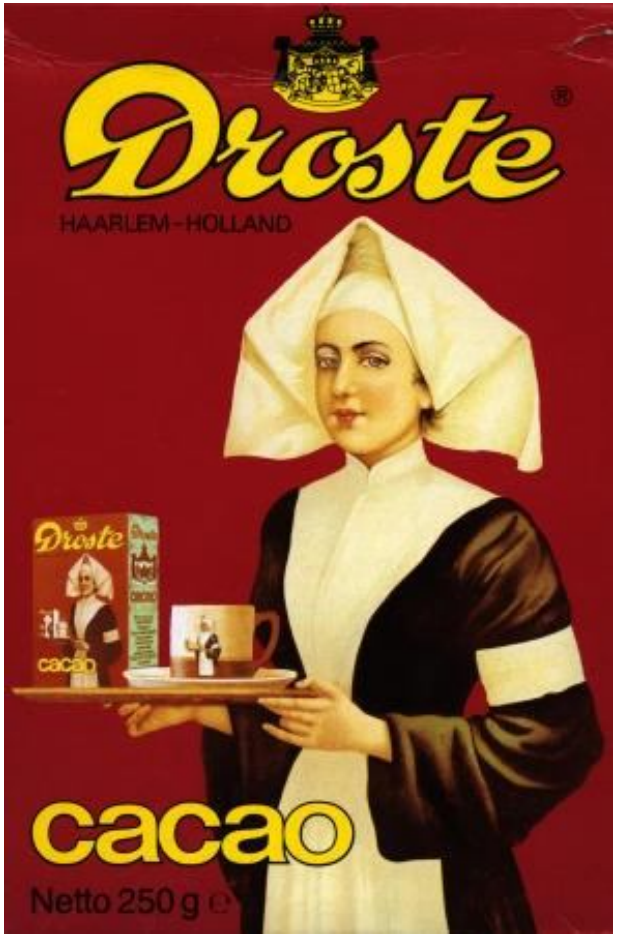
```
3 1
8 5 5
2
3 2
8
Press any key to continue . . .
```

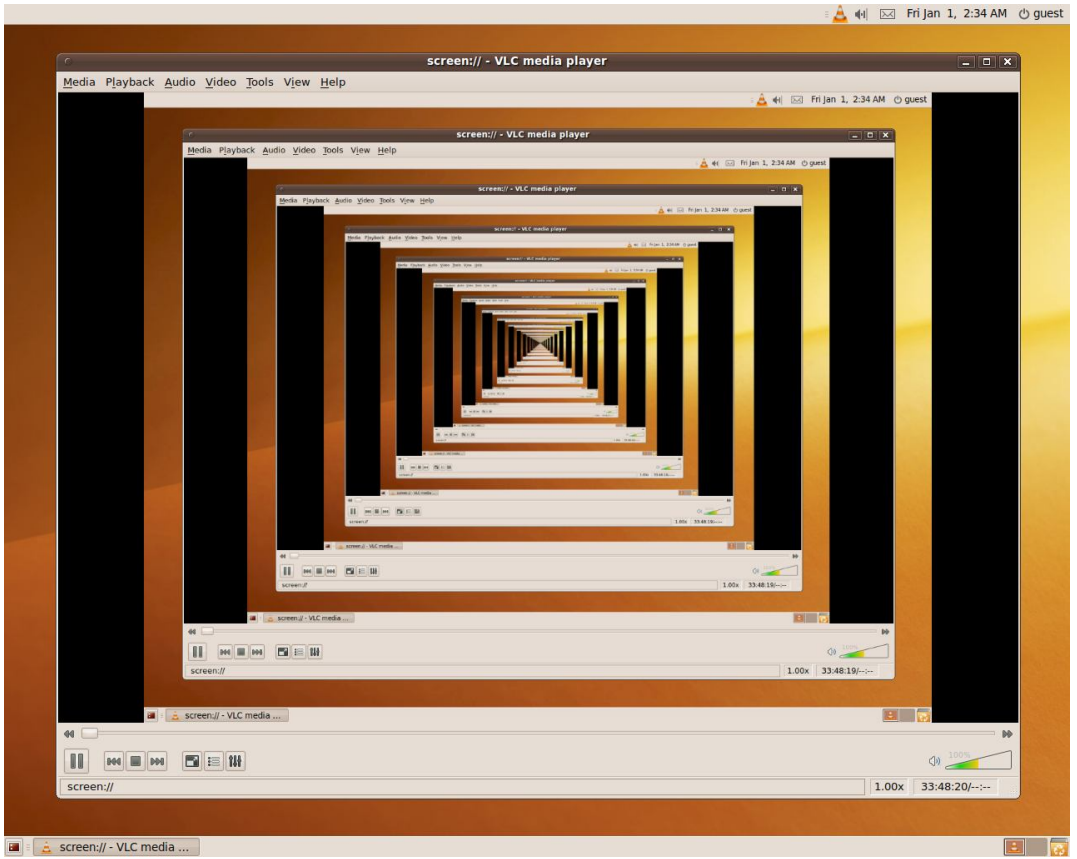
```
4 10
45 30 42 10
4
2 1 3 3
7
Press any key to continue . . .
```

```
6 6
70 10 4 32 40 21
4
3 5 1 2
13
Press any key to continue . . .
```

wifi.c

Rekurzija


Droste
HAARLEM - HOLLAND
cacao
Netto 250 g


screen:// - VLC media player
Media Playback Audio Video Tools View Help
Fri Jan 1, 2:34 AM guest
screen://
1.00x
33:48:20/--:--
screen:// - VLC media ...

Rekurzija: Lijeni zombiji


Mačka


Čovjek

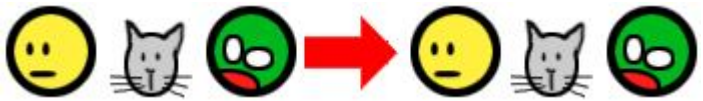

Čovjek koji je postao zombi

Zombiji su lijeni i ugristi će samo čovjeka neposredno do sebe

Ugrizeni čovjek (novi zombi) ugristi će svog susjeda i tako proširiti zarazu

Zombiji ne grizu mačke

Dok je mačka između čovjeka i zombija, čovjek je siguran

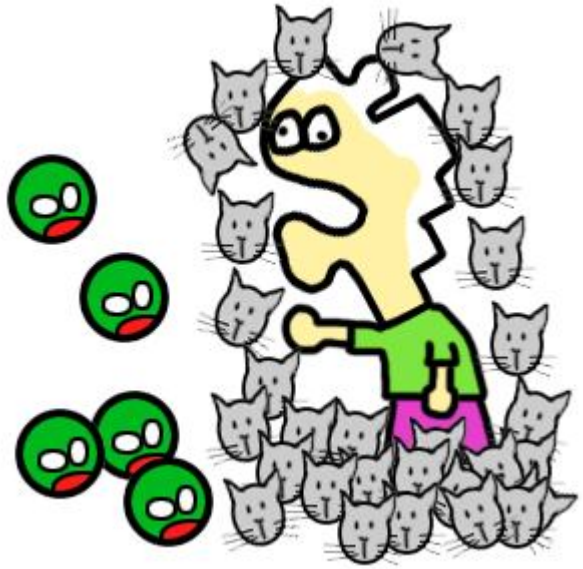


Prilagođeno s: Al. Sweigart, Recursion Explained with the Flood Fill Algorithm (and Zombies and Cats)
http://inventwithpython.com/blog/2011/08/11/recursion-explained-with-the-flood-fill-algorithm-and-zombies-and-cats

http://en.wikipedia.org/wiki/Zombie_(fictional)

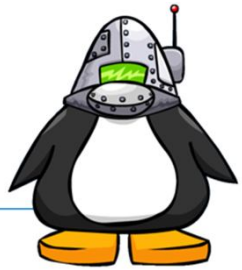
Rekurzija: Lijeni zombiji

Želite biti sigurni od zombija?
Okružite se mačkama!



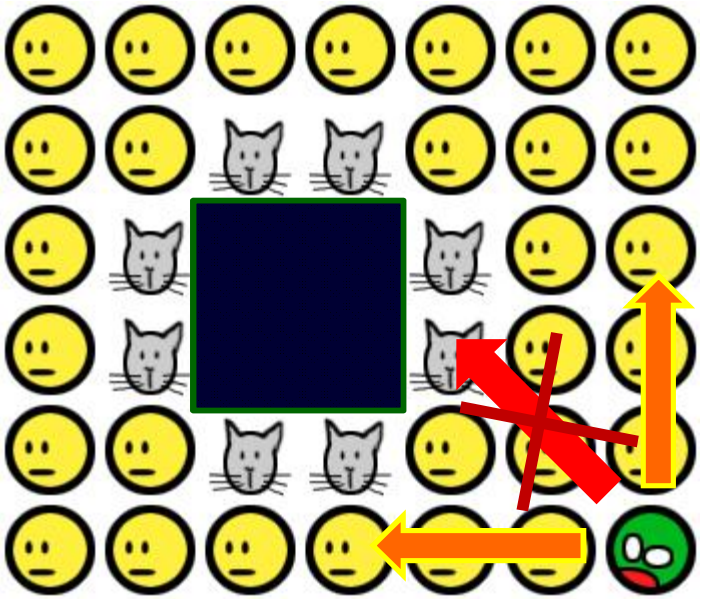
Prilagođeno s: Al. Sweigart, Recursion Explained with the Flood Fill Algorithm (and Zombies and Cats)
http://inventwithpython.com/blog/2011/08/11/recursion-explained-with-the-flood-fill-algorithm-and-zombies-and-cats

Rekurzija: Lijeni zombiji



Zamislimo dvodimenzionalno polje koje se sastoji od ljudi, mačaka i jednog zombija.

Neki ljudi su sa sve četiri strane okruženi mačkama i oni su sigurni.



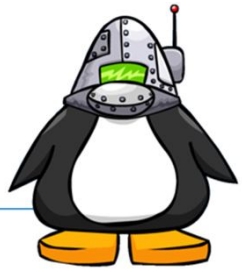
Prilagođeno s: Al. Sweigart, Recursion Explained with the Flood Fill Algorithm (and Zombies and Cats)
http://inventwithpython.com/blog/2011/08/11/recursion-explained-with-the-flood-fill-algorithm-and-zombies-and-cats

Rekurzija: Lijeni zombiji

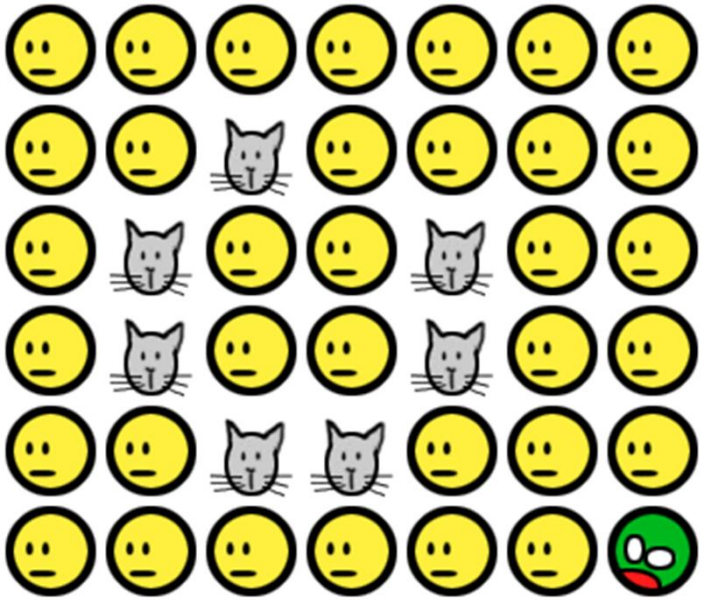
Ako je neka od mačaka pobjegla…

Prilagođeno s: Al. Sweigart, Recursion Explained with the Flood Fill Algorithm (and Zombies and Cats)
http://inventwithpython.com/blog/2011/08/11/recursion-explained-with-the-flood-fill-algorithm-and-zombies-and-cats

Rekurzija: Lijeni zombiji



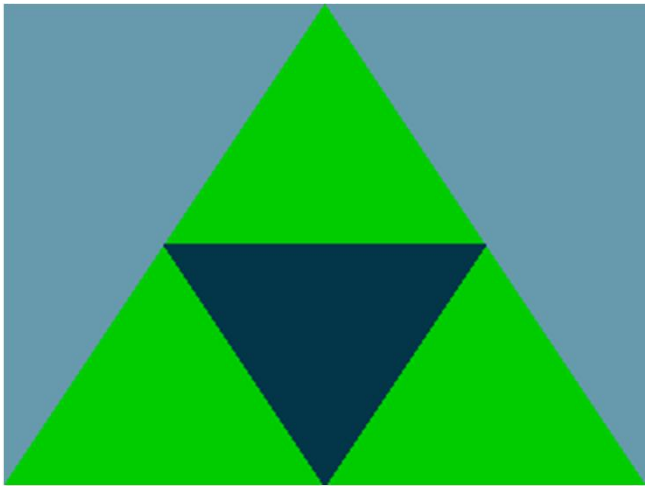
Područje više nije zatvoreno u potpunosti i svi ljudi će biti zaraženi.



Prilagođeno s: Al. Sweigart, Recursion Explained with the Flood Fill Algorithm (and Zombies and Cats)
http://inventwithpython.com/blog/2011/08/11/recursion-explained-with-the-flood-fill-algorithm-and-zombies-and-cats

Rekurzija: Trokut Sierpińskog

- Fraktal koji je opisao poljski matematičar Sierpiński 1915. godine
- Jedan od najjednostavnijih primjera fraktala
- Polazište je jednakostranični trokut



Prilagođeno s: Al. Sweigart, Recursion Explained with the Flood Fill Algorithm (and Zombies and Cats)
http://inventwithpython.com/blog/2011/08/11/recursion-explained-with-the-flood-fill-algorithm-and-zombies-and-cats

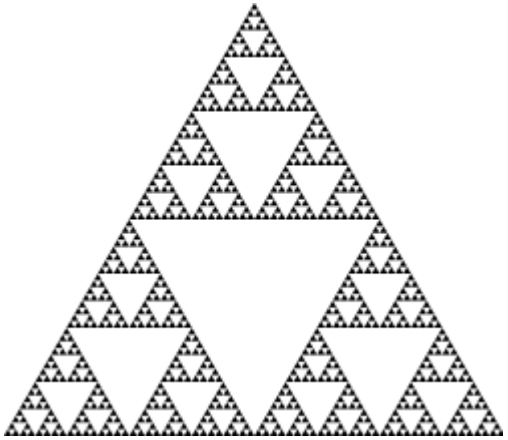
Rekurzivna funkcija

- Funkcija koja poziva samu sebe
- Mora imati konačan broj koraka
- Kod pisanja prvo treba vidjeti da li je problem pogodan za rekurzivno rješavanje
  - Da se svakim korakom veličina problema smanjuje
  - Na kraju se dolazi do jediničnog problema koji se jednostavno rješava
  - Konačno rješenje dobije se nizom jediničnih operacija
- Funkcionalno: rekurzija == petlja
  - Rekurzija se u programu može zamijeniti petljom
  - To ne znači da je takav program jednostavniji

Preuzeto iz skripte s predavanja: Programiranje 2, Sveučilište u Dubrovniku, 2011./12.

Prednost rekurzije

- Rješenje na jediničnoj razini može biti vrlo jednostavno
- Rješenje ukupnog problema se dobiva kombinacijom prethodno izračunatih (jednostavnih) rješenja
  - Na takav način se mogu (relativno jednostavno) riješiti vrlo složeni problemi
- Osnovni nedostatak je utrošak memorije koji kod dubljih rekurzija može biti jako velik



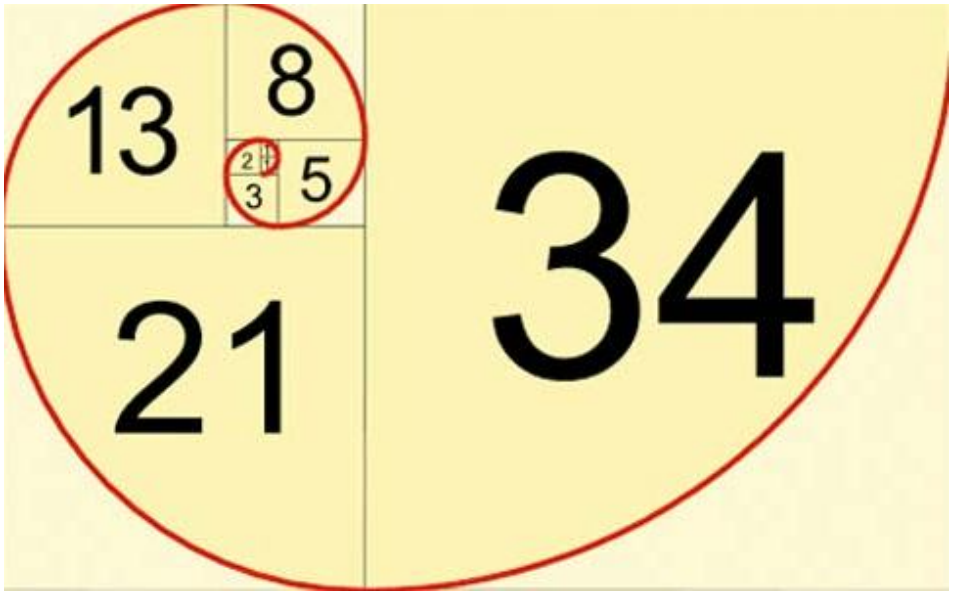
Preuzeto iz skripte s predavanja: Programiranje 2, Sveučilište u Dubrovniku, 2011./12.

Rekurzija – Fibonaccijevi brojevi

- 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, …
- Definicija niza Fibonaccijevih brojeva:

$$F(n) := \begin{cases} 0 & ako\ je\ n = 0; \\ 1 & ako\ je\ n = 1; \\ F_{n-1} + F_{n-2} & ako\ je\ n > 1. \end{cases}$$

- Rekurzivni algoritam:
  - $F_0 = 0$
  - $F_1 = 1$
  - $F_n = F_{n-1} + F_{n-2}$



Primjer – Fibonaccijevi brojevi

```
#include<stdio.h>
long fibonacci(long n){
    if(n <= 1)
        return n;
    else
        return (fibonacci(n-1)+fibonacci(n-2));
}

int main(){
    long fib;
    scanf("%ld", &fib);
    printf("%ld\n", fibonacci(fib));
    system("PAUSE");
    return 0;
}
```

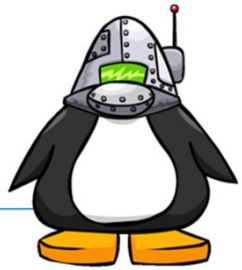
```
3
2
Press any key to continue . . .
```

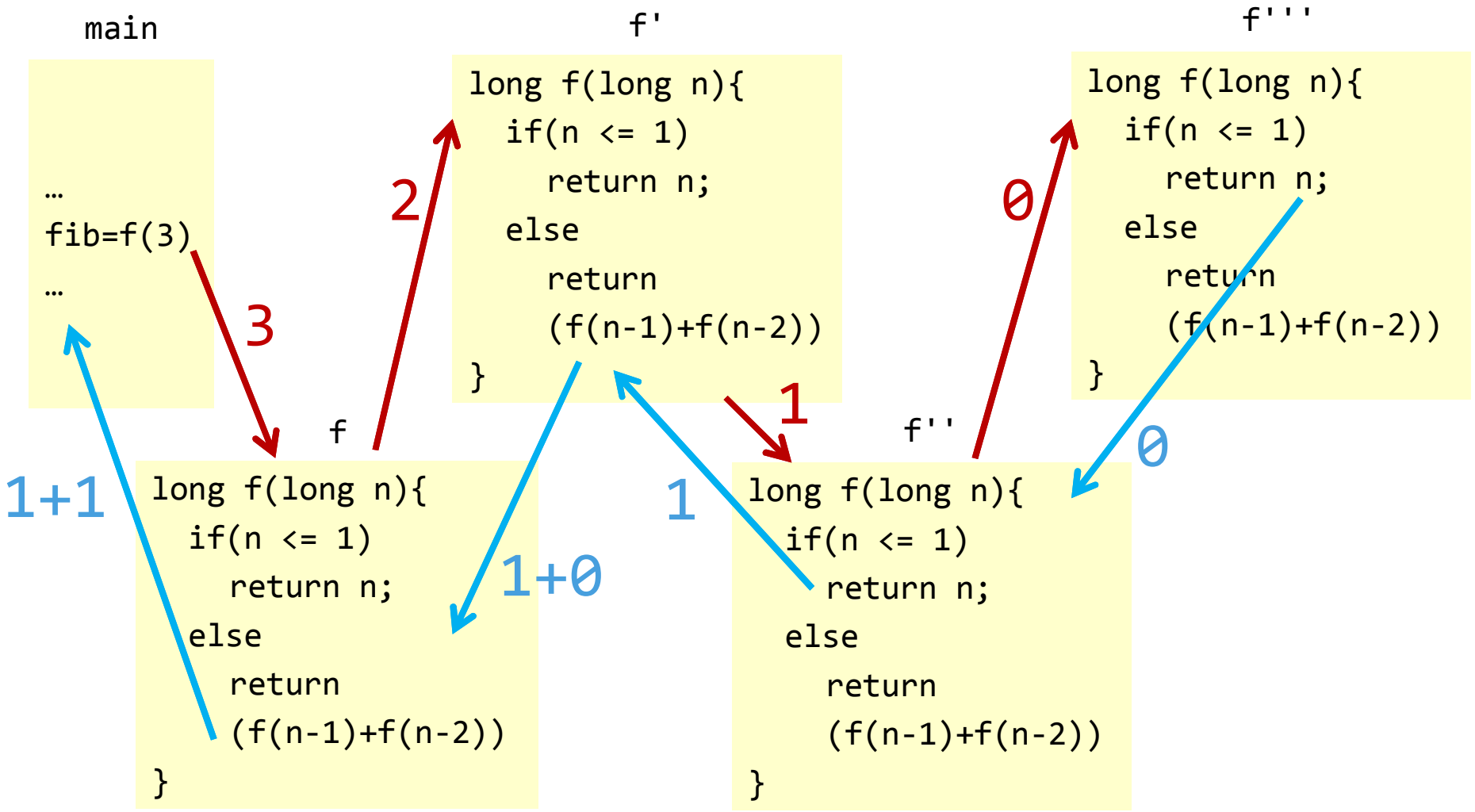
```
7
13
Press any key to continue . . .
```

```
20
6765
Press any key to continue . . .
```

fibonacci.c

Primjer – Fibonaccijevi brojevi





fibonacci.c

Gradovi 1/3

- Jedna mala država je imala velikih problema sa cestovnim prometom. Naime, sve ceste u državi su bile poprilično stare i u jako lošem stanju. Državna vlast odlučila je uložiti veliku količinu novca u preuređenje cesta. Voditelji projekta zaključili su da će čitav posao napraviti za jedan vikend i zaustaviti promet u cijeloj državi. Planirali su u subotu srušiti sve ceste, a u nedjelju izgraditi nove.
- Sve je išlo po planu, dok nije zapuhao jak vjetar i odnio nacrte na kojima su bile ucrtane ceste. Kako se bliži kraj dana, voditelje projekta hvata panika.

Gradovi 2/3

- Voditeljima je poznato da država ima **N gradova** i da je pomoću starih cesta bilo moguće **doputovati iz bilo kojeg grada u bilo koji drugi grad**. Uz to voditelji znaju koliko je cesta do sada izgrađeno.
- Pomozite voditeljima izračunati koliko još **najmanje cesta moraju izgraditi** do kraja dana, tako **da bude moguće putovati između bilo koja dva grada**.
- Ulaz
  - U prvom retku se nalazi prirodan broj **N** (**2 ≤ N ≤ 1000**), koji predstavlja broj gradova u državi.

Gradovi 3/3

Školsko natjecanje
Algoritmi – 2012
Srednja škola

Vrijeme

- Ulaz (nastavak)
  - U drugom retku se nalazi prirodan broj **K** (**2 ≤ K ≤ 1000**), koji predstavlja broj cesta koje su sagrađene.
  - U svakom od sljedećih **K** redaka nalaze se po dva broja x_i i y_i, (**1 ≤ x_i, y_i ≤ N**), koji znače da i-ta sagrađena cesta direktno povezuje gradove x_i i y_i, s tim da su gradovi numerirani od 1 do **N**, a sve ceste su dvosmjerne.
- Izlaz
  - U prvom i jedinom retku potrebno je ispisati najmanji broj cesta koje je potrebno sagraditi, tako da bude moguće **doputovati iz bilo kojeg grada u bilo koji drugi grad**.

Primjeri testnih podataka

ULAZ	ULAZ	ULAZ
3 3 3 2 2 1 1 3	6 5 3 1 5 3 4 5 2 6 3 4	8 6 6 3 2 1 7 4 5 7 4 5 3 4
IZLAZ	IZLAZ	IZLAZ
0	1	2

Gradovi – rješenje 1/2

```
#include<stdio.h>
int ceste[1000][1000] = {0}, posjecen[1000]={0};
int n;

void posjeti(int grad){
    int i;
    posjecen[grad] = 1;
    for(i=0; i<n; i++){
        if(grad != i && ceste[grad][i] == 1
                && posjecen[i] == 0)
            posjeti(i);
    }
}
```

gradovi.c

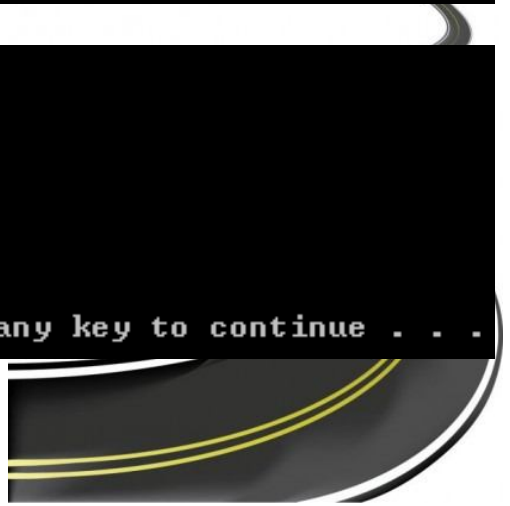
Gradovi – rješenje 1/2

```
int main(){
    int i, k, x, y, preostalo = 0;
    scanf("%d %d", &n, &k);
    for(i=0; i<k; i++){
        scanf("%d %d", &x, &y);
        ceste[x-1][y-1] = 1;
        ceste[y-1][x-1] = 1;
    }
    for(i=0; i<n; i++)
        if(posjecen[i] == 0){
            preostalo++;    posjeti(i);
        }
    printf("%d\n", preostalo-1);
    system("PAUSE");    return 0;
}
```

```
3
3
3 2
2 1
1 3
0
Press any key to continue . . .
```

```
6
5
3 1
5 3
4 5
2 6
3 4
1
Press any key to continue . . .
```

```
8
6
6 3
2 1
7 4
5 7
4 5
3 4
2
Press any key to continue . . .
```



gradovi.c