
Programiranje 2

doc. dr. sc. Goranka Nogo
PMF, Matematički odsjek

Kontakt

- ured: 228, drugi kat
 - e-mail: `nogo@math.hr`
 - konzultacije:
 - ponedjeljak, 10:00-12:00
 - neki drugi termin, uz prethodni dogovor putem e-maila.
 - Na konzultacije je potrebno donijeti vlastite zabilješke s predavanja.
-

Osnovna pravila „lijepog” ponašanja

Molim:

- ne ometajte pričom izvođenje nastave
- ne kasnite na predavanja
- održavajte red u predavaonici (ne ostavljajte iza sebe plastične boce, papire, ...)
- utišajte i pospremite mobitele
- isključite računala
- ...

Preddiplomski studij - računarski kolegiji

- Programiranje 1
 - Programiranje 2
 - Strukture podataka i algoritmi
 - Računarski praktikum 1
-

Ishodi učenja

- Po uspješnom završetku kolegija student/ica može:
 - opisati koncept rekurzije i navesti primjere korištenja
 - usporediti iterativna i rekurzivna rješenja osnovnih problema
 - opisati najčešće primjene i napisati jednostavne programe koji koriste sljedeće strukture podataka: polja, stringove i vezane liste
 - usporediti prednosti i mane statičkog i dinamičkog načina implementacije struktura podataka
-

Ishodi učenja (2)

- Po uspješnom završetku kolegija student/ica može:
 - odabrati prikladnu strukturu podataka za modeliranje danog problema
 - oblikovati, analizirati i implementirati brze algoritme sortiranja (Quicksort, Mergesort).
-

Pregled sadržaja kolegija

- Funkcije
 - Struktura programa
 - Dvodimenzionalna polja
 - Pokazivači
 - Strukture
 - Datoteke
 - Preprocesor
 - Standardna C biblioteka
-

Literatura

- Zabilješke s predavanja i vježbi te popratni materijali dostupni na službenim stranicama kolegija.
- B. W. Kernighan, D. M. Ritchie: *The C Programming Language*, 2nd Edition. Prentice Hall, 1988.
- ...



Način polaganja ispita

- Elementi ocjenjivanja:
 - prvi kolokvij
 - drugi kolokvij.
 - Ukupan broj bodova na prvom kolokviju je najmanje 40, a na drugom najmanje 60 (oba kolokvija mogu imati bonus bodove).
-

Način polaganja ispita (2)

- Za prolaz kolegija je potrebno sakupiti barem 45 bodova **na redovitim kolokvijima** (prvi i drugi zajedno ili popravni).
- Pri tome na barem jednom zadatku, na prvom ili drugom kolokviju, treba sakupiti najmanje 80 % mogućih bodova.

Način polaganja ispita (3)

- Napomene.
 - Studenti/ce koji polože kolegij, a nisu zadovoljni ocjenom, mogu odgovarati usmeno.
 - Nastavnik ima pravo pozvati studenta/icu na ispit.
- Više detalja o načinu polaganja ispita možete naći na službenim web stranicama kolegija:
<https://web.math.pmf.unizg.hr/nastava/prog2/>



Pitanja?

Funkcije

Primjer funkcije: Fibonaccijevi brojevi

- Rekurzivna relacija:

- $f_0 = 0, f_1 = 1,$

- $f_n = f_{n-1} + f_{n-2}, n \geq 2.$

- Eksplicitni oblik:

- $f_n = \frac{1}{\sqrt{5}} \left(\frac{1+\sqrt{5}}{2} \right)^n - \frac{1}{\sqrt{5}} \left(\frac{1-\sqrt{5}}{2} \right)^n, n \geq 0.$

- Moguća implementacija: \Rightarrow

Fibonaccijski brojevi: iterativna verzija

```
int fibonacci(int n) /* n ≥ 0 */
{
    int i;
    int f, f0 = 0, f1 = 1;
    if(n == 0) return 0;
    else if(n == 1) return 1;
        else
            for(i = 2; i <= n; i++){
                f = f0 + f1;
                f0 = f1;
                f1 = f;}
    return f;
}

printf("%d\n", fibonacci(4));
printf("%d\n", fibonacci(4));
printf("%d\n", fibonacci(50));
```

Načini prijenosa argumenata

- Načini prijenosa (predavanja) argumenata prilikom poziva funkcije:
 - prijenosom vrijednosti argumenata – *call by value*
 - prijenosom adresa argumenata – *call by reference*.



Prijenos vrijednosti argumenata

```
#include <stdio.h>

void kvadrat(int x, int y)
{
    y = x * x;
    printf("Unutar funkcije: \ x
= %d, y = %d.\n", x, y);
}
```

```
int main(void)
{
    int x = 3, y = 5;

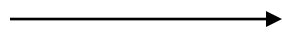
    printf("Prije poziva:
x = %d, y = %d.\n", x, y);
    kvadrat(x, y);
    printf("Nakon poziva: \
x = %d, y = %d.\n", x, y);
    return 0;
}
```

Prijenos vrijednosti argumenata (2)

- Funkcija `main`:

x

y



- Nakon poziva

x

y

- Poziv funkcije

x

y



- Izvođenje funkcije: $y = x * x$;

x

y

Prijenos adrese argumenata

```
#include <stdio.h>

void kvadrat(int *x, int *y)
{
    *y = *x * *x;
    printf("Unutar funkcije: \
x = %d, y = %d.\n", *x, *y);
}
```

```
int main(void)
{
    int x = 3, y = 5;

    printf("Prije poziva: \
x = %d, y = %d.\n", x, y);
    kvadrat(&x, &y);
    printf("Nakon poziva: \
x = %d, y = %d.\n", x, y);
    return 0;
}
```

Prijenos adrese argumenta (2)

- Funkcija `main`:

x

3

 0012FED4

y

5

 0012FEC8

- Nakon poziva

x

3

 0012FED4

y

5 9

 0012FEC8

- Poziv funkcije

&x

0012FED4

&y

0012FEC8



- Izvođenje funkcije: `*y = *x * *x;`

&x

0012FED4

&y

0012FEC8

Rekurzivne funkcije

- Osnovna ideja: funkcija (procedura) poziva samu sebe ili, preciznije
 - u definiciji funkcije koristimo funkciju koju definiramo.
 - Rekurzivna rješenja su u pravilu
 - kraća (imaju elegantniji zapis).
 - Mogu biti
 - memorijski
 - vremenski zahtjevnija.
 - Uvijek moraju postojati osnovni slučajevi koji se izvršavaju nerekurzivno (inicijalizacija rekurzije).
-

Primjer: najveći element polja

```
int T(int a[ ], int l, int d){
    int max1, max2;

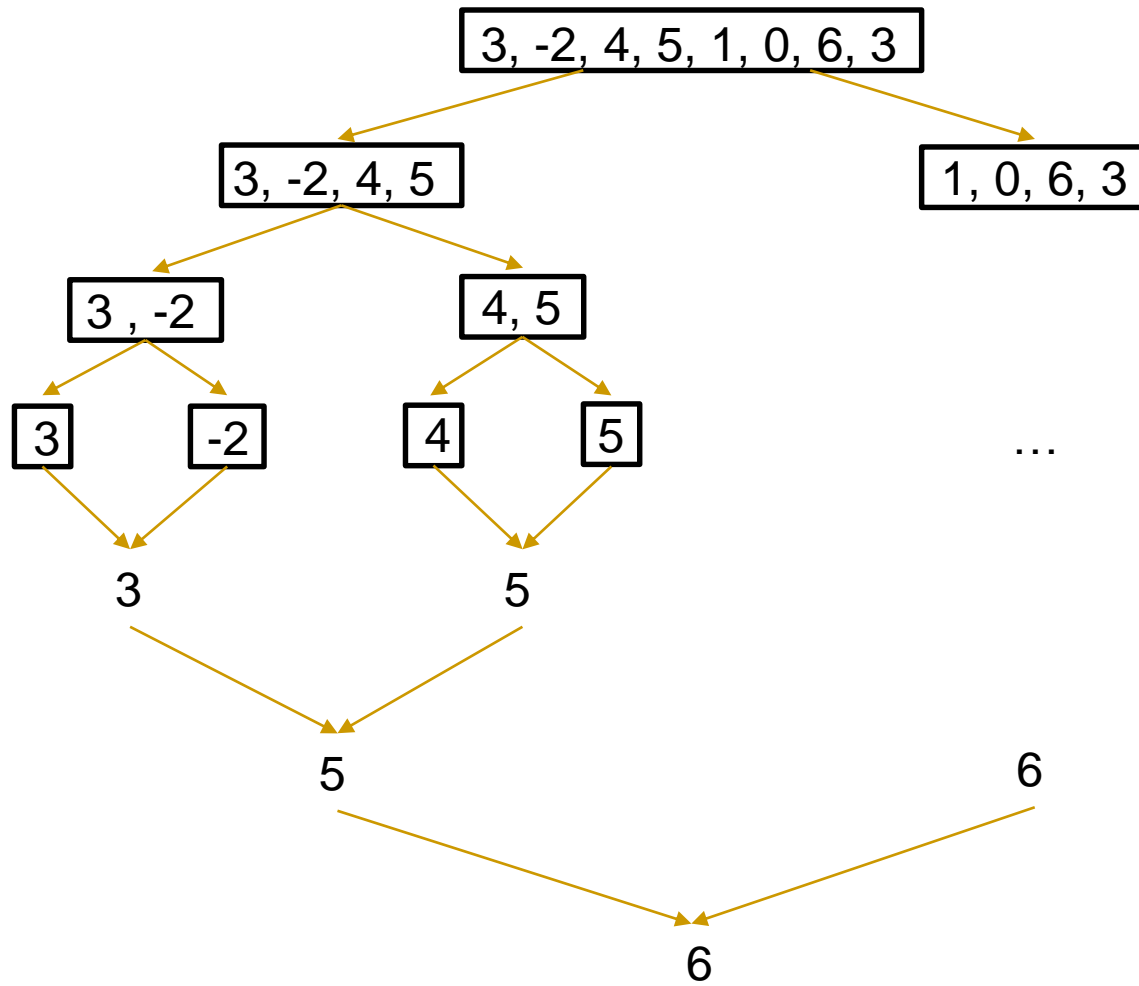
    if (l == d)
        return a[l];
    else{
        max1 = T(a, l, (d + l) / 2);
        max2 = T(a, (d + l) / 2 + 1, d);

        if (max1 > max2)
            return max1;
        else
            return max2; } }
```

Primjer poziva funkcije T:

```
int najveci;
int T[100];
...
najveci = T(a, 0, 99)
```

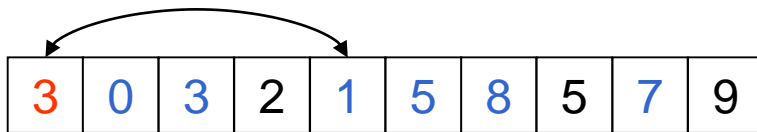
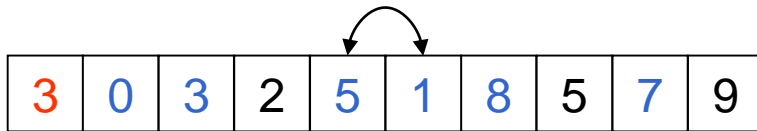
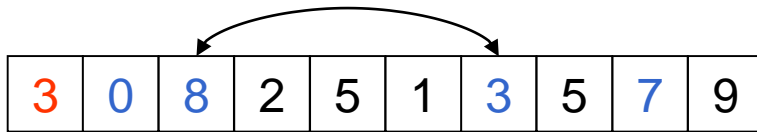
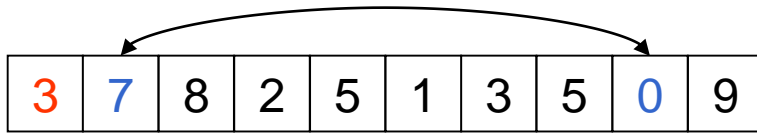
Primjer: najveći element polja



Primjer: algoritam Quicksort

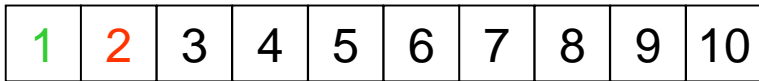
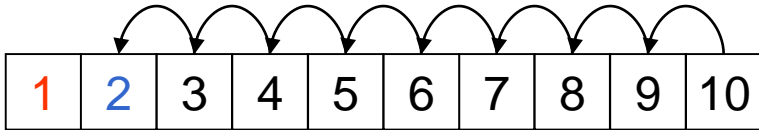
- Odabere se jedan element niza, tzv. **ključni element**.
 - Preslože se elementi niza tako da
 - ključni element dođe na njegovo pravo mjesto u sortiranom nizu
 - se lijevo od ključnog elementa nalaze svi elementi niza koji su manji (ili jednaki) od njega
 - se desno nalaze svi elementi niza koji su veći od njega.
 - Rekurzivno se sortiraju podniz manjih, odnosno podniz većih elemenata.
-

Quicksort (2)

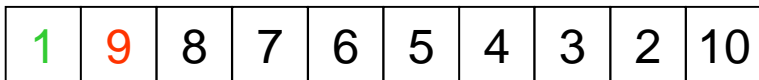
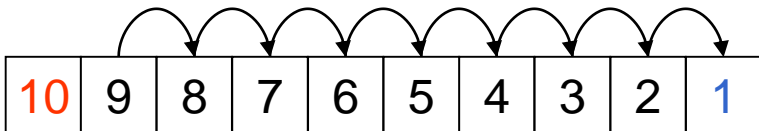


Složenost

- Prosječna složenost: $\Theta(n \log_2 n)$.
- Složenost u najgorem slučaju: $O(n^2)$.



itd, ili



- Autor: C. A. R. Hoare, 1961.

Funkcija swap

```
#include <stdio.h>
```

```
/* Sortiranje algoritmom Quicksort.
```

```
  x[l] je ključni element – dovodimo ga na pravo mjesto u polju. */
```

```
void swap(int *a, int *b){
```

```
    int temp;
```

```
    temp = *a;
```

```
    *a = *b;
```

```
    *b = temp;
```

```
    return;
```

```
}
```

Funkcija quick_sort

```
void quick_sort(int x[ ], int l, int d){
    int i, j;

    if (l < d) {
        i = l + 1;
        j = d;
        /* Mora se ispitati i slučaj i == j. */
        while (i <= j) {
            while (i <= d && x[i] <= x[l]) ++i;
            while (x[j] > x[l]) --j;
            if (i < j) swap(&x[i], &x[j]);
        }
    }
}
```

Funkcija `quick_sort` - nastavak

```
    if (l < j) swap(&x[j], &x[l]);  
    quick_sort(x, l, j - 1);  
    quick_sort(x, j + 1, d);  
}  
  
return;  
}
```

Funkcija main

```
int main(void) {  
    int i, n;  
    int x[] = {42, 12, 55, 94, 18, 44, -67};  
  
    n = 7;  
    quick_sort(x, 0, n - 1);  
  
    printf("\n sortirano polje x\n");  
    for (i = 0; i < n; ++i) {  
        printf(" x[%d] = %d\n", i, x[i]);  
    }  
    return 0;  
}
```
