SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

**Matija Belec**

PRIMJENA REKURZIVNIH ALGORITAMA U PROGRAMIRANJU

ZAVRŠNI/DIPLOMSKI rad

Varaždin, 2016.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Matija Belec

Matični broj: 39912/11

Studij: Informacijski sustavi

PRIMJENA REKURZIVNIH ALGORITAMA U PROGRAMIRANJU

ZAVRŠNI/DIPLOMSKI RAD

Mentor:

Prof.dr.sc. Danijel Radošević

Varaždin, rujan 2016.

**Sadržaj**

[1. Uvod 1](#_Toc458861075)

[2. Matematička indukcija 2](#_Toc458861076)

[3. „Školski“ primjeri rekurzija 3](#_Toc458861077)

[3.1. Faktorijel 3](#_Toc458861078)

[3.2. Fibonaccijev niz 3](#_Toc458861079)

[4. Algoritmi podijeli pa vladaj 4](#_Toc458861080)

[4.1. Sortiranje 4](#_Toc458861081)

[4.1.1. Quick sort 4](#_Toc458861082)

[4.1.2. Merge sort 4](#_Toc458861083)

[4.2. Binarno pretraživanje 4](#_Toc458861084)

[5. Pohlepni algoritmi 5](#_Toc458861085)

[5.1. Vraćanje ostatka novca 5](#_Toc458861086)

[5.2. Rastavljanje broja na egipatske razlomke 5](#_Toc458861087)

[6. Metoda pretraživanja s vraćanjem 6](#_Toc458861088)

[6.1. Problem N-kraljica 6](#_Toc458861089)

[7. Literatura 7](#_Toc458861090)

1. Uvod

T.

1. Matematička indukcija

T.

1. „Školski“ primjeri rekurzija

Faktorijel broja je matematička funkcija kojom se izračunava umnožak brojeva od 1 do odabranog broja n. Na primjer, iznosi 120:

U nastavku slijede klasični jednostavniji primjeri rekurzija.

* 1. Faktorijel

Prvi klasičan primjer rekurzije u ovom radu je izračun faktorijela broja. Primjer jednostavnog programskog koda koji koristi rekurzivni pristup je prikazan sljedećim programskim kodom:

// g++ faktorijel.cpp -o faktorijel.out

#include <iostream>

long long faktorijel(unsigned int n) {

return (n>1 ? n \* faktorijel(n-1) : 1);

}

int main(){

int n;

do std::cin >> n; while(n<1);

std::cout << faktorijel(n) << std::endl;

return 0;

}

Također, faktorijel je moguće programski izračunati i na iterativni način kako slijedi u donjem primjeru programskog koda.

// g++ faktorijel\_iter.cpp -o faktorijel\_iter.out

#include <iostream>

int main(){

int n;

do std::cin >> n; while(n<1);

long long f = 1;

for(int i=2; i<=n; i++) {

f \*= i;

}

std::cout << f << std::endl;

return 0;

}

* 1. Fibonaccijev niz

Drugi jednostavniji primjer rekurzije jest fibonaccijev niz. Fibonaccijev niz određen je sljedećom matematičkom formulom:

Prema navedenoj formuli možemo jednostavno izraditi programski kod koji slijedi u nastavku.

// g++ fibo.cpp -o fibo.out

#include <iostream>

long long fibo(unsigned int n) {

return (n > 1 ? fibo(n-1) + fibo(n-2) : n);

}

int main(){

int n;

do std::cin >> n; while(n<0);

std::cout << fibo(n) << std::endl;

return 0;

}

1. Algoritmi podijeli pa vladaj

T.

* 1. Sortiranje

U programskim algoritmima često se dolazi do upotrebe određene vrste pretraživanja ili sortiranja. Kod sortiranja polja se radi pretraživanje polja ili dijela polja unutar kojeg tražimo elemente koji su veći ili manji od prethodno određenog elementa sa kojim ih uspoređujemo sa ciljem da se nakon procesa sortiranja, u spomenutom polju elemenata nalaze isti elementi ali sortirani redom od najmanjeg do najvećeg (ili obrnuto). Postoji mnogo vrsta sortiranja među kojima su više poznatiji:

* Sortiranje izborom,
* Sortiranje umetanjem,
* Sortiranje zamjenom,
* Quick sort,
* Sortiranje spajanjem (Merge sort).

Dok su prvi spomenuti oblici sortiranja koji se uglavnom implementiraju iterativnim pristupom, zadnja dva spomenuta, quick sort i sortiranje spajanjem, su dva najpoznatija algoritma soritiranja koja koriste metodu „podijeli pa vladaj“ (negdje se spominje naziv „podijeli i ovladaj“).

Metoda „podijeli pa vladaj“ radi upravo to što se i iz samog naziva može isčitati: složeniji problem dijeli na više manjih problema od kojih se nadalje svaki zasebno riješava. Dobivanjem riješenja manjih problema, nadalje se dolazi do riješenja složenijeg (prvozadanog) problema.

U nastavku se nalaze algoritamske reprezentacije i programska riješenja za oba navedena alogoritma za sortiranje koji koriste navedenu metodu „podijeli pa vladaj“.

* + 1. Quick sort

Algoritam sortiranja quick sort je vrlo popularan u većini programskih jezika kao standardni način sortiranja elemenata unutar određenog polja, odnosno liste. Uglavnom se radi o jednoj od „poboljšanih“ varijanta quick sort-a gdje se polje, umjesto na standardnih 2 dijela, dijeli na 3 dijela. U ovom radu ću se ograničiti na klasičan algoritam quick sort sortiranja i koristit ću samostalno napisan programski kod u programskom jeziku C/C++ kao primjer programskog riješenja navedenog sortiranja.

Za početak, ukratko o samom quick sort sortiranju. Navedeno sortiranje koristi sljedeće opisani algoritam:

1. Odredi „pivot“ element,
2. Podijeli polje na dva dijela tako da se u prvi (lijevi) dio polja prebace svi elementi manji od vrijednosti „pivot“ elementa, dok se preostali elementi postave u drugi (desni) dio polja,
3. Ponovi korak 2 za lijevi dio polja, te zatim za desni dio polja sve dok svaki dio polja ima elemenata.

Nakon što se korak 2 ponovi nad svakim dijelom polja rekurzivno, te se u dijelu polja nađe samo jedan element, taj element je već sortiran i nalazi se na točnom indeksu na kojem se treba nalaziti u sortiranom polju. Time se nakon rekurzivnog prebacivanja manjih elemenata u lijevi dio polja, a preostalih u desni dio polja dobiva sortiranje „pivot“ elementa u svakom koraku.

U nastavku se nalazi primjer programskog riješenja.

void qs(int\* p, int a, int b) {

int i=a, j=b;

int m=p[a], s;

while(i<=j) {

while(p[i] < m) i++;

while(p[j] > m) j--;

if(i<=j) {

s=p[i]; p[i]=p[j]; p[j]=s;

i++; j--;

}

}

if(a<j) qs(p, a, j);

if(i<b) qs(p, i, b);

}

//qs(p, 0, n-1)

Na samom kraju programskog riješenja vidljiv je i sam poziv navedene funkcije qs. Prvi argument funkcije je polje (točnije pokazivač na prvi element polja), drugi argument je indeks prvog elementa polja (ili dijela polja koji se želi sortirati), te posljednji, treći argument je indeks zadnjeg elementa polja (ili dijela polja koji se želi sortirati).

* + 1. Merge sort

T.

* 1. Binarno pretraživanje

T.

1. Pohlepni algoritmi

T.

* 1. Vraćanje ostatka novca

T.

* 1. Rastavljanje broja na egipatske razlomke

T.

1. Metoda pretraživanja s vraćanjem

T.

* 1. Problem N-kraljica

T.

1. Literatura
2. Barnett G, Del Tongo L (2008.) Data Structures and Algorithms
3. Drozdek A (2013.) Data Structures and Algorithms in C++
4. Knuth D E (?) Art of Computer Programming, Volume 3
5. Sedgewick R, Flajolet P (2013.) An Introduction to the Analysis of Algorithms.
6. Skiena S S (2008.) The Algorithm Design Manual