

Pojam rekurzije

Pojam rekurzije potiče iz matematike i ima veliku primjenu u programiranju.

U matematičkom smislu rekurzija predstavlja definisanje problema uz pomoć samog tog problema. U matematici postoji veliki broj primjera rekurzije, kao što su Fibonačijevi brojevi koji se definišu na sljedeći način:

$$F(n) = F(n-1) + F(n-2).$$

Generalni alogritam koji rješava probleme rekurzijom može se opisati u tri koraka:

1. Trivijalni slučaj (kada se prekida proces izračunavanja)
2. Izvršavanje akcija koje nas vode prema trivijalnom slučaju
3. Rekurzivni poziv.

Primjer. Skicirati algoritam za izračunavanje faktoriijela prirodnog broja N na rekurzivan način.

Faktoriijel broja N u oznaci N! računa se po formuli $N! = N * (N-1) * \dots * 2 * 1$, i važi $0! = 1$.

Odgovor.

1. Trivijalni slučaj je $N=0$ i u ovom slučaju izlazimo iz rekurzije
2. Mijenjamo N tako što ga smanjujemo za 1 čime se približavamo trivijalnom slučaju (smanjivanjem N za 1 idemo ka nuli)
3. Rekurzivni poziv za N-1.

Rekurzija se implementira preko funkcija i predstavlja situaciju u kojoj funkcija poziva samu sebe.

```
void recursion() {  
    ...  
    recursion(); /* funkcija poziva samu sebe */  
    ...  
}  
  
int main() {  
    recursion(); /* poziv rekurzivne funkcije */  
}
```

U implementaciji rekurzije moramo biti oprezni da ne izazovemo beskonačnu petlju. Na primjer, ako bismo htjeli da implementiramo program koji računa zbir prvih n prirodnih brojeva, ovaj problem bismo mogli da definišemo rekurzivno na sljedeći način: suma prvih n brojeva je jednaka sumi prvih $n-1$ brojeva plus broj n . Ako ovu logiku implementiramo direktno dobijamo program koji je dat na sljedećem listingu. Međutim, ovaj program će ući u beskonačnu petlju, jer funkcija suma stalno poziva sama sebe.

```

#include
#include
int suma(int n);

int main()
{
    int n = 6;
    int rez = suma(6);
    printf("Suma prvih %d brojeva je %d \n", n,
rez);
    return 0;
}

int suma(int n){
    int rez;
    rez = n + suma(n-1);
    return rez;
}

```

Da bi se izbjegla beskonačna petlja u rekurzivnom pozivu, mora se uvesti trivijalni slučaj, odnosno slučaj koji predstavlja izlaz iz rekurzije. U primjeru sabiranja prvih n prirodnih brojeva, za slučaj $n=1$ zbir prvih n prirodnih brojeva ne zahtijeva sabiranje već odmah možemo vratiti rezultat. U funkciji `suma` dodali smo slučaj za $n=1$ koji ne ulazi u rekurziju već samo vraća vrijednost funkcije i to nam je trivijalni slučaj. Na ovaj način petlja poziva funkcije `suma` se zaustavlja u momentu kada n dobije vrijednost 1.

```

#include
#include
int suma(int n);
int main()
{
    int n = 6;
    int rez = suma(6);
    printf("Suma %d brojeva je %d \n", n, rez);
    return 0;
}
int suma(int n){
    int rez;
    if(n==1)
        return 1;
    rez = n + suma(n-1);
    return rez;
}

```

Rekurzija – zadaci

1. Napisati funkciju `int pow_rek(int a, int b)` koja rekurzivno izračunava b-ti stepen broja a. U funkciji main za dva unijeta prirodna broja ispisati rezultat dobijen pozivom te funkcije.
2. Napisati funkciju `int zbir_cif_rek(int a)` koja rekurzivno izračunava zbir cifara broja a. U funkciji main za unijeti prirodni broj ispisati rezultat dobijen pozivom te funkcije.

3. Napisati funkciju `int dec2bin(int n)` koja rekurzivno prevodi zapis broja iz sistema sa osnovom 10, u odgovarajući zapis u sistemu sa osnovom 2. U glavnoj funkciji testirati napisanu funkciju, podrazumijevati da korisnik unosi broj za koji se treba odraditi konverzija.
4. Napisati rekurzivnu funkciju `double f1(int n)` i iterativnu funkciju `double f1_i(int n)` koja za unijeto $n = 6$ izračunava sljedeći razlomački izraz:

$$6 + \frac{1}{5 + \frac{1}{4 + \frac{1}{3 + \frac{1}{2 + \frac{1}{1}}}}}$$

U glavnoj funkciji, za unijeto n , u dva reda ispisati rezultat, u prvom iterativno dobijen, u drugom rekurzivno.

5. Napisati rekurzivnu funkciju `double f2(int n)` i iterativnu funkciju `double f2_i(int n)` koja za unijeto $n = 5$ izračunava sljedeći razlomački izraz:

$$1 + \frac{1}{2 + \frac{1}{3 + \frac{1}{4 + \frac{1}{5}}}}$$

U glavnoj funkciji, za unijeto n , u dva reda ispisati rezultat, u prvom iterativno dobijen, u drugom rekurzivno.