****

Tema 6

****

TEME

- Vezbanje funkcija
- Vezbanje algoritama
- Primena operatora
- Rekurzivne funkcije

PRIMER 1: Napisati funkciju koja racuna srednju vrednost cifara visecifrenog broja. Napisati main funkciju u kojoj se data funkcija testira.

KORACI RESAVANJA:

- Funkcija za aritmeticku sredinu
  - Povratna vrednost mora biti float
  - Parametar funkcije mora biti int
  - Obradjimo posebno slucaj ako je uneta 0
  - Obradimo slucaj kada je visecifren broj u pitanju primenjujuci algoritam da se odsecaju cifre koristeci operatore moduo i deljenje
- U mainu napisati poziv te funkcije ali da je parametar nesto sto korisnik unosi

FINALNA IMPLEMENTACIJA:

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

/* Funkcija racuna aritmeticku sredinu cifara datog celog broja. */
float aritmeticka_sredina(int x) {
  /* Aritmeticka sredina broja 0 je 0.
     To se obradjuje kao zaseban slucaj */
  if (x == 0)
    return 0;

  /* Deklaracija i inicijalizacija brojaca. */
  int zbir_cifara = 0;
  int broj_cifara = 0;

  /* Izracunava se apsolutna vrednost broja x kako bi program
     ispravno radio i za negativne brojeve. */
  x = abs(x);

  /* Sve dok ima neobradjenih cifara, na zbir se dodaje poslednja
     cifra, brojac cifara se uvecava za 1 i iz broja x se uklanja
```

```
     poslednja cifra. */
  while (x) {
    zbir_cifara += x % 10;
    broj_cifara++;
    x /= 10;
  }

  /* Kao povratna vrednost funkcije se vraca odgovarajuci
     kolicnik. */
  return (float) zbir_cifara / broj_cifara;
}

int main() {
  /* Deklaracija potrebne promenljive. */
  int x;

  /* Ucitavanje vrednosti broja x. */
  printf("Unesite broj: ");
  scanf("%d", &x);

  /* Ispis rezultata. */
  printf("Aritmeticka sredina: %.3f\n", aritmeticka_sredina(x));

  return 0;
}
```

OBJASNJENJE

- Objasniti kako bi u Pythonu ovo uradili
- .3f za float nije spominjano u C do sada
- U returnu se koristi konverzija u (float), isprobati sta bi bilo da to ne stavimo

PRIMER 2: Odredjivanje broja neparnih cifara u zapisu datog celog broja

KORACI RESAVANJA:

- Ovo ucenik moze da radi
- Funkcija za brojanje neparnih cifara
    - Povratna vrednost mora biti int
    - Parametar funkcije mora biti int
    - Dodajemo promeljive koje nam trebaju
    - Algoritam slican kao u prethodnom primeru
- U mainu napisati poziv te funkcije ali da je parametar nesto sto korisnik unosi sve dok ne unese broj 0, funkcija se u petlji poziva

FINALNA IMPLEMENTACIJA:

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

/* Funkcija odredjuje broj neparnih cifara u zapisu datog celog
   broja. */
int broj_neparnih_cifara(int x) {
  int brojac_neparnih = 0;
  int cifra;
  x = abs(x);

  while (x) {
    /* Izdvaja se poslednja cifra broja. */
    cifra = x % 10;

    /* Moze se izbeci koriscenje naredbe if pomocu narednog izraza.
       Naime, vrednost izraza cifra%2 je 1 kada je cifra neparna,
       odnosno 0 kada je parna. Tako ce na broj neparnih cifara
       biti dodata jednica ako je cifra neparna, a ako je parna
       bice dodata 0, sto jeste zeljeno ponasanje. */
    brojac_neparnih += (cifra % 2);
    x /= 10;
  }

  return brojac_neparnih;
}

int main() {
  /* Deklaracija potrebne promenljive. */
  int x;

  /* Ucitavanje brojeva sve do unosa broja nula i ispis
     broja neparnih cifara za svaki ucitani broj. */
  printf("Unesite cele brojeve:\n");
  while (1) {
    scanf("%d", &x);
    if (x == 0)
      break;

    printf("Broj neparnih cifara: %d\n", broj_neparnih_cifara(x));
  }

  return 0;
}
```

OBJASNJENJE

- Ne koristiti IF nego isprobati trik sa % operatorom

PRIMER 3: Napraviti funkciju kojom se racuna zbir uzastopnih prirodnih brojeva od 0 pa do tog broja.

KORACI RESAVANJA:

- Objasniti matematicki postupak, 0 + 1 + 2 + 3 + 4 + ..+ x = (x+1)*x/2
- Objasniti drugi pristup, zbir prvih x = zbir prvih (x-1) + x, a zbir prvih (x-1) = zbir prvih (x-2) + (x-1)...
- Potrebno da ucenik vidi patern i da razume da ce funkcija samu sebe da poziva

FINALNA IMPLEMENTACIJA:

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

int sum(int k) {
  if (k > 0) {
    return k + sum(k - 1);
  } else {
    return 0;
  }
}

int main() {
  int x;
  printf("Unesi broj do kog racunamo zbir:\n");
  scanf("%d", &x);
  int result = sum(x);
  printf("%d", result);
  return 0;
}
```

OBJASNJENJE

- Ispisati sa strane na steku sta je sve pozvano da se vidi da je stalno pozivana ista funkcija i da se vidi sta je ona vracala i kojim redosledom idu pozivi tih funkcija i ono sto one izracunaju

PRIMER 4: Napraviti funkciju kojom se racuna faktorijel nekog broja.

KORACI RESAVANJA:

- Ovo ucenik moze sam da uradi
- Objasniti matematicki postupak, sta je faktorijel
- Faktorijel od 0 je 1
- Potrebno da ucenik vidi patern i da razume da ce i ovde funkcija samu sebe da poziva

FINALNA IMPLEMENTACIJA:

```
#include<stdio.h>

int find_factorial(int);

int main()
{
   int num, fact;
   //Ask user for the input and store it in num
   printf("\nEnter any integer number:");
   scanf("%d",&num);

   //Calling our user defined function
   fact =find_factorial(num);

   //Displaying factorial of input number
   printf("\nfactorial of %d is: %d",num, fact);
   return 0;
}

int find_factorial(int n)
{
   //Factorial of 0 is 1
   if(n==0)
      return(1);

   //Function calling itself: recursion
   return(n*find_factorial(n-1));
}
```

OBJASNJENJE

- Objasniti i ovaj nacin zapisa, ako je main() pre funkcije koju koristi, onda se mora dodati pre maina samo deklaracija funkcije, a sama definicija posle maina()