Programski jezik I

– Primjeri sa 6. termina predavanja –

Funkcije. Funkcije sa nizovima i stringovima

1. Napisati program koji sadrži funkciju **`suma_podniza`** koja za niz cijelih brojeva **`x`** koji joj se prosljeđuje kao parametar (zajedno sa dužinom), računa sumu elemenata niza koji su veći od cijelog broja **`T`**. Korisnik unosi u glavnom programu niz i cio broj **`T`**, a zatim se poziva funkcija i štampa rezultat.

   Primjer: ako se unese niz **`x = [1 45 2 19 5]`** i **`T = 10`**, dobija se suma 64.

   Rješenje:

```c
#include <stdio.h>

int suma_podniza(int *, int, int);

int main()
{
    int N, x[200], T, i;
    printf("Unesite duzinu niza: ");
    scanf("%d", &N);

    printf("Unesite niz: ");

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

    printf("Unesite T: ");
    scanf("%d", &T);

    printf("Suma je: %d", suma_podniza(x,N,T));
}

int suma_podniza(int *x, int N, int T)
{
    int i, s = 0;
    for (i = 0; i < N; i++)
        if (x[i] > T)
            s += x[i];

    return s;
}
```

2. Napisati program, koji sadrži funkciju **`sum_array`** koja određuje sumu elemenata niza cijelih brojeva koji su djeljivi sa 2 ili sa 3. Funkcija takođe elemente niza koji su djeljivi sa 2 ili sa 3 duplira. Glavni program od korisnika traži unos niza, poziva funkciju i štampa odgovarajući sumu, kao i modifikovani niz.

Rješenje:

```c
#include <stdio.h>
int sum_array(int *, int);
int main()
{
    int i, x[20], N;
    printf("Unesite duzinu niza: ");
    scanf("%d", &N);
    printf("Unesite niz:");
    for(i = 0; i < N; i++)
        scanf("%d", &x[i]);
    printf("Trazena suma je: %d\n", sum_array(x, N));
    printf("Rezultujuci niz je:\n");
    for(i = 0; i < N; i++)
        printf("%d ", x[i]);
}
int sum_array(int *x, int N)
{
    int s = 0, i;

    for (i = 0; i < N; i++)
        if (x[i]% 2 ==0 || x[i]%3 == 0)
    {
        s += x[i];
        x[i] = 2*x[i];
    }
    return s;
}
```

3. Napisati program koji sadrži funkciju **binar**, koja pretvara prirodan broj **N** u njegovu binarnu reprezentaciju. Binarnu reprezentaciju funkcija smješta u string **s**. Glavni program od korisnika traži unos cijelog broja. Program poziva funkciju **binar** za unijeti broj, i zatim štampa rezultujući string.

   Rješenje: Prisjetimo se procedure za konverziju cijelog broja odgovarajući binarni zapis, posmatrajući $22_{(10)}$.

```
22/2 = 11 ostatak 0  /\
11/2 = 5  ostatak 1  ||
5/2  = 2  ostatak 1  ||
2/2  = 1  ostatak 0  ||
1/2  = 0  ostatak 1  ||
```

   Čitajući odozdo nagore, dobija se binarna reprezentacija dekadnog broja $22_{(10)}$ kao $10110_{(2)}$.

   Uradimo zadatak na teži način:

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

void binar(int, char *);

int main()
{
    int N;
    char s[64];
    puts("Unesite cio broj N:");
    scanf("%d", &N);

    binar(22, s);
    puts(s);
```

```c
}

void binar(int N, char *s)
{
    int cif, k = 0, pom, i;
    pom = N;

    while (pom != 0) // određujemo broj bita
    {
        pom /= 2;
        k++;
    }

    i = k;
    s[i--] = '\0';

    while(N != 0)
    {
        cif = N%2;
        if (cif == 1)
            s[i--] = '1';
        else
            s[i--] = '0';
        N /= 2;
    }
}
```

Dakle, funkcija ne može kao rezultat vratiti string (ili niz), u vidu izlaznog argumenta, već se to radi simuliranjem poziva po referenci. Dakle, stringu pristupamo preko pokazivača i mijenjamo ga unutar same funkcije.

Napomenimo da je moguće, ukoliko je dostupna implementacija, program uraditi korišćenjem funkcije `itoa(N, s, 2)`, gdje je treći argument brojna osnova 2. Za ovo je neophodno učitavanje biblioteke `<stdlib.h>`. Međutim, neke implementacije kompajlera **ne podržavaju** upotrebu ove funkcije. U nastavku je rješenje zadatka korišćenjem funkcije `itoa`:

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

void binar(int, char *);

int main()
{
    int N;
    char s[64];
    puts("Unesite cio broj N:");
    scanf("%d", &N);

    binar(22, s);
    puts(s);
}

void binar(int N, char *s)
{
    itoa(N, s, 2);
}
```

4. Napisati program koji sadrži funkciju `saberi_brojeve`, kojoj se prosljeđuje string `s`. Funkcija sabira sve prirodne brojeve iz stringa. Broj čini uzastopni niz cifara. Glavni program traži od korisnika da unese string, a zatim poziva funkciju i štampa odgovarajući rezultat.

Primjer: ako korisnik unese string

```
"dobar23dan3ucimo55programiranje"
```

dobijena suma je 81.

Prvo ćemo prezentovati manje elegantno rješenje zadatka.

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

int saberi_brojeve(char *);

int main()
{
    int N;
    char s[64];
    puts("Unesite string");
    fgets(s, sizeof(s), stdin);

    printf("%d", saberi_brojeve(s));
}

int saberi_brojeve(char *s)
{
    int suma = 0, i;
    char s1[64];
    i = 0;
    while(s[i] != '\0')
    {
        s1[0] = '\0'; // ili strcpy(s1, ""); prazan string
        while (s[i] >= '0' && s[i] <= '9' && s[i] != '\0')
        {// dok god imamo uzastopnih cifara...
            char pom[2] = {s[i],'\0'}; // pretvaramo karakter u string!!!
            strcat(s1, pom); // nadovezivanje stringova
            i++;
        }
        i++;
        suma += atoi(s1);
        // atoi pretvara string u cio broj (npr. "99" u 99)
    }
    return suma;
}
```

Dalje, razmislimo kako zadatak možemo riješiti sa **samo jednom petljom**. Mnogo **elegantnije rješenje** je dato u nastavku. Šta radi varijabla `ind`? U priloženom rješenju, izbjegnuta je i konverzija karaktera u string, prostim uvođenjem brojačke promjenljive `k` za podstring sastavljen od cifara, koji pretvaramo u broj. U našem primjeru, ovaj podstring je prvo `23`, pa `3`, pa `55`. Uočiti resetovanje brojača kada konvertujemo podstring u broj.

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

int saberi_brojeve(char *);

int main()
{
    int N;
    char s[64];
    puts("Unesite string");
    fgets(s, sizeof(s), stdin);

```

```
13 |     printf("%d", saberi_brojeve(s));
14 | }
15 |
16 | int saberi_brojeve(char *s)
17 | {
18 |     int suma = 0, i, ind = 0, k = 0;
19 |     char s1[64];
20 |     i = 0;
21 |     while(s[i] != '\0')
22 |     {
23 |         if (s[i] >= '0' && s[i] <= '9')
24 |         {
25 |             s1[k++] = s[i];
26 |             i++;
27 |         }
28 |         else
29 |         {
30 |             i++;
31 |             ind = 1;
32 |         }
33 |         if (ind == 1)
34 |         {
35 |             s1[k] = '\0'; // postring sastavljen od cifara sada završavamo
    term. kar.
36 |             suma += atoi(s1);
37 |             s1[0] = '\0';
38 |             ind = 0;
39 |             k = 0;
40 |         }
41 |     }
42 |     return suma;
43 | }
```