Osnovi programiranja – C

Stringovi. Strukture. Fajlovi.

Teme

- Stringovi
- Funkcije sa stringovima
- Stringovi i pokazivači
- Strukture
- Deklaracija i inicijalizacija strukture
- Operatori i funkcije za rad sa strukturama
- Fajlovi
- Funkcije za rad sa fajlovima

Stringovi

Stringovi

- Najčešća upotreba 1D nizova je za kreiranje znakovnih nizova (stringova)
- C podržava dve vrste znakovnih nizova:
  - niz znakova koji se završava nulom (znakom '\0') (engl. null-terminated string), kao u jeziku C
  - tip string definisan u biblioteci klasa
- C kompajler automatski dodaje nulu na kraju znakovnog niza

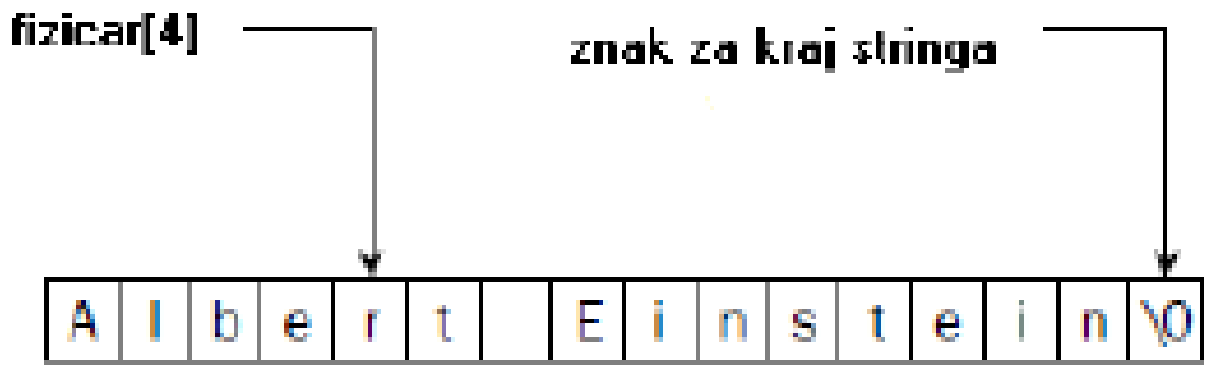
Stringovi (2)

Primer:

char fizicar [16] = {‘A’, ‘l’ ,’b’,’e’,’r’,’t’,’ ‘,’E’,’i’, ‘n’, ‘s’, ‘t’, ‘e’, ‘i’, ‘n’, ‘\0’};

char fizicar [16] =“Albert Einstein”

char fizicar [] =“Albert Einstein”



Definisanje stringova

- Niz se mora deklarisati tako da mu dimenzija bude za jedan veća od dužine najdužeg stringa koji će sadržati
- C automatski dodaje nulu na kraj znakovnih konstanti
- “ “ je prazan string (null string) i sadrži samo nulu
- Generalno, ime znakovnog niza koji sadrži string može se koristiti svuda gde se prihvata string konstanta

Stringovi: Primer

```
#include <stdio.h>
#include<string.h>
int main(){
   char fizicar [16]={'A', 'l' ,'b','e','r','t',' ','E','i', 'n', 's', 't', 'e', 'i', 'n', '\0'};
  printf("%s\n", fizicar);
  char fizicar1 [16] ="Albert Einstein";
  printf("%s\n", fizicar1);
  char fizicar2 [] ="Albert Einstein";
  printf("%s\n", fizicar2);
  return 0;
}
```

```
Albert Einstein
Albert Einstein
Albert Einstein
```

Učitavanje stringa sa tastature

- Najlakši način za unošenje stringa sa tastature jeste korišćenje funkcije scanf i znakovnog niza
- Funkcije scanf i printf prihvataju i samo ime niza, bez indeksiranja
- Funkcija scanf međutim prestaje da učitava string čim naiđe na razmak, tabulator ili nov red (%s označava unos samo 1 reči)

Učitavanje stringa sa tastature (2)

- Problem sa učitavanjem razmaka rešava se korišćenjem funkcije gets(ime_niza) koja je definisana u zaglavlju <string.h>
- Funkcija gets učitava znakove sa tastature sve dok korisnik ne pritisne taster <enter> na tastaturi.

Učitavanje stringa sa tastature (4)

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

int main(){
   char str[80];
   printf("Unesite string: ");
   gets(str);
   printf("Uneli ste %s\n", str);
   return 0;
}
```

```
Unesite string: Divan dan!
Uneli ste Divan dan!
```

Funkcije string biblioteke

- Ove funkcije date su u datotekama <stdio.h>, <string.h> i <ctype.h>

strcpy(char *s1, char *s2)

- Kopira string s2 u string s1. Niz s1 mora da bude dovoljno dugačak, jer će se u suprotnom desiti prekoračenje (engl. overrun) i program će pasti.

strcmp(char *s1, char *s2)

- Poredi stringove s1 i s2; ako su jednaki, vraća 0, ako je s1 > s2 (po leksikografskom redu) vraća 1, u suprotnom vraća negativan broj

strlen(s)

- Vraća dužinu stringa s

strcat(char *s1, char *s2)

- Spaja stringove s1 i s2

strcpy(s1, s2)

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

int main() {
    char s1[30] = "prvi string";
    char s2[30] = "drugi string";
    strcpy(s1,s2);
    printf("s1 = %s\n", s1);
    printf("s2 = %s\n", s2);
    return 0;
}
```

```
s1 = drugi string
s2 = drugi string
```

strncmp (s1, s2)

(const char *str1, const char *str2, size_t n)

```
#include <stdio.h>
#include <string.h>
int main() {
   char s1[20] = "123456789";   char s2[20] = "12345";
   if (strncmp(s1, s2, 8) ==0)
        printf("string 1 i 2 su isti";
    else
        printf("string 1 i string 2 nisu isti");
   return 0;
}
```

strlen (s)

strlen(const char *str)

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

int main(){
    char str1[20] = "Kakav divan dan!";
    printf("Duzina stringa str1 = %s\n", strlen(str1));
    return 0;
}
```

Funkcije za rad sa znakovima

- toupper(c): prevodi string u velika slova (uppercase)
- tolower(c): prevodi string u mala slova (lowercase)
- isupper(c): true ako su sva slova u stringu velika
- islower(c): true ako su sva slova u stringu mala
- isalpha(c): true ako je znak slovo (veliko ili malo)
- isdigit(c): true ako je c broj
- isalnum(c): true ako je znak slovo ili broj
- isspace(c): true ako je whitespace (razmak, nov red, carriage return, vertical tab)

Funkcije za rad sa znakovima (2)

```
#include <stdio.h>
#include <string.h>
#include <ctype.h> //funkcija toupper
int main() {
   int i;
   char str[80];
   strcpy (str, "Funkcija za rad sa znakovima");
   for (int i=0; str[i];i++)
      str[i]=toupper (str[i]);
   printf("%s", str);
   return 0;
}
```

FUNKCIJA ZA RAD SA ZNAKOVIMA

Nizovi stringova

- Niz stringova je specijalan oblik 2D niza

```
//inicijalizacija niza od 7 stringova sa po 11 znakova
char daniUNedelji [7][11] = { “Ponedeljak”, “Utorak”, “Sreda“, “ Četvrtak“, “Petak”, “Subota”, “ Nedelja” };
```

- Pojedinačnim stringovima se pristupa navođenjem levog indeksa

```
daniUNedelji [3]  //četvrtak
```

- Pojedinačnim znakovima pristupa se navođenjem oba indeksa

```
daniUNedelji [3][2] //slovo t u stringu čet vrtak
```

Nizovi stringova (2)

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

int main() {
   char daniUNedelji [7][11] = { "Ponedeljak","Utorak", "Sreda",
                                 "Cetvrtak", "Petak", "Subota"," Nedelja"};
   printf("daniUNedelji [3]= %s\n", daniUNedelji [3]);
   printf("daniUNedelji [3][2]= %c\n", daniUNedelji [3][2]);
   return 0;
}
```

```
daniUNedelji [3]= Cetvrtak
daniUNedelji [3][2]= t
```

Pokazivači i stringovi

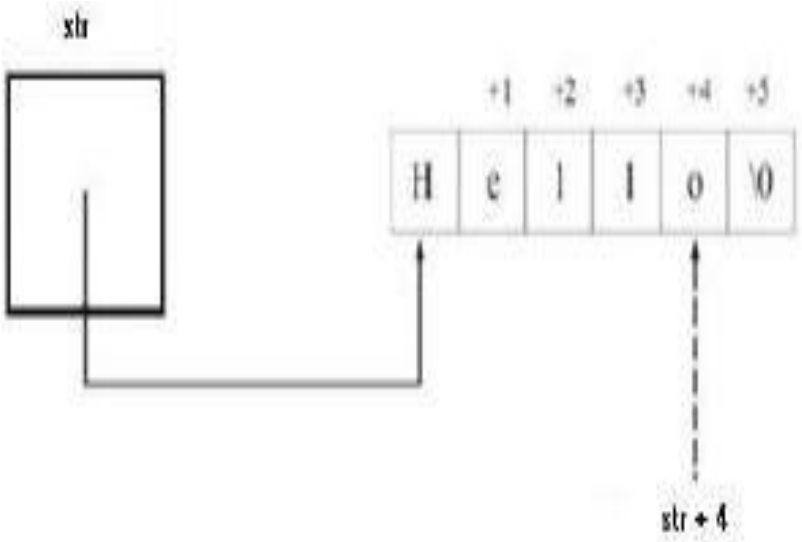
- Pokazivač na char ima zanimljivu osobinu da se može inicijalizovati string literalom (konstantom)
- Kada prevodilac naiđe na string literal, smešta ga u tabelu stringova programa i generiše pokazivač na taj string
- Prevodilac tretira string konstantu kao pokazivač na prvi znak stringa

Pokazivači i stringovi

- Pokazivač na char ima zanimljivu osobinu da se može inicijalizovati string literalom (konstantom)

```
char *str = "Hello";
```

str[2] je isto sto i *(str + 2)



Strukture

Deklaracija strukture

- Strukture vrše grupisanje više podataka različitog tipa.
- Deklaracija strukture:

```
struct ime{
        tip1 ime1;
        tip2 ime2;
        .................
        tipn imen;
 };
```

Deklaracija strukture

- Strukture se koriste kada se žele opisati složeni pojmovi npr. osoba za koju se navodi dosta podataka poput imena, prezimena, godišta, telefona i dr.
- Ovi podaci mogu biti ugrađenih tipova ili nove strukture sa svojim podacima.
- Deklaracija strukture samo definiše tip podatka ali ne rezerviše memorijski prostor.
- Nakon što je struktura deklarisana možemo definisati promenljive tog tipa:

  struct ime varijabla1, varijabla2, ...;

Deklaracija strukture – primer 1

```
struct Osoba {
        char ime[20];
        char prezime[20];
        int godiste;
        char telefon[10];
};

struct Osoba petar, janko, marko;
```

Deklaracija strukture – primer 2

```
typedef struct {
        char ime[20];
        char prezime[20];
        int godiste;
        char telefon[10];
} Osoba;

Osoba petar, janko, marko;
```

Deklaracija strukture – primeri 3

```
struct Osoba {
    char ime[20];
    char prezime[20];
    int godiste;
    char telefon[10];
} petar, janko, marko;
```

```
struct {
    char ime[20];
    char prezime[20];
    int godiste;
    char telefon[10];
} petar, janko, marko;
```

Inicijalizacija strukture

- Promenljiva tipa strukture se može inicijalizovati prilikom definicije

  struct ime varijabla={v1, v2, ..., vn};

- Pri tome se v1 pridružuje prvom članu strukture, v2 drugom itd.

Inicijalizacija strukture – primer 1

Ako imamo strukturu:

```
struct racun {
      int broj_racuna;
      char ime[80];
      float stanje;
};
```

Definišemo varijablu kupac kao:

```
struct racun kupac={1234, "Petar S.", 234.00}
```

Inicijalizacija polja struktura – primer

Za prethodnu strukturu definišemo varijablu kupci kao:

```
struct racun kupci[]={34, "Petar S.", 234.00,
                      35, "Marko T.", 456.00,
                      36, "Predrag R.", 4.00}
```

Operator tačka

- Koristi se za pristup pojedinačnim podacima i ima najviši prioritet.

```
struct Osoba {
      char ime[20];
      char prezime[20];
      int godiste;
      char telefon[10];
};
struct Osoba petar, janko, marko;
petar.ime;
janko.godiste;
marko.telefon;
```

Operator tačka (2)

```
struct racun {
        int broj_racuna;
        char ime[80];
        float stanje;
};
```

```
struct racun kupac={1234, "Petar S.", 234.00}
kupac.ime[6];    → slovo S
```

Operator tačka (3)

```
struct racun {
        int broj_racuna;
        char ime[80];
        float stanje;
};
struct racun kupci[]={34, "Petar S.", 234.00,
                      35, "Marko T.", 456.00,
                      36, "Predrag R.", 4.00}
```

kupci[2].broj_racuna; → broj 36

Operacije nad strukturama

- Jedine operacije koje se mogu sprovoditi na strukturi kao celini su pridruživanje (=) i uzimanje adrese pomoću adresnog operatora (&)
- Struktura može biti argument funkcije
- Funkcija može vratiti strukturu u pozivni program

Struktura i funkcije - primer

- Sledeća funkcija uzima 2 strukture tipa *tačka* kao argumente i vraća strukturu tipa *tačka* koja je suma 2 argumenta:

```
struct tacka suma(struct tacka p1, struct tacka p2)
{
        p1.x += p2.x;
        p1.y += p2,y;
        return p1;
}
```

Strukture i pokazivači

- Pokazivač na strukturu se definiše isto kao i pokazivač na osnovne tipove varijabli.

```
struct tacka {
        int x;
        int y;
} p1, *pp1;
pp1=&p1;
(*pp1).x;
(*pp1).y;
```

Operator strelica (→)

- C nudi jenostavniji način dohvatanja člana strukture putem pokazivača.
- Npr. ako je *ptvar* pokazivač na strukturu a *clan* jedan član strukture:

ptvar → clan

je isto što i

(*ptvar).clan

Operator strelica (→)

- Ako je član strukture i sam struktura onda možemo kombinovati operatore → i . da bismo došli do podčlana uložene strukture.

ptvar → clan.podclan

- Ako je neki član strukture niz onda elemente niza dostižemo izrazom

ptvar → clan[izraz]

Fajlovi

Fajlovi

- Fajlovi omogućavaju da trajno sačuvamo vrednosti tako da kasnije možemo nastaviti rad sa tim istim podacima.
- Npr. adresar ne bi bio praktičan ako bismo pri svakom pokretanju sve podatke ponovo unosili.

Funkcije za rad sa fajlovima

- Fajl otvaramo funkcijom *fopen* koja vraća pokazivač na strukturu *file*.
- Sve ostale funkcije za rad sa fajlovima koriste taj pokazivač umesto imena fajla.
- Sve funkcije osim *fopen* vraćaju *int* vrednost.
- Mnoge funkcije za rad sa fajlovima vraćaju konstantu EOF (End Of File) ukoliko nastane greška pri izvršavanju ili ako se dođe do kraja fajla.

Funkcije za rad sa fajlovima (1)

fopen(ime, rezim)	Otvaranje fajla datog imena. Režim w-pisanje, r-čitanje, a-dodavanje
fclose(f)	Zatvaranje fajla (rezultat 0=uspešno, EOF=neuspešno)
feof(f)	Ispitivanje da li je došlo do kraja fajla (0=nije kraj, >0 kraj)
fseek(f, kol, odakle)	Pozicioniranje. Odakle: SEEK_SET od početka, SEEK_CUR od trenutne pozicije, SEEK_END od kraja fajla.
ftell(f)	Trenutna pozicija u fajlu (vraća int vrednost ili -1 za grešku)

Funkcije za rad sa fajlovima (2)

ferror(f)	Ispitivanje greške u prethodnoj operaciji (0=nema greške)
getc(f) ili fgetc(f)	Čitanje znaka iz fajla (vraća int ili EOF u slučaju greške)
putc(znak, f) ili fputc(znak,f)	Upisivanje znaka u fajl (isti karakter ili EOF u slučaju greške)
fgetc(f, kol, string)	Čitanje stringa iz fajla. NULL ako nema više podataka
fputc(string, f)	Upisivanje stringa u fajl (vraća >0 ili EOF u slučaju greške)

Funkcije za rad sa fajlovima (3)

fflush(f)	Pražnjenje bafera (0=uspešno, EOF neuspešno)
fscanf(f, maska, prom)	Čitanje promenljivih iz fajla. Vraća broj pročitanih znakova, negativnu vrenost za grešku ili EOF ako nema podataka
fprintf(f, maska, prom)	Upisivanje promenljivih u fajl. Vraća broj upisanih znakova ili negativnu vrednost u slučaju greške
fscanf(stdin, "Zdravo!");	scanf("Zdravo!");
fprintf(stdout, "Zdravo!");	printf("Zdravo!");

Zaključak (1)

- Stringovi – znakovni nizovi koji se završavaju nulom (\0)
- Funkcije string biblioteke: **strcpy(s1, s2), strcmp(s1, s2), strlen(s)**
- Strukture vrše grupisanje više podataka različitog tipa i koriste se kada se žele opisati složeni pojmovi npr. osoba za koju se navodi dosta podataka poput imena, prezimena, godišta, telefona i dr.
- Nakon što je struktura deklarisana možemo definisati promenljive tog tipa:

struct ime varijabla1, varijabla2, ...;

Zaključak (2)

- Fajlovi omogućavaju da trajno sačuvamo vrednosti tako da kasnije možemo nastaviti rad sa tim istim podacima.

Kraj prezentacije

HVALA NA PAŽNJI!