PROGRAMIRANJE

PROGRAMSKI JEZIK C

Nizovi

Nizovi

- Pored skalarnih tipova podataka C jezik podržava i složene podatke
  - kolekcije podataka koje mogu biti
    - Nizovi
    - Strukture
- Niz je kolekcija podataka istog tipa
  - Pojedinačni podatak je element niza i pristupa mu se preko imena niza i njegove pozicije u nizu
  - Najprostiji tip niza je jednodimenzioni niz

Jednodimenzionalni niz

- Elementi jednodimenzionog se smeštaju jedan do drugog
- Niz se deklariše navođenjem tipa elemenata niza i broja elemenata niza u uglastim zagradama [,]

  int a[10];
  - Tip elemenata niza je proizvoljan tip
  - Broj elemenata niza je bilo koji celobrojni konstantni izraz
  - Obično se zbog lakše izmene, dužina niza uvodi direktivom #define

    ```
    #define N 10
    …
    int a[N];
    ```
  - Pristup pojedinačnom element niza se vrši preko njegove pozicije u nizu
  - Pozicija elementa u nizu je odredjena njegovim indeksom
    - indeks prvog elementa je 0, drugog 1, … n-tog n-1
  - Konačno selekcija pojedinačnog elementa niza se vrši navođenjem imena niza za kojim sledi njegov indeks u uglastim zagradama

    a[0], a[1], a[2], …, a[9]

Jednodimenzionalni niz

- Izraz a[i] je l-vrednost odnosno može biti levi operand operatora dodeljivanja

```
a[0]=1;
++a[i];
```

- Nizovi i for petlje obično idu zajedno

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

```
sum=0;
for(i=0; i<N; i++) sum+=a[i];
```

- Korektnost indeksiranja se ne proverava, pa je ponašanje programa u slučaju da je indeks van granica nepredvidivo

```
int a[10], I;
for(i=1; i<=10; i++) a[i]=0;
```

- Indeks niza može biti proizvoljan celobrojni izraz,

```
a[i+10*j]=0;
```

  - To može biti i izraz sa bočnim efektom

```
i=0;
while(i<N) a[i++]=0;
```

Jednodimenzionalni niz

- Niz se može inicijalizovati na sličan nacin kao i skalarna promenljiva, zajedno sa deklaracijom
  - Inicijalizator niza je lista konstantnih izraza u vitičastim zagradama {,}

```
int a[10]={1,2,3,4,5,6,7,8,9,10};
```

    - Ako je lista kraća od dužine niza ostali elementi se inicijalizuju sa 0

```
int a[10]={0}; /* svi se inicijalizuju sa 0 */
```

    - Lista ne može biti prazna, niti biti dužine veće od dužine niza
    - Ako je prisutan inicijalizator niza dužina niza se može izostaviti.

```
int a[]={1,2,3,4,5,6,7,8,9};
```

  - sizeof operator vraća dužinu niza u bajtovima

```
sizeof(a)/sizeof(a[0])
```

vrednost izraza je dužina niza a

String literali

- String literal ili samo string je niz karaktera izmedju dvostrukih navodnika
- Elementi string literala mogu biti i escape sekvence
  - Escape sekvenca koja je deo stringa može biti zadana u nekom od oblika
    - karakter reprezentacija escape sekvence
    - oktalna reprezentacija escape sekence
    - heksadekadna reprezentacija escape sekvence
  - Oktalna reprezentacija escape sekvence ima njviše 3 oktalne cifre
    - String "\1234" ima dva karaktera '\123' i '4'
    - Oktalna reprezentacija escape sekvence može imati i manje od 3 oktalne cifre i pri tome kraj escape sekvence je znak koji nije oktalna cifra
    - String "\189" sadrži tri karaktera '\1', '8', '9' jer 8 nije oktalna cifra
  - Heksadekadna reprezentacija escape sekvence nema ograničenje po pitanju broja heksadekadnih cifara
    - Kraj reprezentacije je prvi znak koji nije heksadekadna cifra
    - String "Z\xfcrich" sadrži sledeće karaktere 'Z', '\xfc', 'r', 'i', 'c', 'h'
    - String "\xfcber" sadrži dva karaktera '\xfcbe', 'r'

String literali

- String se nastavlja u sledećem redu ako se u tekući red završava znakom \
  - karakter \ spaja tekuću i sledeću liniju programskog teksta

```
printf("Hello, \
        World!\n");
```



```
printf("Hello,        World!\n");
```

- Ako su dva (ili više) stringa uzastopna (razdvojeni samo belinama) spajaju se u jedan string

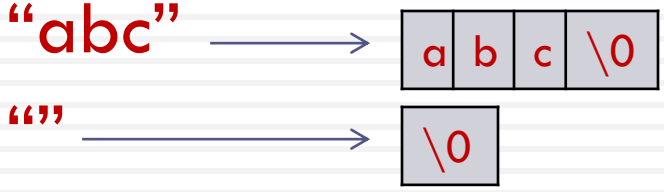
```
printf("Hello, "
       "World!\n");
```



```
printf("Hello, World!\n");
```

- String se smešta u memoriju racunara kao niz karaktera na sledeći nacin
  - String dužine n karaktera se smešta u n+1 bajtova
  - U prvih n bajtova se smeštaju redom karakteri stringa
  - U poslednji bajt se smešta takozvani null-karakter koji je indikator kraja niza karaktera
  - Null-karakter je karakter čiji je kod baš vrednost 0 i njegova oktalna reprezentacija je '\0'

String literali



- String se smešta kao niz karaktera

```
printf(“abc”);
```

- Ako string ima n karaktera dozvoljeni indeksi su 0, 1, 2, …, n
  - indeks n odgovara null-karakteru
- printf i scanf funkcije kao prvi argument očekuju vrednost koja je tipa niz karaktera

```
printf(‘\n’); /* GRESKA */
```

Znakovne konstante i string literali

- String dužine 1 nije isto što i znakovna konstanta
  - String se uvek reprezentuje kao niz karaktera
  - Znakovna konstanta se reprezentuje svojom vrednošću
    - String “a” se reprezentuje kao niz od 2 karaktera

    - Znakovna konstanta ‘a’ se reprezentuje svojim kodom koji je u slucaju ASCII karakter set-a jednak 97

String promenljive

- Neki programski jezici imaju ugrađen tip podataka čije su vrednosti string literali
- U C jeziku string literali su nizovi karaktera koji se završavaju null-karakterom
  - Nije svaki niz karaktera string literal

```
char ch_array[]={'a', 'b', 'c'};
char str[]={'a', 'b', '\0', 'c'};
```

- Da bi se niz karaktera mogao koristiti kao string mora sadržati null-karakter
  - Dužina stringa nije određena dužinom niza već pozicijom null-karaktera
- Ako želimo deklarisati string promenljivu koja za vrednost može uzeti string sa najviše 80 karaktera

```
#define STR_LEN 80
char str[STR_LEN+1];
```

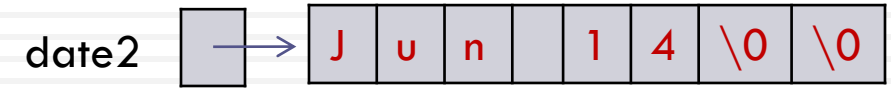
- Dužina odgovarajućeg niza karaktera je za 1 veća (null-karakter)

Inicijalizacija string promenljive

`char date1[7]="Jun 14";` → `char date1[7]={'J', 'u', 'n', ' ', '1', '4', '\0'};`

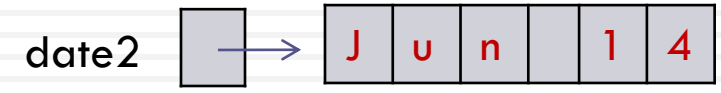
- "Jun 14" u deklaraciji sa inicijalizacijom je skraćeni zapis inicijalizatora
- Ako je dužina inicijalizatora manja od dužine niza preostali karakteri niza se inicijalizuju null-karakterom

`char date2[8]="Jun 14";`



- U slučaju da je dužina inicijalizatora veća od dužine niza vrši se odsecanje i ne dodaje se null-karakter na kraju

`char date3[6]="Jun 14";`



- Slično kao kod inicijalizacije bilo kog niza dužina se može ispustiti ako se navede inicjalizator

Izlaz string literala

- printf: Odgovarajući specifikator konverzije za ispisivanje string literala je %s

```
char str[]=”Zar nije zabavno?”;
printf(“%s\n”, str);
```

  - specifikatorom konverzije %s ispisuje se karakter po karakter sve dok se ne naidje na null-karakter
  - Ako null-karakter nije elemenat odgovarajućeg niza karaktera printf funkcija nastavlja i posle kraja niza sve dok ne naidje na null-karakter negde u memoriji
  - Specifikator konverzije %s se može zadati u obliku %[m][.p]s
    - m je minimalna širina polja
      - Ako je dužina stringa veća od minimalne širine polja, polje će se proširiti do odgovarajuće širine (nema odsecanja)
      - U slučaju da je dužina stringa manja od minimalne širine polja string će se dopuniti blanko karakterima sleva (desno poravnanje) sem ako se znakom – kao vodećim u specifikatoru konverzije, ne zahteva levo porananje u polju duzine m
    - p je maksimalni broj karaktera koji će se ispisati

Izlaz string literala

- puts: funkcija iz standardne biblioteke ispisuje string koji se zadaje kao argument

```
char str[]=”Zar nije zabavno?”;
puts(str);
```

  - Nakon ispisivanje stringa prelazi se u novi red (ispisuje se new-line karakter)

Ulaz string literala

- scanf: Odgovarajući specifikator konverzije za učitavanje string literala je %s

```
char str[81];
…
scanf("%s", str);
```

Karakter & (adresni operator) se ne ispisuje kada se radi o nizu karaktera

  - Specifikator konverzije %s na sledeći način upravlja učitavanjem string podatka
  - Preskaču se sve beline, prvi karakter koji nije belina je pocetak stringa dalje se učitavaju karakteri stringa sve do prve beline (pretražuje se niz karaktera koji ne sadrži beline)
  - scanf funkcija na kraju stringa uvek dopisuje null-karakter
- Za učitavanje kompletne linije (koja uključuje i beline) koristi se gets funkcija iz standardne biblioteke
  - gets funkcija ne preskače beline
  - gets funkcija učitava karaktere sve do kraja linije odnosno do pojave new-line karaktera koji se odbacuje i na njegovo mesto se dopisuje null-karakter (kraj stringa)

Ulaz string literala

- scanf odnosno gets funkcija ne mogu detektovati kada je niz karaktera koji prihvata string koji se učitava, popunjen
- Postoji opasnost da se upis karaktera stringa koji se učitava bude nastavljen i nakon što se prihvatni niz popuni
- U specifikatoru konverzije scanf funkcije se može zadati maksimalna širina polja odnosno maksimalni broj karaktera koji će se učitati
- Specifikator konverzije za učitavanje string podatka je %[m]s
  - m je maksimalna širina polja

```
scanf("%80s", str);
```

- gets funkcija nema ovu mogućnost

```
…
char str[81];
int ch, i=0;
while((ch=getch()) != '\n')
    if(i<80) str[i++]=ch;
    else break;
str[i]='\0';
…
```

Pristup karakterima u stringu

- Kako se string smešta kao niz, indeksiranjem se može pristupati svakom pojedinom karakteru u stringu

```
char str[81];
int count=0, i;
…
for(i=0; s[i]!='\0', i++)
    if(s[i]==' ') count++;
…
```

Ako niz karaktera s nije string (ne završava se null-karakterom) morala bi se koristiti još jedna vrednost koja bi predstavljala stvarnu dužinu niza