Nizi

Niz je vsako zaporedje znakov, ki ga napišemo v dvojnih narekovajih. Do sedaj smo nize uporabljali v funkcijah `printf` in `scanf`, sedaj pa si bomo pogledali, kako jih shranimo v spremenljivko in delamo z njimi.

Niz ni nič drugega, kot tabela znakov, katere zadnji element je vedno znak s kodo 0 (tega napišemo kot `'\0'`). Niz bi torej lahko definirali takole:

```
char niz[] = {'J', 'a', 'n', 'e', 'z', ' ', 'N', 'o', 'v', 'a', 'k', '\0'};
```

A ker je tak način precej neroden, nam C omogoča, da niz napišemo kar v dvojnih narekovajih. V tem primeru nam znaka s kodo 0 na koncu ni treba pisati, saj ga prevajalnik doda avtomatsko.

```
char niz[] = "Janez Novak";
```

Za nize torej velja vse, kar smo se naučili že pri tabelah, za lažje delo z nizi pa imamo na razpolago še knjižnico `string.h`, ki vsebuje vrsto uporabnih funkcij (če jo potrebujemo, jo moramo vključiti na začetku datoteke).

`strlen`

Funkcija `strlen` vrne dolžino danega niza v znakih. Pri tem znaka '\0' ne šteje zraven. Niz `""` (prazen niz) ima dolžino 0, zgoraj definirani niz pa 11. Zanko po vseh znakih v nizu bi torej lahko napisali takole:

```
for (int i = 0; i < strlen(niz); ++i) ...
```

Vendar je to precej neučinkovito, saj moramo dolžino niza računati znova pri vsakem preverjanju pogoja. Veliko bolje je dolžino niza izračunati že pred zanko in si dobljeno vrednost zapomniti v posebno spremenljivko:

```
int n = strlen(niz);
for (int i = 0; i < n; ++i) ...
```

Še najbolje pa je, če nam dolžine sploh ni treba izračunati. V tem primeru zanko ponavljamo tako dolgo, da pridemo do znaka za konec niza:

```
for (int i = 0; niz[i] != '\0'; ++i) ...
```

`strchr`

Funkcija `strchr` v danem nizu poišče dani znak. Če iskanega znaka ne najde, vrne vrednost `NULL`, sicer pa vrne kazalec na najdeni znak. Več o kazalcih pride na vrsto kasneje.

```
if (strchr(niz, znak) == NULL) printf("Iskanega znaka ni v nizu");
```

`strcmp`

Funkcija `strcmp` primerja dana niza po abecedi (leksikografsko). Funkcija vrne vrednost 0, če sta niza enaka, negativno vrednost, če je prvi niz manjši od drugega (prvi leksikografsko pred drugim) in pozitivno vrednost, če je prvi niz večji od drugega (prvi leksikografsko za drugim).

```
int c = strcmp(niz1, niz2);
if (c < 0) printf("Prvi niz je manjsi od drugega");
else if (c > 0) printf("Prvi niz je vecji od drugega");
else printf("Niza sta enaka");
```

Vnos in izpis nizov

Za vnos in izpis nizov imamo na razpolago formatno določilo `%s`. Pri izpisu ni nobenih posebnosti, pri branju pa moramo upoštevati, da bo program prebral samo prvo besedo v

vrstici in ne cele vpisane vrstice. Paziti je potrebno tudi, da pred imenom spremenljivke v funkciji `scanf` ne dodamo znaka `&`, kot smo bili navajeni pri številskih spremenljivkah.

```
scanf("%s", niz);
printf("%s", niz);
```

Za branje cele vrstice, vključno z vsemi presledki, imamo na razpolago funkcijo `gets`. Prostor za prebrani niz si moramo prej rezervirati, pri vnosu niza pa paziti, da ne bomo vnesli daljšega niza, kot ga lahko shranimo v pomnilnik.

```
char niz[MAX];
printf("Vnesi niz: ");
gets(niz);
```