



Univerza v Ljubljani

Fakulteta *za računalništvo
in informatiko*

Programski jezik C

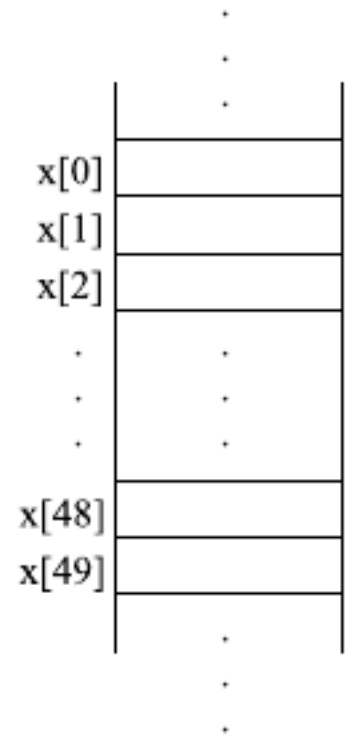
Tomaž Dobravec

Branje podatkov iz tipkovnice

- `getchar()` ... prebere en znak
 - problem: šele tipka `Enter` sproži branje tipkovnice
 - rešitev za okolje Linux: `/bin/stty raw`
- `scanf(format, spremenljivka)` ... prebere vrednost v spremenljivko
 - `scanf("%d", &i)` ... branje celega števila
 - `scanf("%s", niz)` ... branje ene besede
- `gets()` ... prebere celotno vrstico (brez znaka `'\n'`)
 - pozor: lahko zmanjka prostora v spremenljivki
 - namesto tega raje uporabi `fgets()`

Tabela

```
float x[50];
```



- tabela ima fiksno, vnaprej določeno velikost

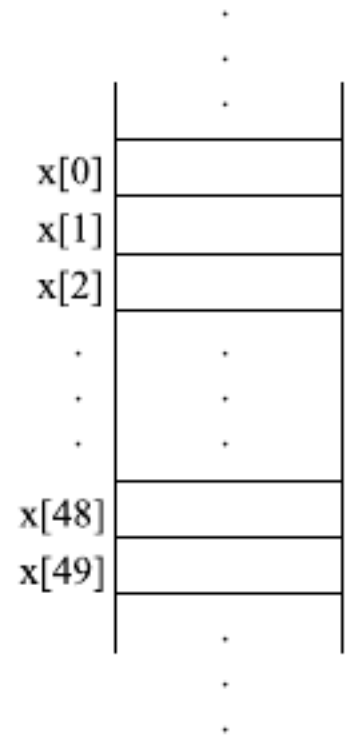
- pri deklaraciji tabele povemo njeno velikost:

```
int x[50] ... tabela za 50 celih števil
```

- tabele NE moremo deklarirati z `int x[];`

Tabela

```
float x[50];
```



- dolžina (velikost) tabele: `sizeof(tabela)`

- število možnih elementov:

```
(int) sizeof(tabela) / sizeof(tabela[0])
```

- če želimo vedeti, koliko elementov imamo v tabeli, potrebujemo števec (n)! (kot `argc` za `args`)
- elementi tabele: `tabela[0], ..., tabela[n-1]`

Tabelarične konstante

Tabelarične konstante podamo v zavutih oklepajih

```
short meseci [] =  
    {31,28,31,30,31,30,31,31,30,31,30,31};
```

```
int d1[]    = {10,20,30,40};  
int d2[10]  = {0};           // inicializacija cele tabele  
int d3[3]   = {1,2,3,4};     // napaka
```

Znakovni nizi v C

- jezik C ne pozna posebnega podatkovnega tipa za delo z znakovnimi nizi; znakovni niz v njem je preprosto tabela znakov,
- niz v C se konča z znakom `'\0'`,
- niz `"abc"` v jeziku C pomeni tabelo s štirimi znaki: `a`, `b`, `c` ter `'\0'`,
- spremenljivko, v kateri lahko hranimo nize dolžine $< N$ (N je neka konkretna številka), deklariramo z

```
| char niz[N];
```

Znakovni nizi v C

- Ob deklaraciji lahko niz tudi inicializiramo

```
char niz[6]="TEST";
```

T	E	S	T	\0	
0	1	2	3	4	5

- Kaj izpiše spodnji program?

```
char niz1[10]="ABC";  
char niz2[]  ="DEF";  
printf("%d, %d\n", sizeof(niz1), sizeof(niz2));
```

Opozorila za delo z nizi

1. Dolžine niza ne dobimo z uporabo operatorja `sizeof` ampak s funkcijo `strlen`

2. Nizov med seboj ne primerjamo z operatorjem `==` ampak s funkcijo `strcmp`

`if (niz1 == niz2) ...` ... NAROBE

`if (strcmp(niz1,niz2)==0) ...` ... PRAVILNO!

3. Za prirejanje nizov ne uporabljamo operatorja `=` ampak funkcijo `strcpy`

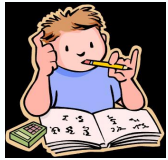
`niz1 = niz2;` ... NAROBE

`strcpy(niz1,niz2);` ... PRAVILNO!

Funkcije za delo z nizi

- ◆ `strlen` dolžina niza
- ◆ `strcmp` primerja dva niza
- ◆ `strcpy` skopira prvi niz v drugi niz
- ◆ `atoi` niz pretvori v številko
- ◆ `strcat` združi dva niza
- ◆ `strchr` poišče pojavitev znaka v nizu
- ◆ `strstr` poišče pojavitev niza v nizu

Delo z nizi



Naloga 5-VIII.

(nizi/najdaljsa.c)

Napiši program, ki s standardnega vhoda bere vrstice, dokler uporabnik ne vpiše prazne vrstice. Med branjem si program zapomni najdaljšo prebrano vrstico in jo na koncu izpiše.

A screenshot of a PuTTY terminal window. The title bar reads 'kepa.fri.uni-lj.si - PuTTY'. The terminal shows the following sequence of events: the user runs './najdaljsa', the program reads three lines of input ('abc', 'dolga vrstica', 'def'), and then outputs 'Najdaljsa: dolga vrstica'. The prompt '[user@localhost]#' is visible at the end of the output line, followed by a green cursor.

```
kepa.fri.uni-lj.si - PuTTY
[user@localhost]# ./najdaljsa
abc
dolga vrstica
def

Najdaljsa: dolga vrstica
[user@localhost]#
```