

Základy programovania

Cvičenie 3

Dátové typy, argumenty programu

Cvičiaci: Ing. Magdaléna Ondrušková, (iondruskova)



1. října 2024

Integer

```
1 #include <stdio.h>
2 int main() {
3     int num;
4     scanf("%d", &num); // nacitanie celeho cisla
5     printf("Zadali ste cislo: %d\n", num); // vypis cel ho
6     return 0;
7 }
```

Float

- float - presnosť cca 8 číslic

```
1 include <stdio.h>
2 int main() {
3     float num;
4     scanf("%f", &num); // nacitanie desatinneho cisla (float)
5     printf("Zadali ste cislo: %f\n", num); // vypis desatinneho
6     return 0;
7 }
```

Double

- double - presnosť cca 16 číslíc

```
1 #include <stdio.h>
2 int main() {
3     double num;
4     scanf("%lf", &num); // nacitanie desatinneho cisla (double)
5     printf("Zadali ste cislo: %lf\n", num); // vypis
6     // desatinneho cisla s vyssou presnostou
7     return 0;
8 }
```

Char

```
1 #include <stdio.h>
2 int main() {
3     char ch;
4     scanf("%c", &ch); // nacitanie jedneho znaku
5     printf("Zadali ste znak: %c\n", ch); // vypis jedneho znaku
6     return 0;
7 }
```

String

- Postupnosť znakov ukončená špeciálnym znakom `\0`

```
1 #include <stdio.h>
2 int main() {
3     char str[6] = "Hello"; // pole obsahuje 5 znakov + '\0'
4     char str[] = "Hello, world!"; // Ako veľké bude toto pole?
5     return 0;
6 }
```

Načítanie a vypísanie reťazcov:

```
1 #include <stdio.h>
2 int main() {
3     char str[101]; // pole pre 100 znakov + 1 pre '\0'
4     printf("Zadajte reťazec (max 100 znakov): ");
5     scanf("%100s", str); // načítanie reťazca, obmedzené na 100
6     printf("Zadali ste reťazec: %s\n", str); // vypis reťazca
7
8     return 0;
9 }
```

Dátový typ	PRINTF formát	SCANF formát
int	%i alebo %d	%i alebo %d
float	%f	%f
double	%lf	%lf
char	%c	%c
char()	%s	%s

Tabuľka: Prehľad základných typov v jazyku C

- Obsahuje funkcie pre prácu s reťazcami

Zistenie dĺžky reťazca:

```
1 #include <stdio.h>
2 #include <string.h> // NEZABUDNUT nacistat kniznicu
3
4 int main() {
5     char str[] = "Hello, world!"; // Zistenie dlzky retazca
6     int len = strlen(str);
7     int len2 = sizeof(str);
8
9     printf("Dlzka retazca: %i\n", len); // Vypise: 13
10    printf("Dlzka retazca: %i\n", len2); // Vypise: 14
11
12    return 0;
13 }
```

Kopírovanie reťazcov

- Dva parameter: kam a odkiaľ

```
1 #include <stdio.h>
2 #include <string.h>
3
4 int main() {
5     char src[6] = "Hello";
6     char dest[50]; // Musi byt dostatočne veľke
7
8     // Kopírovanie src do dest
9     strcpy(dest, src);
10
11     printf("Dest: %s\n", dest); // Vypise: Hello
12
13     return 0;
14 }
```


Porovnanie reťazcov

- Nie je možné porovnávať pomocou ==, porovnávalo by to adresy reťazcov
- Návratové hodnoty:
 - Vráti 0 ak sa reťazce zhodujú
 - Vráti kladné číslo ak je prvý reťazec (prvý nezhodujúci sa znak v ASCII) väčší
 - Vráti záporné číslo ak je druhý reťazec (prvý nezhodujúci sa znak v ASCII) väčší

```
1 // Nacitanie kniznic
2 int main() {
3     char str1[50] = "Ahoj";
4     char str2[50] = "Ahoj";
5     int result = strcmp(str1, str2); // Porovnanie str1 a str2
6     if (result == 0) {
7         printf("Retazce su rovnake\n");
8     } else if (result > 0) {
9         printf("Prvy retazec je vacsi\n");
10    } else {
11        printf("Druhy retazec je vacsi\n");
12    }
13    return 0;
14 }
```

Úloha 1: Načítajte dva refazce (každý max 100 znakov) a vypíšte refazec, ktorý refazec je dlhší.

- Rozšírenie: Vypíšte, ktorý refazec má viac malých písmen.

Úloha 2: Načítajte refazec (max 20 znakov), a následne vypíšte tento refazec odzadu

- Vstup: *Dom*
- Výstup: *moD*

Úloha 2: Načítajte refazec (max 20 znakov), a následne vypíšte tento refazec odzadu

- Vstup: *Dom*
- Výstup: *moD*

Úloha 3: Upravte kód, aby overil, či daný refazec je palindróm

- kajak
- krk
- oko

Program môžeme spúšťať s nejakými argumentami, ktoré píšeme za meno programu, oddelené medzerami

- `./hello argument1 argument2 argument3 ...`

Argumenty predávame do programu ako parametre funkcie `main`

- `argc` - celé číslo, udáva počet argumentov
- `argv[]` - pole textových reťazcov obsahujúce jednotlivé argumenty

```
1 int main(int argc, char *argv[]) {  
2     // Telo programu  
3 }
```

Program má vždy aspoň 1 argument:

- `argv[0]` - názov programu
- `argv[1]`, `argv[2]`, ..., `argv[argc-1]` - ďalšie argumenty zadané v príkazovom riadku

Výpis argumentov programu:

```
1 #include <stdio.h>
2 int main(int argc, char *argv[]) {
3     // argc obsahuje počet argumentov
4     printf("Pocet argumentov: %d\n", argc);
5     // argv je pole retazcov (argumentov)
6     for (int i = 0; i < argc; i++) {
7         printf("Argument %d: %s\n", i, argv[i]);
8     }
9     return 0;
10 }
```

Ak program spustíme s argumentami:

```
./program Hello World
```

Výpis programu by bol:

```
1 Pocet argumentov: 3
2 Argument 0: ./program
3 Argument 1: Hello
4 Argument 2: World
```

Pozor!

- Argumenty != Vstup
- Vstupy sú načítavané za behu programu
- Argumenty sa načítajú raz, pri spustení (vo funkcii main)

Argumenty po načítaní sú vždy reťazce.

- `atoi` - funkcia, ktorá prevedie reťazec na celé číslo
- `atof` - funkcia, ktorá prevedie reťazec na desatinné číslo

Použitie:

```
1 #include <stdlib.h> // Pre funkciu atoi
2
3 ...
4
5 int num1 = atoi(argv[1]);
```

Úloha A: Spustíte program s dvoma argumentami. Program vypíše súčet tých čísiel.

Úloha B: Spustíte program s dvoma argumentami. Program vypíše, ktoré číslo je väčšie (Niečo ako *Väčší argument má hodnotu: ...*).

Úloha: Vytvorte program, ktorý bude spúšťaný s práve jedným argumentom. V programe načítajte užívateľský vstup o dĺžke max 20 znakov. Výstup programu vypíšte.

- **Varianta A:**

- Ak je zadaný argument `count` vypíšte počet samohlások v reťazci
- Ak je zadaný argument `toupper` transformujte reťazec do upper case
 - Môžete použiť funkciu `toupper()`
- Ak je zadaný argument `replace`, nahradíte znaky `a/A` v reťazci znakom `-`

- **Varianta B:**

- Ak je zadaný argument `count` vypíšte počet znakov ktoré nie sú písmeno
- Ak je zadaný argument `tolower` transformujte reťazec do lower case
 - Môžete použiť funkciu `tolower()`
- Ak je zadaný argument `find`, nájdete v reťazci všetky znaky `a/A` a postupne vypíšte na ktorých indexoch (pozíciách) sa dané znaky nachádzali.