Typy proměnných a práce s konzolí

7. října 2013

Abychom mohli zkoušet, co náš program dělá, musíme začít u toho, jak s ním budeme komunikovat. K tomu potřebujeme speciální funkce, které umožňují vypisování na konzoli a čtení z konzole. Tyto funkce jsou zabalené v knihovně, která se jmenuje `stdio.h` (zkratka ze STanDard Input Output). Abychom měli k těmto funkcím přístup, musíme v úvodu našeho programu určit, že tuto knihovnu chceme používat. To provedeme následujícím příkazem:

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

Jiné specifické funkce v tuto chvíli potřebovat nebudeme a proto můžeme přejít k tělu programu. To je tvořeno tzv. hlavní funkcí a její podoba je nastíněna tady:

```
int main (void)     //toto je povinná hlavička hlavní funkce
{                   //tělo jakékoliv funkce vždy začíná levou složenou závorkou
  příkaz1;          //zde následují jednotlivé příkazy, které se mají provádět
  příkaz2;          //každý příkaz je ukončen středníkem
  příkaz3;          //složená závorka způsobí, že jsou následné příkazy odsazené doprava

  return 0;         //některé překladače vyžadují, aby program končil tímto příkazem
}                   //ukončení programu pravou složenou závorkou
```

Podrobnosti k hlavičce funkce main a k příkazu return si řekneme až na hodině věnované funkcím. V tuto chvíli si ukážeme nejjednodušší program, který jen vypíše na konzoli Ahoj! K výpisování na konzoli budeme používat funkci `printf()`, jejímž prvním parametrem je v uvozovkách uvedený text, který chceme vypsat. Program tedy bude vypadat takto:

```
#include<stdio.h>

int main (void)
{
  printf( "Ahoj!\n"); //speciální znak \n slouží k odřádkování

  return 0;
}
```

Další speciální znaky jsou:
`\t` je tabulátor
`\a` provede systémové pípnutí
`\b` smaže poslední vypsaný znak

V programu je také použit řádkový komentář, který začíná dvojitým lomítkem `//`, to znamená, že text na řádku za ním překladač nezpracuje. Pokud potřebujeme zakomentovat úsek přesahující jeden řádek, pak začátek uvedeme lomítkem a hvězdičkou `\*` a na konci je pak zopakujeme v opačném pořadí, tj. `*/`.

Typy proměnných

Pro výpočty potřebujeme vytvořit symbolické proměnné, jejichž hodnota se může v průběhu programu měnit. V jazyce C existují tři hlavní kategorie proměnných rozdělených podle toho, jaké hodnoty mohou nabývat. Jsou to celá čísla, reálná čísla a znaky. Pro celá a reálná čísla pak existují různé speciální typy, které se liší tím, kolik místa zaujímají v paměti počítače, resp. jak velkou hodnotu je do nich možné uložit a s jakou přesností. V následujícím přehledu jsou uvedeny velikosti, které platí pro můj osobní počítač. Na vašem počítači se velikosti jednotlivých proměnných mohou mírně lišit. Abyste si mohli jejich velikost vyzkoušet, stáhněte si ze stránek předmětu program `test-sizeof.c`, přeložte a spusťte.

Celá čísla:	short int	2B
	int	4B
	long int	4B
Reálná čísla:	float	4B
	double	8B
	long double	10B
Znaky:	**char**	1B

Tučně jsou vyznačené nejběžněji používané typy.

Před prvním použitím musíme každou proměnnou definovat (programu tak říkáme, kolik místa si má v paměti vymezit). Definici proměnných uvádíme vždy na začátku funkce `main`. Poté, co proměnnou definujeme, v paměti počítače se pro ni rezervuje místo, ale protože na daném místě bylo předtím pravděpodobně něco zapsáno, může tam být uložena libovolná sekvence nul a jedniček, které se po definici proměnné interpretují na příslušnou hodnotu. Z toho vyplývá, že **nově definovaná proměnná nemá nulovou hodnotu**. Proto je nutné každou proměnnou vždy včas patřičně inicializovat, to znamená, přiřadit jí patřičnou hodnotu:

```
int i,j;                //definice proměnných
long double bignumber;  //proměnné mohou mít libovolný jednoslovný název
double a=108.1;         //definice s inicializací
char znak;

i = 5;                  //inicializace
j = i+3;                //inicializace, výraz vpravo se vyhodnotí a výsledek uloží do j
bignumber = 1000.;      //u reálného čísla píšeme vždy i desetinnou tečku
znak = 'a';             //znaky píšeme vždy do apostrofu
```

Jazyk C je „case-sensitive“, tedy je rozdíl mezi proměnnou m a M, teoreticky je tedy možné mít dvě stejně se jmenující proměnné rozlišené jen velikostí písmen. Pokud k tomu ale nemáme velmi dobrý důvod, je to nevhodné, protože to působí zmatky.

Výpis proměnných na obrazovku

K vypsání hodnoty proměnné na obrazovku budeme opět používat funkci `printf()`, tentokrát bude mít ale větší počet parametrů. Vypsání proměnných `i` a `j` může vypadat např. takto:

```
printf( "Hodnoty promennych jsou i=%d a j=%d.\n", i, j );
```

kdy nám program vypíše:

```
Hodnoty promennych jsou i=5 a j=8.
```

První parametr v uvozovkách určuje formát toho, co se bude vypisovat, mohou tam být text i proměnné.

Proměnné jsou uvedeny znakem % a symbolem, který odpovídá jejich datovému typu a u reálných čísel také námi zvolenému nastavení stylu výpisu:

int	%d	printf("j=%d", j);	j=8
	%3d	printf("j=%3d", j);	j= 8
	%03d	printf("j=%03d", j);	j=008
double	%lf	printf("a=%lf", a);	a=108.100000
	%11lf	printf("a=%11lf", a);	a= 108.100000
	%11.3lf	printf("a=%11.3lf", a);	a= 108.100
	%011.3lf	printf("a=%011.3lf", a);	a=0000108.100
	%e	printf("a=%e", a);	a=1.081000e+02
	%15e	printf("a=%15e", a);	a= 1.081000e+02
	%11.3e	printf("a=%11.3e", a);	a= 1.081e+02
	%g	printf("a=%g", a);	a=108.1
	%7g	printf("a=%7g", a);	a= 108.1
char	%c	printf("znak=%c", znak);	znak=a

Za prvním parametrem následuje výčet proměnných, které se mají vypsat v pořadí, v jakém mají být vypsány.

Načítání proměnných z obrazovky

Pro načítání budeme používat funkci `scanf()`, která je také definovaná v knihovně `stdio.h`. Její použití se až na pár drobných rozdílů podobá funkci `scanf()`:

```
scanf( "%d %d", &i, &j ); //načtení proměnných i a j
```

První parametr je opět v uvozovkách a určuje jakého typu se budou hodnoty načítat. V tomto případě nepoužíváme speciální formátování, pouze procento a znak odpovídající přesnému datovému typu:

short int	%hd
int	%d
long int	%ld
float	%f
double	%lf
long double	%Lf
char	%c

Za prvním parametrem opět následuje výčet proměnných, do kterých se mají příslušné hodnoty uložit, ale pozor: **před názvem každé proměnné musí být vždy uveden znak `&`**.

Matematické operace

Jazyk C sám zahrnuje definici základních matematických operací:

sčítání	`j = i+3;`
odčítání	`j = i-3;`
násobení	`j = i*3;`
dělení	`bignumber = a/3.;`

Pokud se jedna proměnná vyskytuje na pravé i levé straně rovnítka, znamená to, že její původní hodnotu chceme změnit, např:

```
i = 3;
i = i+3;
```

Na druhém řádku se nejprve vyhodnotí výraz na pravé straně, jeho hodnota je 6 a ta se poté uloží do proměnné i. To znamená, že původní hodnotu proměnné i jsme zvedli o 3.

Základní operace, ve kterých měníme hodnotu proměnné, lze zkrátit:

```
i = i+1;   i += 1;   i++;
i = i-1;   i -= 1;   i--;
i = i-3;   i -= 3;
i = i*4;   i *= 4;
a = a/2;   a /= 2;
```

Pro případ změny čísla o jedničku existují ještě speciální operátory inkrement `++` a dekrement `--`. Např. příkaz `i++;` provede zvětšení hodnoty proměnné `i` o jedničku.

Pro další matematické funkce je nutné použít matematickou knihovnu tím, že v úvodu programu přidáme řádek:

```
#include<math.h>
```

V této knihovně jsou k dispozici např. tyto funkce:

odmocnina	`sqrt(a)`	
mocnina a^b	`pow(a,b)`	v C neexistuje operátor `^`!
exponent e^a	`exp(a)`	
přirozený logaritmus $\ln(a)$	`log(a)`	
dekadický logaritmus $\log(a)$	`log10(a)`	
goniometrické funkce	`sin(a), cos(a), tan(a), atan(a)`	úhel `a` musí být v radiánech
absolutní hodnota	`abs(i)`	pouze pro celá čísla
	`fabs(a)`	pouze pro reálná čísla
Ludolfovo číslo π	`M_PI`	pozor na velikost písmen