Programování v jazyce C

1 Základní prvky jazyka C

Program v jazyce C se sestává z klíčových slov, konstant, proměnných a identifikátorů.
Jazyk C rozlišuje malá a velká písmena.

Klíčové slovo je definovaný symbol jazyka: příkaz, pojmenování datového typu, ...

Identifikátor – pojmenování nového prvku (jméno proměnné, funkce, ...),
může se sestávat z písmen, číslic a znaku _ (nesmí začínat číslicí).

Program v jazyce C se sestává z funkcí. Je tvořen minimálně jednou funkcí ***main***.

```
int main() {
        .
        .
        .
}
```

Příkaz se ukončuje znakem ***;***
Znak *//* uvozuje poznámku (vše od tohoto místa do konce řádku se ignoruje)
Znaky ***{ }*** uzavírají tzv. blok, seskupují příkazy do jednoho složeného příkazu

2 Funkce

Funkce je samostatná část programového kódu.

```
typ_návratove_hodnoty identifikátor(parametry) {
        .
  tělo funkce
        .
}
```

Typ návratové hodnoty říká, jakého typu bude hodnota, kterou funkce vrací (viz. proměnné).
Parametry jsou hodnoty, které předáváme funkci (obvykle z nich něco počítá).

Př.:

```
int soucet(int a, int b) {
  return a+b;
}
```

3 Konstanty

Konstanta je neměnná hodnota.

Př::

13 ... konstanta celé číslo
0x24 ... konstanta celé číslo zapsaná v šestnáctkové soustavě
0b101 ... konstanta celé číslo zapsaná v dvojkové soustavě
-0.34 ... konstanta desetinné číslo
"ahoj"... konstanta typu textový řetězec

'b' ... konstanta typu znak

4 Proměnné

Proměnná je pojmenované (vyhrazené) místo v paměti. Slouží k „uschování“ nějaké hodnoty.

Před prvním použitím musí být proměnná deklarována (řečeno překladači, jak se bude jmenovat a jaký typ hodnoty bude obsahovat).

Deklarace proměnné:

```
typ identifikátor [= hodnota];
```

Základní datové typy:

int	...	celé číslo (rozsah je dán implementací jazyka a hardwarem)
float	...	číslo v plovoucí řádové čárce (desetinné číslo) v základní přesnosti
char	...	celé číslo, znak (celé číslo v rozsahu -128 až 127)

Př.:

```
int n = 5;
int a;
float x, y;
```

5 Formátovaný výpis na standardní výstup (obrazovku)

Připojení standardní knihovny pro vstup a výstup:

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

Výpis textového řetezce na obrazovku

```
printf(textový řetezec);
```

Př.:

```
printf("Ahoj");        // výpis textu Ahoj
printf("Ahoj\n");      // výpis textu Ahoj a přechod
                          na nový řádek
```

Výpis textového řetezce s vloženými hodnotami na obrazovku

```
printf(formátovací řetezec, parametr1, parametr2, ..);

// %d  ...  vložení celého čísla (int)
// %f  ...  vložení čísla v plovoucí čárce (float)
//     %8.2f     číslo formátováno na 8 znaků a
                 2 destinná místa
//     %.2f      číslo formátováno na 2 destinná místa
// %c  ...  vložení znaku (char)
// %s  ...  vložení znakového řetezce (char*)
```

Př.:

```
int a = 5;
printf("%d na druhou je %d.\n", a, a*a);
```

6 Cyklus FOR (s daným počtem opakování)

```
for (inicializace; podminka; iterator) {
        .
  tělo cyklu (to, co se opakuje)
        .
}

// inicializace  ... provede se jednou na začátku cyklu
// podminka      ... pokud je podmínka splněna, provádí
                     se tělo cyklu
// iterator      ... provádí se po každém průchodu
                     cyklu
```

Př.:

```
for (int i = 0; i < 10; i++) {
        .
        .
}
```

7 Podmíněný příkaz (větvení, podmínky)

```
if (podminka) {
  příkaz1
}


if (podminka) {
  příkaz1
} else {
  příkaz2
}

// podminka      ... podmínka větvení
// příkaz1       ... provádí se pokud je podmínka
                     splněna
// příkaz2       ... provádí se pokud není podmínka
                     splněna
```

Př.:

```
if (x < 10) {
        .
        .
} else {
        .
        .
}
```

```
>       ...  větší než
>=      ...  větší nebo rovno než
<       ...  menší než
<=      ...  menší nebo rovno než
==      ...  rovno
!=      ...  nerovno
&&      ...  logický součin (a zároveň)
||      ...  logický součet (nebo)
```

8 Cyklus s podmínkou

```
while (podminka) {
        .
  tělo cyklu (to, co se opakuje)
        .
}

// podminka       ... pokud je podmínka splněna, provádí
                       se tělo cyklu
```

Př.:

```
while (i < 10) {
        .
        .
}
```

9 Pole

10 Logické operace s celými čísly

Logické operace lze provádět nejen s logickými výrazy (podmínkami), ale i s celými čísly. Operace se provádí s odpovídajícími bity čísel.

Operátory

~	negace
&	logický součin
|	logický součet
^	výhradní logický součet
>>	posun vpravo (dělení 2)
<<	posun vlevo (dělení 2)

Př.:

```
7 & 3  // 0b0111 & 0b0011  = 0b0011 ... 3
4 | 5  // 0b0100 | 0b0101  = 0b0101 ... 5
1 << 3 // 0b0001 << 3      = 0b1000 ... 8 (*2^3)
```

Typické použití:

maskování bitů (zjištění hodnoty vybraného bitu)

`x & (1 << 3) != 0` zjistí, zda je 3. bit čísla x nenulový (nastavený)

nastavení hodnoty zvoleného bitu na 1

`x | (1 << 3)` nastaví 3. bit čísla x na jedničku

nulování hodnoty zvoleného bitu

`x & ~(1 << 3)` vynuluje 3. bit čísla x