Jazyk C

- Je to vyšší kompilovaný programovací jazyk. Kompilovaný znamená, že zdrojový kód musí být nejprve kompilován do strojového kódu, aby bylo možné soubor s instrukcemi spustit.
- Zdrojový kód, kde se nachází příkazy se dělí na dva konkrétní druhy:
  - Syntaxe = tvar příkazu + parametrů („jak se to napíše“)
  - Sémantika = význam příkazu + parametrů („co to dělá“)
- Case sensitive – rozlišuje malá a velká písmena v názvech proměnných, konstant, funkcí a hodnot. (pinMode (13, OUTPUT);)

Hlavní program a podprogramy (funkce)

Hlavní program

Hlavní program v jazyce C je část kódu, která začíná funkcí main(void) / main(). Tato funkce je vstupním bodem každého programu v C — když se spustí program, jeho běh vždy začíná právě ve funkci main(void) / main().

Začátek se zapisuje: int main(void) nebo pouze jako int main()

Konec se zapisuje: return 0;

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

int main(void) {

                              Hlavní program

  return 0;
}
```

Příklad

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

int main(void) {              // Začátek hlavního programu
  printf("Hello, World!");    // Zobrazení zprávy
  return 0;                   // Ukončení programu
}
```

Funkce

① **Deklarace funkce** – vždy nad hlavním programem!

a) Funkce bez parametrů

Nepotřebuje žádné vstupní hodnoty, aby něco udělala. Když se tato funkce zavolá, spustí se kód, který v ní je, ale nemusí se jí nic posílat. Používá se, když chceme opakovaně vykonat nějakou akci, která je vždy stejná.

```
char getOneChar () {     // V závorkách není parametr
  return 'C';            // Návratová hodnota
}
```

```
int getOneNumber () {    // v závorkách není parametr
  return 13;             // Návratová hodnota
}
```

b) Funkce s parametry – hodnotám v závorkách (A, B) se říká formální parametry!

Potřebuje vstupní hodnoty, aby mohla něco udělat. Když se tato funkce zavolá, musí se jí poslat konkrétní hodnoty, které pak použije ve svých výpočtech nebo operacích. Parametry jsou jako vstupy, které dávají funkci data potřebná k práci.

```
int addNumbers (int A, int B) {     // Hodnoty v závorkách
  return A + B;                     // Návratová hodnota
}
```

② **Volání funkce** – při volání funkce jsou uvnitř závorek skutečné parametry! (deklarované nad programem)

Volání funkce znamená spuštění kódu, který je definovaný v těle funkce. Když se funkce zavolá, program přejde na místo, kde je funkce definovaná, provede příkazy uvnitř ní a poté se vrátí zpět do místa volání. Funkce může také vrátit hodnotu.

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

int Vysledek;                       // Deklarace proměnné
int Num1 = 7;                       // Deklarace proměnné
int Num2 = -5;                      // Deklarace proměnné

int addNumbers (int A, int B) {     // Funkce s form. par.
  return A + B;                     // Návratová hodnota
```

```
}

int main(void) {                        // Začátek hl. prog.
  printf("Napiš dvě čísla: ");          // Zápis čísel
  scanf("%d %d", &Num1, &Num2);         // Čtení z klávesnice

  Vysledek = addNumbers (Num1, Num2);     // Volání funkce
                                          // Skutečné par.

  return 0;                             // Ukončení
}
```

Návratová hodnota

výsledek, který funkce vrátí, když skončí svou práci. Když se funkce zavolá, může tento výsledek použít dál v programu, například ho uložit do proměnné nebo ho zobrazit na obrazovce. Funkce vrací hodnotu pomocí příkazu return.

Globální a lokální proměnné

Globální proměnná

- jsou deklarovány mimo blok složených závorek {}
- platí v hlavním programu i ve všech funkcích (pro celý program)

Lokální proměnná

- jsou deklarovány uvnitř funkce, v bloku složených závorek {}
- platí pouze uvnitř funkce, kde jsou deklarovány

Formální parametr funkcí se uvnitř funkce chová jako lokální proměnná

! Uvnitř funkce neměnit hodnotu globální proměnných !

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

int n1, n2, RES, RES1;                  // Globální proměnné

int f_soucetCisel(int a, int b) {   // Lokální proměnné
  RES1 = (a + b);
  return RES1;
}

int main(void) {                        // Začátek programu
  printf(„Napiš dvě čísla: „);          // Zobrazení zprávy
  scanf(„%d %d", n1, n2);               // Čtení z klávesnice
```

```
  RES = f_soucetCisel(n1, n2);      // volání funkce
  printf(„Součet čísel je: %d, RES); // Výpis výsledku

  return 0;                         // Ukončení programu
}
```

Praktický příklad – Replit C-25

```c
//·Nadeklaruj·funkci·f_najdiMaximum()·datového·typu·int
//·Funkce·bude·mít·tři·formální·parametry·"a,·b,·c"·datového·typu·int
//·Funkce·vrátí·návratovou·hodnotu·maximum·z·těchto·tří·čísel
//·Napiš·hlavní·program,·ve·kterém·uživatel·zadá·postupně·tři·celá·čísla,·zavolej·tuto·funkci·a·předej·jí·tyto·skutečné·parametry
//·Funkce·vrátí·do·hlavního·programu·hodnotu·maxima,·kterou·v·hlavním·programu·vypíšeš

#include·<stdio.h>

int·cislo1,·cislo2,·cislo3,·maximum,·soucet;

int·f_najdiMaximum(int·a,·int·b,·int·c)
{
··if(a·>·maximum)·maximum·=·a;
··if(b·>·maximum)·maximum·=·b;
··if(c·>·maximum)·maximum·=·c;·
··return·maximum;
}

int·main(void)
{
··printf("Napiš·tři·čísla:·\n");
··scanf("%d·%d·%d",·&cislo1,·&cislo2,·&cislo3);

··printf("\n");

··soucet·=·f_najdiMaximum(cislo1,·cislo2,·cislo3);
··printf("Největší·číslo·je:·%d",·soucet);

··return·0;
}
```