Jazyk C - Funkce

Matěj Hanke

17. září 2025

1 Co je funkce

Funkce je samostatná část programu, která plní konkrétní úkol. Používáme je, aby byl kód přehlednější, kratší a snadněji udržovatelný. Můžeme si funkci představit jako malý podprogram, kterému dáme vstup (parametry), on něco provede a vrátí nám výsledek (návratová hodnota).

Důležité body k zapamatování

- **Deklarace (prototyp), definice a volání funkce.** Deklarace říká kompilátoru, že funkce existuje (jaký má návratový typ a parametry). Definice obsahuje samotný kód funkce. Volání je místo, kde funkci opravdu použijeme.
- **Parametry se předávají hodnotou.** To znamená, že do funkce se posílá kopie proměnné. Pokud uvnitř funkce parametr změníme, původní proměnná mimo funkci se nezmění.
- **Návratová hodnota.** Funkce může vrátit jednu hodnotu (například `int`, `double`, `char`). Pokud nechceme nic vracet, použijeme klíčové slovo `void`.
- **Lokální a globální proměnné.** Proměnné vytvořené uvnitř funkce existují jen během jejího běhu (lokální). Proměnné definované mimo všechny funkce jsou globální – jsou přístupné odkudkoliv v programu, ale snižují přehlednost, proto je používáme opatrně.
- **Modularita.** Funkce dělají program čitelnější a umožňují kód znovu použít vícekrát. Díky nim můžeme rozdělit složitý problém na menší části.

Co se stane v paměti při zavolání funkce

Když program zavolá funkci, stane se několik důležitých věcí:

1. Program si **zapamatuje, odkud funkci zavolal**, aby se tam mohl po skončení vrátit.
2. **Na zásobník (stack)** se uloží parametry funkce a také lokální proměnné. Každé volání funkce má svou vlastní kopii těchto hodnot.
3. Spustí se tělo funkce, provádějí se příkazy uvnitř.
4. Pokud funkce vrací hodnotu, ta se pošle zpět na místo, kde byla funkce zavolána.
5. Po ukončení funkce se **uvolní paměť na zásobníku**, která patřila parametrům a lokálním proměnným, a program pokračuje dál za místem volání.

2 Příklady s řešením

1. Větší ze dvou čísel

```c
int max(int a, int b) {
    if (a > b) return a;
    else return b;
}

int main() {
    printf("%d\n", max(3, 7));
    printf("%d\n", max(10, -5));
    return 0;
}
```

2. Celsius na Fahrenheit

```c
float celsius_na_fahrenheit(float c) {
    return c * 9.0 / 5.0 + 32;
}

int main() {
    printf("%.2f\n", celsius_na_fahrenheit(0));   // 32.00
    printf("%.2f\n", celsius_na_fahrenheit(100)); // 212.00
    return 0;
}
```

3. Průměr tří čísel

```c
float prumer(float x, float y, float z) {
    return (x + y + z) / 3;
}

int main() {
    printf("Prumer: %.2f\n", prumer(2.0, 4.0, 6.0));
    return 0;
}
```

4. Faktoriál

```c
int faktorial(int n) {
    if (n < 0) return -1; // chyba
    int vysledek = 1;
    for (int i = 1; i <= n; i++) {
        vysledek *= i;
    }
    return vysledek;
}

int main() {
    printf("%d\n", faktorial(5));
    printf("%d\n", faktorial(-3)); // chyba
    return 0;
}
```

5. Absolutní hodnota

```c
int absolut(int x) {
    if (x < 0) return -x;
    else return x;
}

int main() {
    printf("%d\n", absolut(-10));
    printf("%d\n", absolut(7));
    return 0;
}
```

6. Čtverec hvězdiček

```c
void vypis_ctverec(int velikost) {
    for (int i = 0; i < velikost; i++) {
        for (int j = 0; j < velikost; j++) {
            printf("*");
        }
        printf("\n");
    }
}

int main() {
    vypis_ctverec(4);
    return 0;
}
```

7. Je číslo sudé?

```c
int je_parny(int x) {
    if (x % 2 == 0) return 1;
    else return 0;
}

int main() {
    printf("%d\n", je_parny(10));
    printf("%d\n", je_parny(7));
    return 0;
}
```

8. Směrování podle znaménka

```c
void smerovani(int cislo) {
    if (cislo > 0) printf("Kladne\n");
    else if (cislo == 0) printf("Nula\n");
    else printf("Zaporne\n");
}

int main() {
    smerovani(5);
    smerovani(0);
    smerovani(-2);
    return 0;
}
```

9. Výběr podle volby

```c
float vypocitej_cele_pole(float a, float b, float c, int volba) {
    if (volba == 1) return a + b + c;
    else if (volba == 2) return a * b * c;
    else return 0;
}

int main() {
    printf("%.2f\n", vypocitej_cele_pole(1.5, 2.0, 3.0, 1));
    printf("%.2f\n", vypocitej_cele_pole(1.5, 2.0, 3.0, 2));
    printf("%.2f\n", vypocitej_cele_pole(1.5, 2.0, 3.0, 3));
    return 0;
}
```

10. Součet 1..n

```c
int spocitej_soucet(int n) {
    if (n <= 0) return 0;
    int soucet = 0;
    for (int i = 1; i <= n; i++) {
        soucet += i;
    }
    return soucet;
}

int main() {
    printf("%d\n", spocitej_soucet(5));
    return 0;
}
```

11. Podíl dvou čísel

```c
void tiskni_mnozinuvy_podil(int x, int y) {
    if (y == 0) {
        printf("Chyba: deleni nulou!\n");
    } else {
        printf("Podil: %d\n", x / y);
    }
}

int main() {
    tiskni_mnozinuvy_podil(10, 2);
    tiskni_mnozinuvy_podil(10, 0);
    return 0;
}
```

12. Mocnina

```c
float mocnina(float zaklad, int exponent) {
    if (exponent == 0) return 1;
    if (exponent < 0) return -1; // zatim neumi zlomky

    float vysledek = 1;
    for (int i = 0; i < exponent; i++) {
        vysledek *= zaklad;
    }
    return vysledek;
}

int main() {
    printf("%.2f\n", mocnina(2, 3));  // 8
    printf("%.2f\n", mocnina(5, 0));  // 1
    printf("%.2f\n", mocnina(3, -2)); // -1 (chyba)
    return 0;
}
```