KAPITOLA 3

Datové typy pro reálná čísla

V této kapitole:
- Vlastnosti datových typů pro reálná čísla
- Vstupně/výstupní operace z pohledu reálných čísel
- Aritmetické operace s reálnými čísly
- Implicitní a explicitní typové konverze
- Priorita a asociativita dosud probraných operátorů

V kapitole 2 jsme se seznámili s celočíselnými datovými typy. Pro uložení číselných hodnot ve velkém rozsahu obvykle nepožadujeme tak velkou přesnost, jakou nám poskytují celá čísla. Obvykle vystačíme s přesností například 6 desítkových číslic.

Pro reprezentaci reálných čísel pak používáme datové typy obecně označované jako *čísla v plovoucí řádové čárce* (anglicky *float-point*). Takové číslo je reprezentováno dvěma složkami: mantisou a exponentem. *Mantisa* (M) je hodnota čísla, *exponent* (E) určuje řád. Viz tento předpis (pro desítkovou soustavu):

reálné číslo = $M \cdot 10^E$

Příklady zápisů čísel v tomto formátu: $1{,}602 \cdot 10^{-19}$ (elementární náboj); $6{,}023 \cdot 10^{23}$ (Avogadrova konstanta).

Vlastnosti datových typů pro reálná čísla

K dispozici jsou celkem tři typy: `float`, `double` a `long double`, které se liší oborem hodnot. Tyto typy se také liší velikostí v bajtech a přesností, která je určena počtem platných cifer (číslic).

Formát zápisu reálných literálů v desetinném tvaru se vyznačuje používáním tečky místo obvyklé desetinné čárky, například: `1.2345`. Reálné literály lze také zapisovat v semilogaritmickém tvaru, například: `1.602E-19`.

Tabulka 3.1. Vlastnosti datových typů pro reálná čísla

Typ	Přibližný obor hodnot
`float` velikost: 4 bajty, přesnost: 6 platných desítkových cifer	$\pm 1{,}2 \cdot 10^{-38}$ až $\pm 3{,}4 \cdot 10^{+38}$
`double` velikost: 8 bajtů, přesnost: 15 platných desítkových cifer	$\pm 2{,}2 \cdot 10^{-308}$ až $\pm 1{,}8 \cdot 10^{+308}$
`long double` velikost: 16 bajtů, přesnost: 18 platných desítkových cifer	$\pm 3{,}4 \cdot 10^{-4932}$ až $\pm 1{,}2 \cdot 10^{+4932}$

Přesné charakteristiky těchto datových typů lze získat pomocí symbolů z hlavičkového souboru **FLOAT.H**.

Níže je doplněn krátký program, který pro každý z uvedených typů vypíše postupně: velikost v bajtech, počet platných desítkových číslic, minimální a maximální hodnotu. Můžete tyto údaje porovnat s tabulkou 3.1.

PROG_01:

```
#include <iostream>
#include <float.h>
using namespace std;
int main(int argc, char** argv)
{
  cout<<"sizeof(float)="<<sizeof(float)<<endl;
  cout<<"FLT_DIG="<<FLT_DIG<<endl;
  cout<<"FLT_MIN="<<FLT_MIN<<endl;
  cout<<"FLT_MAX="<<FLT_MAX<<endl;
  cout<<endl;
  cout<<"sizeof(double)="<<sizeof(double)<<endl;
  cout<<"DBL_DIG="<<DBL_DIG<<endl;
  cout<<"DBL_MIN="<<DBL_MIN<<endl;
  cout<<"DBL_MAX="<<DBL_MAX<<endl;
  cout<<endl;
  cout<<"sizeof(long double)="<<sizeof(long double)<<endl;
  cout<<"LDBL_DIG="<<LDBL_DIG<<endl;
  cout<<"LDBL_MIN="<<LDBL_MIN<<endl;
  cout<<"LDBL_MAX="<<LDBL_MAX<<endl;
  cout<<endl;
  return 0;
}
```

Výpis programu v konzoli:

```
sizeof(float)=4
FLT_DIG=6
FLT_MIN=1.17549e-038
FLT_MAX=3.40282e+038
sizeof(double)=8
DBL_DIG=15
DBL_MIN=2.22507e-308
DBL_MAX=1.79769e+308
sizeof(long double)=16
LDBL_DIG=18
LDBL_MIN=3.3621e-4932
LDBL_MAX=1.18973e+4932
```

Vstupně/výstupní operace z pohledu reálných čísel

Informace ke vstupně/výstupním operacím z kapitoly 2 můžeme nyní doplnit o přehled dalších manipulátorů a rozšířit informacemi, které platí pro reálná čísla. Při výpisu hodnoty můžeme nastavit parametry, které určí, jak přesně výpis proběhne:

- **šířka** – určuje minimální počet znaků, které se mají vypsat. Uvažujme například výpis čísla 56 na 5 míst. V tomto případě je třeba přidat další 3 znaky, aby celková šířka výpisu byla 5 znaků. Znak, který se použije jako výplň, je obvykle mezera (je možné jej ale změnit pomocí parametru *výplňový znak*). Pokud uvažujeme výpis čísla 56 v šířce 0, vypíše se prostě `56` (výpis hodnoty nesmí být zkreslen).
- **přesnost** – určuje počet číslic za desetinnou tečkou, které se zobrazí. Například výpis čísla π (3,14159265 …) při šířce 5 a přesnosti 3 vypadá takto: `3.142` (při výpisu probíhá nezbytné zaokrouhlení); platí pro formáty `fixed` a `scientific`.
- **výplňový znak** – určuje znak, který se použije jako výplň při výpisu hodnoty ve větší šířce, jako výchozí výplňový znak je použita mezera.
- **zarovnávání** – určuje způsob zarovnání výpisu hodnoty a místo, kam se budou vkládat výplňové znaky. Rozlišujeme (uvažujme výpis čísla -1 při šířce 5):
  - zarovnání doleva (`left`) – výpis hodnoty je zarovnán doleva, výplň se vkládá zprava: `-1□□□`,
  - zarovnání doprava (`right`) – výpis hodnoty je zarovnán doprava, výplň se vkládá zleva: `□□□-1`,
  - mezi znaménko a hodnotu (`internal`) – výpis znaménka je vlevo, následují výplňové znaky a nakonec vypisovaná hodnota: `-□□□1`.

Přehled běžně používaných manipulátorů pro výpis uvádí tabulka 3.2. Na začátku jsou připomenuty dříve popsané manipulátory `endl`, `dec`, `oct` a `hex`.

Tabulka 3.2. Přehled manipulátorů

Manipulátor	Použitelnost pro čísla	Význam
`endl`	celá/reálná	ukončí řádek a nastaví kurzor na začátek následujícího řádku
`dec`	celá	přepne zobrazení celých čísel do desítkové soustavy
`oct`	celá	přepne zobrazení celých čísel do osmičkové soustavy
`hex`	celá	přepne zobrazení celých čísel do šestnáctkové soustavy
`showpos`	celá/reálná	zajistí zobrazení znaménka i pro nezáporná čísla (kladná čísla včetně nuly)
`noshowpos`	celá/reálná	vypne zobrazení znaménka pro kladná čísla (znaménko se zobrazuje pouze pro záporná čísla)
`left`	celá/reálná	přepne na zarovnávání doleva (výplň zprava)
`right`	celá/reálná	přepne na zarovnávání doprava (výplň zleva)
`internal`	celá/reálná	přepne na vyplnění mezi znaménkem a hodnotou
`fixed`	reálná	přepne výpis reálných čísel do tvaru s pevnou pozicí desetinné tečky
`scientific`	reálná	přepne výpis reálných čísel do exponenciálního tvaru
`uppercase`	celá/reálná	přepne výpis čísel tak, že znaky se zobrazují jako velká písmena (týká se výpisu celých čísel v šestnáctkové soustavě a reálných čísel v exponenciálním tvaru)
`nouppercase`	celá/reálná	přepne výpis čísel tak, že znaky se zobrazují jako malá písmena (týká se výpisu celých čísel v šestnáctkové soustavě a reálných čísel v exponenciálním tvaru)
`setw(w)`	celá/reálná	nastaví šířku následujícího výpisu dle w, chybějící znaky do požadovaného počtu jsou realizovány zvolenou výplní (pokud zadání šířky nezopakujeme před každým výpisem, bude mít následující výpis opět šířku 0)
`setfill(f)`	celá/reálná	nastaví výplňový znak na f (znakové literáry píšeme mezi apostrofy)
`setprecision(p)`	reálná	nastaví výpis reálných čísel na počet desetinných míst daných p

Poslední tři manipulátory jsou parametrické (jsou řízeny parametrem zapsaným v závorkách). Pro použití těchto manipulátorů je nutné do zdrojového textu vložit hlavičkový soubor **iomanip**:

```
#include <iomanip>
```

Pro lepší pochopení připojujeme několik příkladů použití výše popsaných manipulátorů pomocí tabulky 3.3.

Tabulka 3.3. Příklady formátování výpisu

Příklad	Výpis	Vysvětlení
`cout<<56;`	56	normální výpis kladné hodnoty (bez znaménka +)
`cout<<-56;`	-56	výpis záporné hodnoty musí vždy obsahovat znaménko –
`cout<<showpos<<56;`	+56	výpis kladné hodnoty včetně znaménka
`cout<<setw(5)<<left<<-56;`	-56□□	šířka 5, zarovnání doleva: po výpisu čísla se připojí 2 mezery
`cout<<setw(5)<<right<<-56;`	□□-56	šířka 5, zarovnání doprava: před výpisem čísla se vloží 2 mezery
`cout<<setw(5)<<internal<<-56;`	-□□56	šířka 5, zarovnání mezi: mezi znaménko a číslice se vloží 2 mezery
`cout<<fixed<<3.141519265;`	3.141519	výpis reálného čísla v desetinném tvaru
`cout<<fixed<<setprecision(3)` `<<3.141519265;`	3.142	zobrazí se 3 číslice za desetinnou tečkou (+zaokrouhlení)
`cout<<scientific<<setprecision(3)` `<<3.141519265;`	3.142e+000	výpis v exponenciálním tvaru, zobrazí se 3 číslice za desetinnou tečkou (+zaokrouhlení)

`cout<<setw(10)<<left<<setfill('0')`
`<<fixed<<setprecision(3)`
`<<3.141519265;`
	3.14200000	výpis v desetinném tvaru, 3 platné číslice za desetinnou tečkou, zbytek se doplní nulami do celkové šíře 10 znaků (zarovnání doleva)

`cout<<setw(10)<<right<<setfill('0')`
`<<fixed<<setprecision(3)`
`<<3.141519265;`
	000003.142	výpis v desetinném tvaru, 3 platné číslice za desetinnou tečkou, vlastní výpis předchází nuly do celkové šíře 10 znaků (zarovnání doprava)

Aritmetické operace s reálnými čísly

Pro reálná čísla lze používat běžné aritmetické operátory stejné jako pro celá čísla. Jsou zde však dva drobné rozdíly:

1. Operátor % (modulo, zbytek po celočíselném dělení) nemá pro aritmetiku reálných čísel smysl, jeho použití s reálným číslem je chápáno jako chyba.
2. Operátor / (dělení) má v reálné aritmetice smysl reálného dělení, tedy například výraz 5.0/2.0 má výsledek 2,5.

Přehled operátorů použitelných pro reálná čísla je uveden v tabulkách 3.4 a 3.5. V tabulce 3.4 jsou běžné binární operátory (mají levý a pravý operand), v tabulce 3.5 jsou unární operátory (mají pouze jeden operand).