

Programování F1400 + F1400a: Počítání s čísly

Programování F1400 + F1400a
doc. RNDr. Petr Mikulík, Ph.D.

podzimní semestr 2020

$$1/2 = 0 \quad ?$$

$$1/2 = 0.5 \quad ?$$

$$1.0/2 = 0.5$$

$$1/2.0 = 0.5$$

$$1.0/2.0 = 0.5$$

$$1./2. = 0.5$$

$$\begin{aligned} 2020 &= 2.020 \times 10^3 \\ &= 2 \times 10^3 + 2 \times 10^1 \\ &= 11111100100_2 \\ &= 2^{10} + 2^9 + 2^8 + 2^7 + 2^6 + 2^5 + 2^2 \\ &= \left(1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \frac{1}{32} + \frac{1}{256} \right) \times 2^{10} \end{aligned}$$

Zobrazení celých čísel v (digitálním) počítači

- **Číselné soustavy:** *Dvojková* soustava (binární, cifry 0 a 1), *osmičková* (oktalová, 0–7), *desítková* (dekadická, 0–9), *šestnáctková* (hexadecimální, 0–9 a A–F)
- **Reprezentace čísel v počítači:** digitální počítač počítá ve dvojkové soustavě, čísla v ní musíme reprezentovat pomocí nějakého systému (konvence)
- Čísla **bez znaménka (unsigned)**
- Čísla **se znaménkem (signed)**
- **Omezený rozsah** při počítání, možnost *přetečení (overflow)* – nelze zobrazit příliš velké číslo (v absolutní hodnotě), nebo *podtečení (underflow)* – nejmenší číslo po zmenšení zkolabuje (záporné na kladné)
- **Rozsahy:** bit 1 B, byte 1 B, word 2 B, short 2 B, integer (např. nyní typicky 4 B), long integer (např. nyní typicky 8 B)

Zobrazení reálných čísel v počítači

- Zápis reálných čísel – **desetinný tvar**:
12.34, +12.34, -123.456, -0.001234
- Pozor na národní konvence při zápisu či zobrazení: **desetinná tečka** versus **desetinná čárka**; problémy při přenositelnosti
- Zápis čísla v **exponenciálním tvaru** (desítková soustava):
znaménko mantisa E znaménko exponent
 $100 = 10E1 = 1e2 = +1E+2$ $0.012 = 0.12E-1 = 1.2e-2 = 12E-3$
- Reprezentace počítačem v **binární soustavě**:
mantisa je reálné číslo v intervalu $[1;2)$, exponent je dvojkový
 $\pm 1.xxxx \times 2^{yyyy} = (-1)^s \times \left(1 + \sum_{n=1}^{\dots} b_n 2^{-n}\right) \times 2^{e-1023}$
- IEEE 754 Floating-Point Standard (Standard IEEE pro dvojkovou aritmetiku v pohyblivé řádové čárce):
4 B = 1+23+8 b **float** (jednoduchá přesnost, single precision)
8 B = 1+52+11 b **double** (dvojitá přesnost)
16 B = 1+112+15 b **long double** (čtyřnásobná přesnost)
Poznámka: exponent 000...000 a 111...111 je rezervovaný pro ± 0 , $\pm \text{Inf}$, NaN
- Mimo standard např. Pascal v DOSu: 6 B **real**

- **Unární operátory:** +, -
- **Binární operátory:** $1 + 2$, $3 - 4$, $4 * 6$,
Pozor při dělení celých čísel, např. **9/3 vs 9.0/3.0**: Jsou definičním oborem reálná nebo celá čísla?! Jaký má být obor hodnot? Počítač neví, co vy chcete, aby mu vyšlo: **1/2 = 0 nebo 1.0/2.0 = 0.5**
Celočíselné dělení, $3/2=1$: C, C++, gnuplot, ...
Reálná čísla, $3/2=1.5$: GNU Octave, Matlab, ...
Matoucí – Python2: $3/2=1$, ale Python3: $3/2=1.5$
- Zbytek po dělení – modulo: $9 \% 2$ nebo $\text{mod}(9,2)$
Pozn.: používá se na test parity čísla, tj. sudé/liché číslo, $\text{mod}(x,2)=0$ či 1
- **Ternární operátor:** (podmínka) ? výsledekPravda : výsledekNepravda
Příklad: $x = (4 > 6) ? 1 : 42 \rightarrow x$ vyjde 42
Úloha: jak použijete ternární operátor na test parity čísla x či na výpočet absolutní hodnoty čísla x ?
- Dále **logické** unární, logické binární, ... operátory.

- *Základní, běžné funkce*
odmocnina \sqrt{x} ... `sqrt(x)`,
absolutní hodnota $|x|$... `fabs(x)` nebo `abs(x)`,
exponenciála a logaritmus e^x ... `exp(x)`, $\ln x$... `log(x)` (přirozený),
případně dekadický `log10(x)`
- *Zaokrouhlování* $\lfloor x \rfloor$... `floor(x)`, $\lceil x \rceil$... `ceil(x)`, celočíselná část
... `trunc(x)`, `round(x)`, `int(x)`
- *Umocnění* 2^3 se zapíše jako `2^3` nebo `2**3` nebo `pow(2,3)`;
otázka do cvičení: jak se spočítá $4.5^{7.3}$?
- *Goniometrické funkce* `sin(x)`, `cos(x)`, `acos(x)`, `asin(x)`, `atan(x)`, `atan2(y,x)`
(rozlišení kvadrantů hodnot inverzní funkce); výpočet obvykle v radiánech,
nikoliv ve stupních
- *Pokročilé funkce (vlastní numerické implementace nebo knihovny funkcí)*
`sinc`, Besselovy funkce, gamma funkce, (inverzní) chybová funkce,
Hermiteovy polynomy, ...

- **Komplexní číslo** je uspořádaná dvojice reálných čísel (reálná a imaginární složka) se specifickými pravidly pro počítání
- Různé programovací jazyky reprezentují zápisy a zobrazení komplexních čísel různě: uspořádaná dvojice, nebo pomocí symbolu i nebo j , např. $\{1,2\} + \{2.0,3.0\}$ nebo $1+2i + 2+3i$ nebo $1+2j + 2+3j$
- Různé programovací jazyky reprezentují zápisy a zobrazení komplexních čísel různě – např.:
 - uspořádaná dvojice, např. $\{1,2\} + \{2.0,3.0\}$
 - pomocí symbolu i nebo j , např. $1+2i + 2+3i$ nebo $1+2j + 2+3j$
- Zkuste si nechat spočítat $\sqrt{-1}$ a uvidíte, co vyjde.
- Pro výpočty by měly fungovat stejné operátory a funkce jako pro reálná čísla, ideálně i stejně nazvané funkce, někdy ovšem s prefixem c (complex)

Jak na nedefinovanou funkci?

- Nemá-li programovací jazyk definovanou nějakou funkci, kterou potřebujeme, musíme ji vyjádřit pomocí těch definovaných, nebo pomocí řady, iterace, numerického integrálu, apod.
- **Příklad:** chceme spočítat $\alpha = \arcsin x$, ale je k dispozici pouze výpočet funkce \arccos .

$$\begin{aligned}\alpha = \arcsin x &\rightarrow \sin \alpha = x \rightarrow \sin^2 \alpha = x^2 \rightarrow \\ 1 - \cos^2 \alpha = x^2 &\rightarrow \cos^2 \alpha = 1 - x^2 \rightarrow \\ \cos \alpha = \sqrt{1 - x^2} &\rightarrow \alpha = \arccos \sqrt{1 - x^2}\end{aligned}$$

Ovšem pozor na definiční obor a znaménko výsledku. Správně:

$$\alpha = \operatorname{sign} x \cdot \arccos \sqrt{1 - x^2}$$

- **Domácí úloha:** vyjádřete $\beta = \arcsin x$ a $\gamma = \arccos x$ pomocí funkce $\arctan x$.