Základy programování (IZP)

První počítačové cvičení

Brno University of Technology, Faculty of Information Technology
Božetěchova 1/2, 612 66 Brno - Královo Pole
Alena Tesařová, atesarova@fit.vut.cz

22.09.2021, 1. týden

Důležité informace

- Jmenuji se **Alena Tesařová**
- **Můj profil**: https://www.fit.vut.cz/person/atesarova/
  - Kancelář: L204
  - Konzultační hodiny: po domluvě emailem nebo přes Teams, případně na kartě
- **Přestávky?**
- **Plán semestru**

Důležité informace

- **FIT**: https://www.fit.vut.cz/
- **WIS**: https://wis.fit.vutbr.cz/
- **EMAIL**: http://roundcube.fit.vut.cz/
- **Karta předmětu IZP**:
  - https://www.fit.vut.cz/study/course/IZP/
- **Wiki stránky IZP**:
  - https://wis.fit.vutbr.cz/FIT/st/cwk.php?id=10033&csid=569324

Náplň cvičení

- **Přihlášení k počítačům, informační systém, kontrola přihlášení, atd.**
- **Vývojové nástroje**
  - Pokud s programováním začínáte, **nepoužívejte** žádné složité nástroje
  - Můžete programovat ve Windows, Linux, Mac OS X
  - Doporučené vývojové prostředí (IDE): **Code::Blocks**
- **Jednoduché příklady**

Užitečné nástroje

- **Vývojová prostředí**
  - Code::Blocks (Linux, Windows, Mac OS)
    - http://www.codeblocks.org/downloads/26
    - Windows: stáhnout verzi s překladačem GCC a debuggerem GDB (codeblocks-20.03mingw-nosetup.zip)
- **Textové editory**
  - Windows: PSPad, Notepad++, VS Code, …
  - Linux: nano, gedit, vim, …
- Video návod pro VS Code na wiki

CODE::BLOCKS IDE
→ WIKI STRÁNKY IZP

Získávání nápovědy

- Nápovědu můžete získat
  - z **internetu**: vygooglit název příkazu + c
    - Výborný zdroj: http://www.cplusplus.com/
    - Doporučení AS: http://devdocs.io/
  - pomocí **příkazu `man`** (linux)
    - např. pokud chcete získat informace o funkci **`printf`**, zadejte:

```
man 3 printf
```

- **Kniha**
  - Herout, P. Učebnice jazyka C.
    - Mohla by ještě být v knihovně

Číselné proměnné a konstanty

- Základní pojmy
  - **Proměnná**
    - Pojmenované místo v paměti, ve kterém uchováváme data.
    - Má určitý datový typ a její hodnota se **může** za běhu programu měnit.
    - Příklad: `int a=10;`

Číselné proměnné a konstanty

- Základní pojmy
  - **Proměnná**
    - Pojmenované místo v paměti, ve kterém uchováváme data.
    - Má určitý datový typ a její hodnota se **může** za běhu programu měnit.
    - Příklad: `int a=10;`
  - **Konstanta**
    - Pojmenované místo v paměti, ve kterém uchováváme data.
    - Má určitý datový typ, její hodnota se **nemůže** za běhu programu měnit.
    - Příklad: `const int b=10;`
  - **Přiřazení (=)**

Číselné proměnné a konstanty

- Základní pojmy
  - **Inicializace**
    - Nastavení hodnoty proměnné nebo konstanty
  - **Příkaz**
    - Definuje činnost, kterou program vykoná (např. výpis textu na obrazovku)

Pár poznámek

- V příkladech budeme používat **jedinou knihovní funkci** `printf()`
- Funkce se nachází v knihovně `stdio.h`

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

- **Výpis konstantního řetězce**

```
printf("text");
```

Číselné proměnné a konstanty

- Napište program, který:
  - Vypíše na obrazovku **číslo 38** (%d)
  - Deklarujte **proměnnou typu `int`** (může se jmenovat libovolně)
  - Do proměnné uložte hodnotu 38
  - K deklarované proměnné **přičte hodnotu 4**
  - **Opět vypíše** novou hodnotu na obrazovku (%d)

```
int i = 10;
printf("cislo: %d", i);
```

Načtení vstupů

- K načítání vstupů se používá funkce scanf
  `scanf("formátovací řetězec",&proměnná)`
- Formátovací řetězec indikuje formát dat, která načítáme např. `%f`, `%d`
- Proměnná, do které budou zadaná data uložena
  - Znak `&`: chceme adresu, na kterou ukládáme
  - Pozor na řetězce/pole
- Příklad:

```
int celeCislo; // kam ukládáme
scanf("%d",&celeCislo); // načtení
printf("%d\n",celeCislo); // výpis
```

Příklad: diskriminant

- Napište program, který načte koeficienty kvadratické rovnice a, b, c v $ax^2 + bx + c = 0$ a vypočítá diskriminant D
- $D = b^2 - 4 * a * c$
- Načtení parametrů

Příklad: diskriminant

- Napište program, který načte koeficienty kvadratické rovnice a, b, c v $ax^2 + bx + c = 0$ a vypočítá diskriminant D
- $D = b^2 - 4 * a * c$
- Načtení parametrů

```
int a; // kam ukládáme
scanf("%d",&a); // načtení
…
```

Příklad: diskriminant

- Napište program, který načte koeficienty kvadratické rovnice a, b, c v $ax^2 + bx + c = 0$ a vypočítá diskriminant D
- $D = b^2 - 4 * a * c$
- Načtení parametrů

```
int a; // kam ukládáme
scanf("%d",&a); // načtení
…
```

- Výpis

```
printf("Diskriminant: %d", D);
```

Operátory

- **Aritmetické**: unární, binární
- **Relační**: ==, !=, >, <, >=, <=
- **Logické**: && (AND), || (OR)

Operátory

- **Aritmetické**: unární, binární
- **Relační**: ==, !=, >, <, >=, <=
- **Logické**: && (AND), || (OR)

A	B	AND	OR
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

Jednoduché podmínky (`if`)

- Základní pojmy
  - podmíněný příkaz
  - `if (podmínka){ příkazy } else {příkazy}`

Příklad: operátor %

- Operátor % vrací zbytek po celočíselném dělení
- $5\%5 = 0,\ 6\%5 = 1$

- Napište program, který zjistí, zda bylo načtené číslo sudé.

- Načtení čísla: scanf
- Výpis: printf
- Podmínka

```
if (podmínka){ příkazy } else {příkazy}
```

Příklad: číslo v intervalu

- Napište program, který určí, zda je číslo x v intervalu $\langle a, b \rangle$.
- Varianta 1: pevně zadaný interval
- Varianta 2: uživatelem zadaný interval

Příklad: hledání maxima ze tří čísel

- Najděte maximum z čísel $3,10,18$
- Varianta 1: porovnejte všechny dvojice mezi sebou
- Varianta 2: uchovávejte si aktuálně největší číslo v pomocné proměnné

Příklad: přestupný rok

- Číslo roku je dělitelné 4 a
  - číslo roku není dělitelný 100 → **rok je přestupný**,
  - číslo roku je dělitelný 100 a
    - číslo roku je dělitelný 400 → **rok je přestupný**,
    - číslo roku není dělitelný 400 → **rok není přestupný**,
- Číslo roku není dělitelné 4 → **rok není přestupný**.

- Zkuste stejnou složenou podmínku napsat pomocí logických operátorů `&&, ||, ()`

- Přidejte kontrolu `rok >= 1582` (dříve se přestupné roky nepočítaly), pokud selže, vypište chybu a return 1;

Řešení

- Příklady jsou vyřešeny na wiki předmětu

Děkuji za pozornost