# Procedurální programování

## Úvod do programování

Stanislav Vítek

Katedra radioelektroniky
Fakulta elektrotechnická
České vysoké učení v Praze

# Přehled témat

- Část 1 – O předmětu

  Organizace předmětu

  Dostupné prostředky

  Studijní výsledky

- Část 2 – O programování

  Než začneme programovat

  První program

- Část 3 – Zadání 0. domácího úkolu

# I. O předmětu

# Předmět a lidé

- Webové stránky předmětu
  https://cw.fel.cvut.cz/b181/courses/b0b99prpa
- Moodle – pouze částečně (PDF přednášek a zdrojové kódy)
  https://moodle.fel.cvut.cz
- Přednášející a garant předmětu
  - Ing. Stanislav Vítek, Ph.D.
    http://mmtg.fel.cvut.cz/personal/vitek/
- Cvičící
  - Ing. Petr Janout, Ph.D.
  - Ing. Martin Mudroch, Ph.D.
  - Ing. Ondřej Nentvich
  - Ing. Ján Kučerák

# Cíle předmětu

- Motivovat k programování
  - Programování je klíčová dovednost, která může hrát rozhodující roli na trhu práce
- Naučit se algoritmizovat
  - Formulace problému a návrh řešení
  - Rozklad problému na dílčí úlohy
- Získat zkušenosti s programováním
  - Programovací jazyk C

    *cvičení, domácí úkoly, zkouška*
  - Povědomí o tom, jaké úlohy lze výpočetně řešit

    *Programátorovi nestačí perfektní znalost programovacího jazyka, ale především musí vědět, jak vůbec danou úlohu řešit.*

# Organizace a hodnocení předmětu

- B0B99PRPA – Procedurální programování pro EK a EEM
- Rozsah: 2p+2c; Zakončení: KZ; Kredity: 4;

---

- Průběžná práce v semestru – domácí úkoly a test
- Započtový a případně implementační test

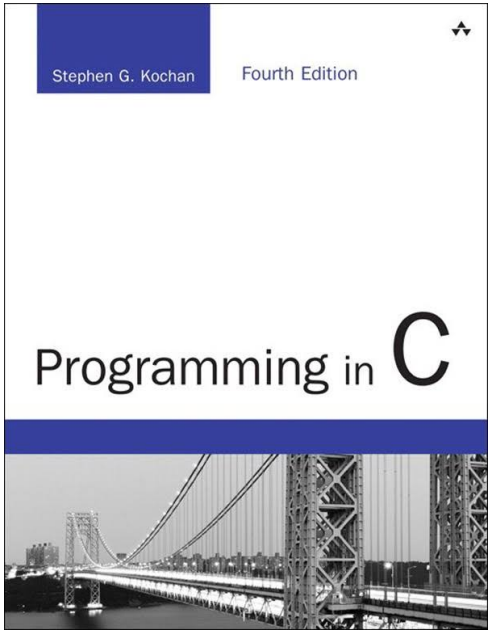
---

- Docházka na cvičení
  - Cvičení jsou povinná – možné dvě omluvené absence
  - Na cvičení je třeba se připravit, nejlépe návštěvou přednášky a studiem podkladů (příklady)

# Zdroje a literatura

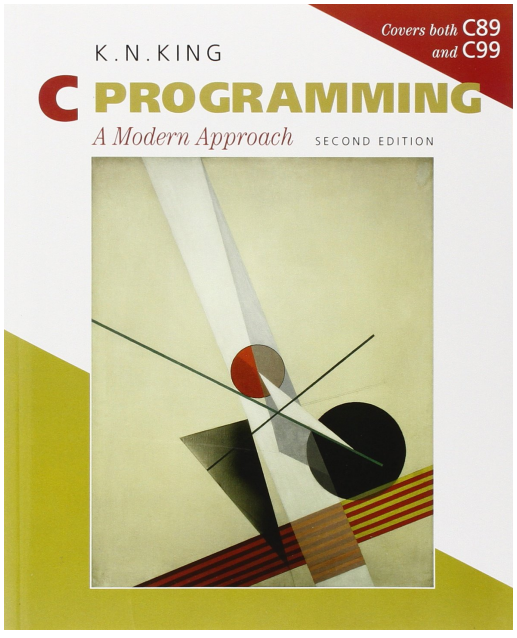

Pavel Herout
Učebnice jazyka C
Kopp, 2011

ISBN 978-80-7232-383-8




Stephen G. Kochan
Programming in C
Addison-Wesley
2014

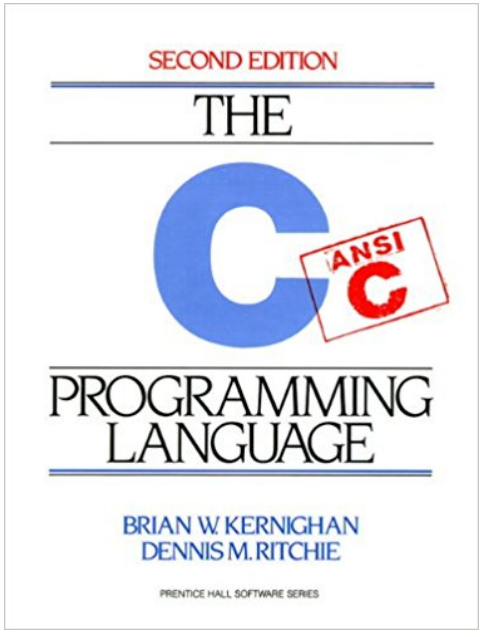
ISBN 978-0321776419




K. N. King
C Programming:
A Modern Approach
W. W. Norton & Company
2008

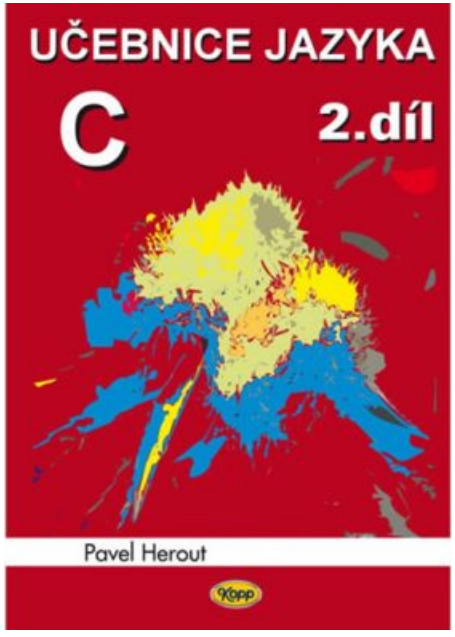
ISBN 860-1406428577

# Zdroje a literatura




Brian W. Kernighan
Dennis M. Ritchie
The C Programming Language (ANSI C)
Prentice Hall
1988

ISBN 978-0131103627




Pavel Herout
Učebnice jazyka C – 2. díl
Kopp
2008

ISBN 978-80-7232-367-8


Peter van der Linden
Expert C Programming: Deep C Secrets
Prentice Hall
1994

ISBN 978-0131774292

# Řešení problémů

- Obracejte se na svého cvičícího
- Pokud komunikujete elektronickou poštou (e-mail)
  - Pište vždy ze své fakultní adresy
  - Do předmětu zprávy uvádějte zkratku predmětu PRP
  - V případe zásadních problémů (napr. týkajících se zápočtu) uvádějte do Cc též přednášejícího

# I. O předmětu

# Počítačové učebny

- OS Linux (Ubuntu)
- Síťové bootování a síťové domovské adresáře (NFS v4)

  Přenos a synchronizace souborů – ownCloud, SSH, FTP, USB
- Vývoj v C:
  - Překladace gcc a clang

    `https://gcc.gnu.org` a `http://clang.llvm.org`
  - Sestavení projektu nástrojem make (GNU make)

    Ukážeme si později na přednáškách a cvičení
  - Textový editor – gedit, atom, sublime, vim

    `https://atom.io`, `http://www.sublimetext.com`
  - C/C++ vývojová prostredí
    - Visual Studio Code
    - Geany – `https://www.geany.org`
    - Code::Blocks – `http://www.codeblocks.org`
    - NetBeans, Eclipse

# Služby akademické sítě

- SVTI – http://svti.fel.cvut.cz/cz
- Diskové úložište ownCloud – https://owncloud.cesnet.cz
- Zasílání velkých souboru – https://filesender.cesnet.cz
- Rozvrh a termíny – https://portal.fel.cvut.cz
- FEL Google Account – http://google-apps.fel.cvut.cz

  autentizovaný přístup do Google Apps for Education
- Gitlab FEL – https://gitlab.fel.cvut.cz
- Přístup k informačním zdrojům – https://dialog.cvut.cz

  IEEE Xplore, ACM, Science Direct, Springer Link
- Akademické a kampusové licence – https://download.cvut.cz
- MetaCentrum – http://www.metacentrum.cz

  Národní Gridová Infrastruktura

# I. O předmětu

# Domácí úkoly

- Samostatná práce s cílem osvojit si praktické zkušenosti

  Průběžná práce a řešení úkolů.

- Jednotné zadání na prednášce a jednotný termín odevzdání
- Odevzdání domácích úkolů prostřednictvím systému BRUTE

  `https://cw.felk.cvut.cz/upload`

  - Nahrání archivu s nezbytnými zdrojovými soubory
  - Ověření správnosti implementace automatickými testy
  - Penalizace za překročení počtu uploadů
  - Detekce plagiátů

  Cílem řešení úkolů je získat vlastní zkušenost

  - Úkoly jsou jednoduché a navrhované tak, aby byly stihnutelné

Pokud nečemu nerozumíte, ptejte se!

# Přehled domácích úkolů

- Domácí úkoly s povinným a případně bonusovým zadáním

  `https://cw.fel.cvut.cz/wiki/courses/b0b99prpa/hw/start`

  1. HW00 – Hello world!
  2. HW01 – Načítání vstupu, výpočet a výstup
  3. HW02 – Cykly
  4. HW03 – Kreslení (ASCII art)
  5. HW04 – Maticové počty
  6. HW05 – Zpracování textu
  7. HW06 – Validace strukturovaného textu
  8. HW07 – Zpracování číselné řady
  9. HW08 – Kruhová fronta v poli
- Podmínkou zápočtu je odevzdání všech úkolů
- Celkem lze získat
  - za povinná zadání **30b**,
  - za bonusová dalších **15b**.

# Kontrola domácích úkolů

- Odevzdávací systém BRUTE

  Bundle for Reservation, Uploading, Testing and Evaluation

  - Formální kontrola – kompilace programu
  - Testování funkčnosti a správnosti – kontrola výstupu pro daný vstup
    - Veřejné vstupy a odpovídající výstupy / neveřejné vstupy
  - Před uploadem programu si program otestujete sami
    - S využitím dostupných vstupů a výstupů
    - Vytvořením vlastních vstupů a laděním programu
- Porozumění kódu a kontrola možných stavů
  - Schopnost vysvětlit roli každého řádku kódu
    - Pro každou funkci nebo načtení vstupu od uživatele analyzujte možné vstupní hodnoty nebo návratové hodnoty funkcí
    - Pokud je z hlediska funkčnosti vstup nebo návratová hodnota zásadní, proveďte kontrolu vstupu a/nebo příslušnou akci, např. vypsání hlášení a ukončení programu

      Např. očekávaný vstup je číslo a uživatel zadá něco jiného.

# Hodnocení

| Zdroj bodů | Maximum | Nutné minimum |
|---|---|---|
| Domácí úkoly | 45 | 30 |
| Test v semestru | 10 | |
| Zápočtový test | 35 | 15 |
| Implementační test | 15 | - |
| Součet | 105 | |

- Za práci v semestru je třeba získat nejméně 30 bodů, všechny domácí úkoly musí být odevzdány a to nejpozději do 13.1.2019 ve 23:59 CET!
- Implementační test – schopnost pochopit problém a napsat krátký program (cca 4 hodiny)

# Klasifikace

| Klasifikace | Bodové rozmezí | Slovní hodnocení |
|---|---|---|
| A | $\geq 90$ | výborně |
| B | $80 - 89$ | velmi dobře |
| C | $70 - 79$ | dobře |
| D | $60 - 69$ | uspokojivě |
| E | $50 - 59$ | dostatečně |
| F | $< 50$ | nedostatečně |

# Přehled přednášek

| | | | |
|---|---|---|---|
| 1. | Informace o předmětu, úvod do programování | 2.10. | HW00 |
| 2. | Základy programování v C | 9.10. | HW01 |
| 3. | Základní řídící struktury | 16.10. | HW02 |
| 4. | Řídící struktury, výrazy | 23.10. | HW03 |
| 5. | Funkce, modularita programů | 30.10. | HW04 |
| 6. | Pole, ukazatele, textové řetězce | 6.11. | HW05 |
| 7. | Ukazatele, práce s pamětí, ladění | 13.11. | HW06 |
| 8. | Agregované datové typy, přesnost výpočtů | 20.11. | HW07 |
| 9. | Spojové struktury, abstraktní datový typ | 27.11. | HW08 |
| 10. | Standardní knihovny C, algoritmy | 4.12. | |
| 11. | Programování ARM – MBED | 11.12. | |
| 12. | Programování ARM – HW prostředky | 18.12. | |
| 13. | Zápočtový test | 8.1. | |

# II. O programování

# Co je to program?

- Program je **recept** – posloupnost kroků (výpočtů), popisující průběh řešení nějakého problému pomocí dostupných prostředků – programovací prostředí, počítač, ...

  Receptu budeme říkat algoritmus.

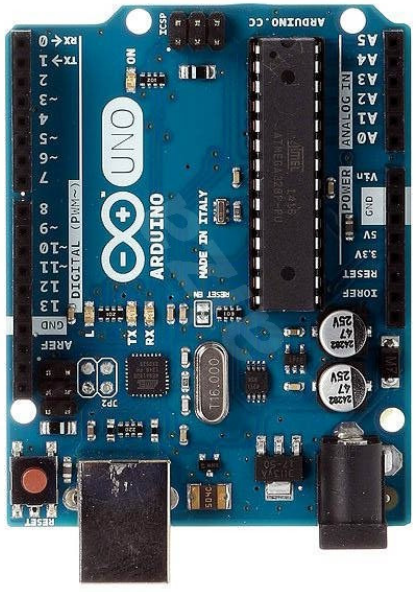
- Programování je schopnost samostatně
  - tvořit programy
  - dekomponovat úlohy na menší celky
  - sestavovat z dílčích částí větší programy řešící komplexní úlohu
- Jak začít?
  - Scratch – MIT Media Lab

    https://scratch.mit.edu/
  - Angry Birds

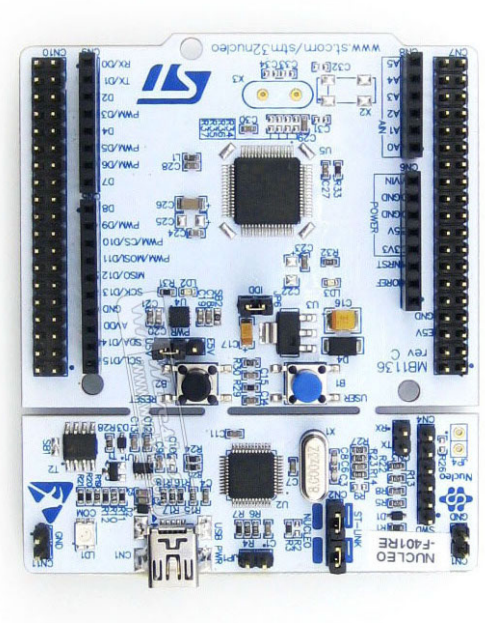
    https://studio.code.org/hoc/1/
  - Code with Anna and Elsa

    https://studio.code.org/s/frozen/

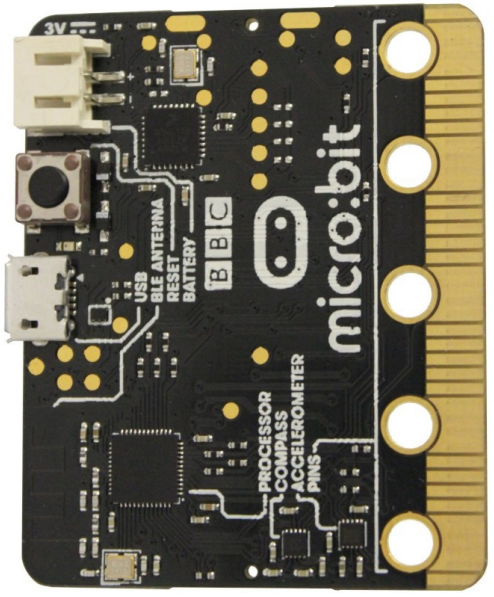
# Programování může být skvělá zábava




Arduino
Open Source
Procesory AVR
http://arduino.cc




Nucleo
ST Microelectronics
Procesory ARM
http://st.com




BBC Micro:bit
Open Source
Procesory ARM
http://microbit.org

# Programování počítače

- Počítač počítá, tj. pracuje s čísly
  - Výpočet je realizován aritmeticko-logickou jednotkou (ALU)
  - Číselné hodnoty jsou uloženy v paměti počítače

    Jejich význam je pak určen datovým typem
  - Předpis jak a co počítat je zapsán programem

    Opět jako posloupnost číselných hodnot se specifickým významem
- Výpočty probíhají ve dvojkové soustavě

  V minulosti se používala i desítková (ENIAC) nebo trojková soustava (Setuň)
  - jednotkou pro uložení informace je bit
  - bity jsou organizovány do skupin – bytů ($= 8$ bitů)

# Data v paměti počítače

Paměťová místa s daty jsou odkazována proměnnými

- pojmenované místo v paměti počítače
- vytvoří se základě **deklarace**, ve které sdělíme její **jméno** (identifikátor) a **datový typ**
- počítač zachází s proměnnou prostřednictvím její adresy
- v programu je adresa vyjádřena jménem proměnné

**Příklad**

deklarace proměnných pro uložení celých čísel datového typu `int`

```
int a;
int b;
// dale zachazime s promennymi beznym zpusobem
a = 10;
b = a - 3;
```

# Co je to algoritmus?

- Návod, jak provést určitou činnost.
- V případě programování jde zpravidla o transformaci množiny vstupních dat na množinu dat výstupních.
- Vlastnosti algoritmu:
  1. Je elementární.

     Skládá se z konečného počtu jednoduchých činností – kroků
  2. Je determinovaný.

     Po každém kroku lze určit, jak má algoritmus pokračovat nebo skončit.
  3. Je konečný.

     Počet opakování jednotlivých kroků algoritmu je vždy konečný.
  4. Je rezultativní.

     Vede ke správnému výsledku.
  5. Je hromadný.

     Algoritmus lze použít k řešení celé (velké) skupiny podobných úloh.

# Základní složky algoritmu

Kombinace základních složek algoritmu umožňuje vytvářet komplexní programy.

- Posloupnost (sekvence) – tvořena jedním nebo několika kroky, které se provedou právě jednou v daném pořadí.
- Cyklus (iterace) – opakování nějaké posloupnosti, dokud je splněna podmínka opakování.
- Větvení (podmíněná operace) – volba posloupnosti instrukcí na základě vyhodnocení podmínky.

Pokud se některé části algoritmu opakují, je vhodné poslupnosti orgranizovat do větších celků: procedur a funkcí (podprogramů).

# Zápis algoritmu

- Existují 4 hlavní způsoby, jakými lze algoritmus popsat:
  - **slovně** – Vyjádříme slovně postup řešení a jednotlivé kroky
  - **graficky** – Použití vývojových diagramů a struktogramů
  - **matematicky** – jednoznačnný popis matematickou konstrukcí (např. rovnicí nebo konstrukčním popisem geometrické úlohy)
  - **programem** – kroky algoritmu jsou popsány instrukcemi procesoru, resp. převedeny z vyššího programovacího jazyka, tedy algoritmus programujeme
- Návrhy algoritmů:
  - **shora dolů** – problém rozdělíme na několik podúloh, které řešíme a spojením dostaneme celý algoritmus
  - **zdola nahoru** – z triviálních úloh skládáme vyšší úlohy a spojením dostaneme celý algoritmus
  - **kombinace obou metod**

*V praxi vždy záleží především na komplexnosti a povaze řešeného algoritmus, který postup bude nejlepší aplikovat.*

# II. O programování

# První program

```
1  #include <stdio.h>
2
3  int main()
4  {
5    printf("Naucim se programovat!\n");
6    return 0;
7  }
```

Někde na disku existuje soubor *stdio.h*, který potřebuji k překladu.

Kód spustitelného programu obsahuje funkci main().

Kód je organizován do bloků ohraničených {}.

Funkce printf tiskne text na displej.

Program (funkce) má návratovou hodnotu.

- Program můžeme zkompilovat a spustit
- Na displej počítače (STDIO, standardní výstup) se vypíše textová informace

# První program

```
1  #include <stdio.h>
2
3  int main()
4  {
5    printf("Naucim se programovat!\n");
6    return 0;
7  }
```

Někde na disku existuje soubor *stdio.h*, který potřebuji k překladu.

Kód spustitelného programu obsahuje funkci main().

Kód je organizován do bloků ohraničených {}.

Funkce printf tiskne text na displej.

Program (funkce) má návratovou hodnotu.

- Program můžeme zkompilovat a spustit
- Na displej počítače (STDIO, standardní výstup) se vypíše textová informace

# První program

```
1  #include <stdio.h>
2
3  int main()
4  {
5    printf("Naucim se programovat!\n");
6    return 0;
7  }
```

Někde na disku existuje soubor *stdio.h*, který potřebuji k překladu.

Kód spustitelného programu obsahuje funkci main().

Kód je organizován do bloků ohraničených {}.

Funkce printf tiskne text na displej.

Program (funkce) má návratovou hodnotu.

- Program můžeme zkompilovat a spustit
- Na displej počítače (STDIO, standardní výstup) se vypíše textová informace

# První program

```
1  #include <stdio.h>
2
3  int main()
4  {
5    printf("Naucim se programovat!\n");
6    return 0;
7  }
```

Někde na disku existuje soubor *stdio.h*, který potřebuji k překladu.

Kód spustitelného programu obsahuje funkci main().

Kód je organizován do bloků ohraničených {}.

Funkce printf tiskne text na displej.

Program (funkce) má návratovou hodnotu.

- Program můžeme zkompilovat a spustit
- Na displej počítače (STDIO, standardní výstup) se vypíše textová informace

# První program

```
1  #include <stdio.h>
2
3  int main()
4  {
5    printf("Naucim se programovat!\n");
6    return 0;
7  }
```

Někde na disku existuje soubor *stdio.h*, který potřebuji k překladu.

Kód spustitelného programu obsahuje funkci main().

Kód je organizován do bloků ohraničených {}.

Funkce printf tiskne text na displej.

Program (funkce) má návratovou hodnotu.

- Program můžeme zkompilovat a spustit
- Na displej počítače (STDIO, standardní výstup) se vypíše textová informace

# III. Zadání 0. domácího úkolu (HW00)

## Zadání 0. domácího úkolu (HW00)

# Zadání 0. domácího úkolu (HW00)

## Téma: První program

- **Motivace:** Seznámení se s odevzdávacím systémem BRUTE
- **Cíl:** Osvojit si kompilaci a odevzdávání domácích úkolů
- **Zadání:**

  `https://cw.fel.cvut.cz/wiki/courses/b0b99prpa/hw/hw00`

  - Napište program, který vytiskne na obrazovku text Hello PRP! zakončený znakem nového řádku `\n`
- **Termín odevzdání:** 12.10.2018, 23:59:59 CET