Procedurální programování

Jan Faigl

Katedra počítačů
Fakulta elektrotechnická
České vysoké učení technické v Praze

Organizace předmětu

B0B36PRP – Procedurální programování

Přehled témat

Část I

Organizace předmětu

Předmět a přednášející

B0B36PRP – Procedurální programování

- Webové stránky předmětu
  `https://cw.fel.cvut.cz/wiki/courses/b0b36prp`
- Odevzdávání domácích úkolů
  `https://cw.felk.cvut.cz/brute`
- Přednášející:

  - prof. Ing. **Jan Faigl**, Ph.D.
    - Katedra počítačů – `https://cs.fel.cvut.cz`
    - Centrum umělé inteligence – **Artificial Intelligence Center** (**AIC**)
      `https://aic.fel.cvut.cz`
    - Centrum robotiky a autonomních systému
      Center for Robotics and Autonomous Systems – CRAS
      `https://robotics.fel.cvut.cz`
    - **Laboratoř výpočetní robotiky** (**Computational Robotics Laboratory**)
      `https://comrob.fel.cvut.cz`

Cvičící

- Ing. **Rudolf Jakub Szadkowski**

- Ing. **Jakub Dupák**

- Ing. **Jan Feber**

- Ing. **David Valouch**

- Ing. **Martin Zoula**

- Ing. **Jindřiška Deckerová**

Organizace a hodnocení **B0B36PRP** – Procedurální programování

- Rozsah: 2p+2c; Zakončení: Z,ZK; Kredity: 6; 1 ECTS kredit je 25–30 hodin za semestr, cca 180 h.
  - Kontaktní část (přednáška a cvičení): 3 hodiny týdně, tj. 42 hodin celkem.
  - Zkouška včetně přípravy: *10 hodin*.
  - Domácí příprava (úkoly) cca **9 hodin týdně**. *Medián zátěže*
- **Průběžná práce v semestru** – domácí úkoly a test.
- **Zkouškový test a implementační zkouška.** *Schopnost samostatné práce na počítačích v učebnách.*
- Docházka na **cvičení** a odevzdání domácích úloh. *Samostatná práce (kontrola plagiátů).*
  - Postupujte systematicky, budete tak postupně rozvíjete své schopnosti.
  - Využijte čas v prvních úlohách a naučte se psát programy správně.

  Program musí být nejen správný a funkční, ale také čitelný a udržovatelný!
- Konzultace – Pokud nevíte, tápete nebo řešíte domácí úkol příliš dlouho, **konzultujete** s cvičícím/přednášejícím. *Čtěte (učebnici), pochopte principy (nejen hledat řešení), hlídejte si čas a včas konzultujete!*
  - **Maximálně využijte kontaktní čas na cvičení/přednášce, ptejte se, diskutujete!**
- *„Alternativní"* absolvování předmětu pro velmi zkušené – **Seminář ACM** - předmět A4B36ACM!

Co je programování?

- Schopnost (samostatně) implementovat výpočetní řešení problému (algoritmem).
- Implementovat nejen funkční, ale správné řešení. *Programování vs. algoritmizace. Jen funkční (nebo jakože funkční) nestačí.*
- Programování (implementování) je aplikování známých vzorů řešení a způsobů zápisu programu v konkrétním programovacím jazyce (syntax a semantika jazyka).

  Programování je dovednost, řemeslo, která se získá praxí, zkoušením, implementováním.
- Základní dovednosti a postupy.
  1. **Dekompozice** - rozdělení problému na jednodušší části.

     Části mohou být znovupožitelné (zoobecňování).
  2. **Paměťový** (datový) model výpočtu, práce s pamětí.

     Jaký je životní cyklus paměti, jak data předáváme a přistupujeme k nim.
  3. **Výpočetní** model, práce s paměti (čtení, modifikace, změna - **přiřazení**) **posloupnost příkazů**, **větvení**, **cyklus**.
  4. **Správnost** - **kontrola správnosti vstupních dat**, reakce na nechtěný, ale možný uživatelský vstup, **kontrola úspěšného provedení kódu**.
  5. **Čitelnost**, **srozumitelnost** a **udržitelnost** kódu.

     Schopnost orientovat se v kódu, přehlednost pro sebe i v rámci týmu.

Cíle předmětu

- Osvojit si pohled na výpočetní prostředky jako *„počítačový vědec"* a naučit se je efektivně používat. *Computer scientist*
  - Formulovat problém a jeho řešení počítačovým programem.
  - Získat povědomí jaké problémy lze výpočetně řešit.
- Získat zkušenost s programováním *získání vlastní zkušenosti*
  - Programování v C

  cvičení, domácí úkoly, zkouška
- Osvojit si schopnost číst, psát a porozumět malým programům
- Získat programovací návyky jak psát
  - Srozumitelné a přehledné zdrojové kódy;
  - Opakovaně použitelné programy.

Výuka programování

„Separating Programming Sheep from Non-Programming Goats“

http://blog.codinghorror.com/separating-programming-sheep-from-non-programming-goats

http://www.eis.mdx.ac.uk/research/PhDArea/saeed/paper1.pdf

- Efektivní metody výuky programování se hledají již od dob prvních počítačů

  Déle než 50 let
- Přesto se zdá, že je každý základní kurz programování obtížný a 30 % až 60 % studentů jej na poprvé nezvládne.

  2023/2024: 63 % (92 % z udělených zápočtů), 85% ve stavu studuje

  2022/2023: 67 % (87 % z udělených zápočtů)

  2021/2022: 79 % (92 % z udělených zápočtů)

  2020/2021: 73 % (93 % z udělených zápočtů)

  2019/2020: 64 % (96 % z udělených zápočtů)
- **Základní koncept je pochopení principu přiřazení hodnoty proměnné!**

  Je to především práce s pamětí, která je v Cčku velmi přímá.

Princip přiřazení

- Zápis programu pro přiřazení hodnot do proměnných *a* a *b* a následné přiřazení proměnné *b* do *a*.

Přiřazení hodnoty proměnné

```
1 int a = 10;
2 int b = 20;
3
4 a = b;
```

- Jaké jsou hodnoty proměnných *a* a *b*?

a. a = 20, b = 0
b. a = 20, b = 20
c. a = 0, b = 10
d. a = 10, b = 10
e. a = 30, b = 20
f. a = 30, b = 0
g. a = 10, b = 30
h. a = 0, b = 30
i. a = 10, b = 20
j. a = 20, b = 10

Program vlastně „jen“ přesouvá a upravuje číselné hodnoty v paměti na základě definovaných podmínek!

Program je „recept“

- Program je „recept“ – posloupnost kroků (výpočtů) popisující průběh řešení problému.
- Programování je schopnost **samostatně**
  - **tvořit programy**;
  - **dekomponovat** úlohy na menší celky;
  - sestavovat z **dílčích částí větší programy** řešící komplexní úlohu.

B0B36PRP – je příležitostí, jak se těmto schopnostem naučit!

Zdroje a literatura

- **Knihy (učebnice)**

  Základní učební text „Programming in C“ (Kochan, 2014)

  Programming in C, 4th Edition, *Stephen G. Kochan*, Addison-Wesley, 2014.

  Recommended textbook.

  C Programming: A Modern Approach, 2nd Edition, *K. N. King*, W. W. Norton & Company, 2008.

  More like a reference manual, still comprehensive textbook.

  The C Programming Language, 2nd Edition (ANSI C), *Brian W. Kernighan, Dennis M. Ritchie*, Prentice Hall, 1988 *1st edition 1978*

- Přednášky – podpora učebního textu, slidy, videa, poznámky a **vlastní poznámky**.

  Součástí přednášek jsou také zdrojové kódy s příklady!
- Cvičení – získání praktických dovedností řešením domácích úkolů a dalších úloh.

  programovat, programovat, programovat

Další učebnice jazyka C

- Practical C Programming, *Steve Oualline*, O'Reilly Media, Inc., 3rd edition, 1997)

  Briefer than Kochan's textbook, still comprehensive.

- Effective C: An Introduction to Professional C Programming, *Robert C. Seacord*, William Pollock, 2020.

  Great if you already known some of C syntax and like to improve your skill further.

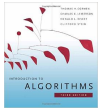
- Fluent C, Principles, Practices, and Patterns, *Christopher Preschern*, O'Reilly Media, Inc., 2022.

  Suitable if you like to know more about coding practices.

- 21st Century C: C Tips from the New School, *Ben Klemens*, O'Reilly Media, 2012.

Další zdroje

- Introduction to Algorithms, 3rd Edition, *Cormen, Leiserson, Rivest, and Stein*, The MIT Press, 2009, ISBN 978-0262033848



- Algorithms, 4th Edition, *Robert Sedgewick, Kevin Wayne*, Addison-Wesley, 2011, ISBN 978-0321573513

- The C++ Programming Language, 4th Edition (C++11), *Bjarne Stroustrup*, Addison-Wesley, 2013, ISBN 978-0321563842

Způsob výuky programování v B0B36PRP

- Naší snahou je vybudovat zkušenost a rozvinout dovednost programování.
  - Programování vs. algoritmizace;
  - Programování je „řemeslo“, jak správně implementovat nějaký algoritmus.
  - **Jen funkční nestačí - program musí být i správně!** *Očekávaný vstup vs. co všechno může uživatel na vstup zadat.*
- Studijní zátěž je proto rozložena do výukové části semestru.
  - Úkoly na cvičení a termíny domácích úkolů.
- Systematické rozvíjení dovednosti programování v průběhu semestru považujeme za zásadní.

  Typicky je na začátku semestru čas pro pochopní principů (čtení učebnice)!
- Bez znalosti konstruktů a základní příkazů nelze efektivně programovat.

  Nezávislost na našeptávači!
  - Začínáme relativně jednoduchými úlohami k osvojení programovacích konstruktů a způsobu organizace zdrojového kódu. *Přehledný kód a schopnost se efektivně orientovat v kódu je zásadní!*
  - *Úkoly jdou vždy realizovat s tím, co si řekneme na přednášce/cvičení.*

    Řešení s pokročilejšími konstrukty může být elegantnější(kratší), nemusí však dodat potřebný vhled.
  - V prvních přednáškách pokrýváme nezbytné znalosti, které jsou dále prohlubovány.
    - Cvičení dopňují přednášky a dávají více prostoru pro praktické osvojení problematiky.
- Můžete volit praktický způsob vstřebávání znalosti programování z příkladů, který je vhodný doplnit teoretickou přípravou z učebnic(e).

Přednášky – zimní semestr (ZS) akademického roku 2024/2025

- Harmonogram akademického roku 2024/2025.

  `https://www.fel.cvut.cz/cz/education/harmonogram2425.html`
- Přednášky
  - Karlovo náměstí, místnost KN:E-107, čtvrtek, 12:45–14:15.
- 14 výukových týdnů. *13+1 přednášek*
- Státní svátek – 28.10.2024 (pondělí).
- Děkanský den (den bez výuky) – 29.10.2024 (úterý).
- Konzultace: čtvrtek 14:30–16:00 - KN:E-205. *Po přednášce, případně po domluvě.*

Přehled přednášek

- 01 - Informace o předmětu, Úvod do programování — *S. G. Kochan: kapitoly 1–3*
- 02 - Programování v C — *S. G. Kochan: kapitoly 2–5 a část 6*
- 03 - Řídicí struktury, výrazy a funkce — *S. G. Kochan: kapitoly 4–6 a 12*
- 04 - Pole, ukazatel, textový řetězec, vstup a výstup programu — *S. G. Kochan: kapitoly 7, 10 a 11*
- 05 - Ukazatele, paměťové třídy a volání funkcí — *S. G. Kochan: kapitoly 8 a 11*
- 06 - Struktury a uniony, přesnost výpočtů a vnitřní reprezentace číselných typů — *S. G. Kochan: kapitoly 9, 14, 17 a Appendix B*
- 07 - Standardní knihovny C. Rekurze. (Základní vlastnosti jazyka C probrány.)
- 08 - Spojové struktury — *S. G. Kochan: kapitola 16 a Appendix B*
- 09 - Abstraktní datový typ (ADT) - zásobník, fronta, prioritní fronta
- 10 - Stromy
- 11 - Prioritní fronta, halda. Příklad použití při hledání nejkratší cesty v grafu
- 12 - Přesnost a rychlost výpočtu
- 13 - C++ konstrukty v příkladech / Paralelní programování (vlákna)
- 14 - Rezerva – *Dotazy, informace ke zkoušce*

Přednáška nemusí být prezentace slidů – je očekávána interakce, řešení dotazů a diskuse problematických a náročnějších částí.

Podklady k přednášce jsou k dispozici před přednáškou podobně jako **učebnice**.

Domácí úkoly a další úlohy

- Samostatná práce s cílem osvojit si praktické zkušenosti.
- Jednotné zadání na přednášce a jednotný termín odevzdání.
- Odevzdání domácích úkolů prostřednictvím BRUTE.

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

  - Nahrání (upload) archivu s nezbytnými zdrojovými soubory.
  - Ověření správnosti implementace automatickými testy.
  - Penalizace za překročení počtu uploadů.

Odevzdávejte funkční kódy, nikoliv „pouze“ kódy, které projdou testy!

  - Detekce plagiátů

*Cílem řešení úkolů je **získat vlastní zkušenost**!*

- Úkoly jsou navrhovány tak, aby byly stihnutelné. Plánujte a hlídejte si čas, včas konzultujte.
- Klíčem k úspěšnému dokončení předmětu je samostatná práce a osvojení si technik a znalostí

Průběžná práce a řešení úkolů!

- Pokud něčemu nerozumíte, **ptejte se**!

Pokud chybujete, tak se učíte, pokud nechybujete, tak už to umíte!

Přehled domácích úkolů

- Domácí úkoly s povinným, volitelným, případně bonusovým zadáním. *47 h, bonus +23 h*

`https://cw.fel.cvut.cz/wiki/courses/b0b36prp/hw/start`

0. HW 00 - První program — *1 h*
1. HW 01 - Načítání vstupu, výpočet a výstup — *1 h*

   Seznámení se s prostředím, psaním programů, jejich laděním, testováním a odevzdáváním. ∼ 20–40 h
2. HW 02 - První cyklus (**Kontrola přehlednosti kódu** – až -100% z dosažených bodů) — *2 h*
3. HW 03 - Kreslení (ASCII art) (**Kontrola kódu** – až -100%) — *3 h*
4. HW 04 - Prvočíselný rozklad (**Kontrola kódu** – až -100%) — *5 h, bonus +8 h*
5. HW 05 - Caesarova šifra (**Kontrola kódu** – až -100%) — *6 h*
6. HW 06 - Maticové počty (**Kontrola kódu** – až -100%) — *7 h bonus +7 h*
7. HW 07 - Hledání textu v souborech — *5 h*
8. HW 08 - Kruhová fronta v poli - *Dynamicky linkovaná knihovna* — *5 h*
9. HW 09 - Načítání a ukládání grafu — *5 h*
10. HW 10 - **Integrace** načítání grafu a prioritní fronta v úloze hledání nejkratších cest

    HW 09 + 12. přednáška, soutěž na extra body — *8 h bonus +8 h*

- Podmínkou zápočtu je úspěšné odevzdání všech domácích úkolů.
- Odevzdání volitelného zadání je doporučeno (není částečné odevzdání).

Celkové body za povinné zadání **25b**, volitelné zadání **20b**, bonusové **10b**$^{+}$.

Odevzdávání domácích úkolů – BRUTE

- **BRUTE** – Bundle for Reservation, Uploading, Testing and Evaluation.
  - Formální kontrola – kompilace programu.
  - Testování funkčnost 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.
    - Vytvoření vlastních vstupů a laděním programu.
    - Vytvoření vstupů **přiloženým generátorem vstupů**.
    - Ověření výstupu **přiloženým testovacím nebo referenčním programem**.
- Porozumění kódu a kontrola možných stavů.
  - **Pro každý řádek byste měli být schopni odpovědět proč tam je a co dělá!**
  - 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.

Úkoly a **BRUTE**

- Úkoly nejsou jen o odevzdání implementace, která projde BRUTE testy.
  - Cíl není odevzdat úkoly v BRUTE, je to prostředek ověření funkčnosti programu.
  - BRUTE je nástroj průběžné kontroly postupu a získávání znalostí.
  - Cíl je naučit se **samostatně programovat** funkční programy správně.
- Úkoly jsou především o **postupném získání zkušeností** s konkrétními konstrukty.
- Úkoly mají relativní obtížnost velmi podobnou.
  - Je důležité postupně samostatně řešit jednotlivé úkoly a osvojovat si dílčích dovednosti.

  Absolutně jsou úlohy postupně náročnější a náročnější!
- Netrapte se s řešením příliš dlouho sami, ptejte se na fóru, cvičících, přednášce, **konzultaci**.
- Úlohy HW02–HW06 budou manuálně kontrolovány na správnost, přehlednost kódu.
  - Zaměřeno na konzistenci, čitelnost, a **modularitu** (rozdělní do funkcí).
  - *Motivace je netrávit příliš mnoho času implementací bez výrazného postupu.*

Hodnocení předmětu

Zdroj bodů	Maximum bodů	Přípustné minimum bodů
Domácí úkoly	45	**35**
Bonusové úkoly	10^+	
Test v semestru	5	
Písemný zkouškový test	20	†10
Implementační zkouška	20	**10**
Součet	100^+ bodů	

†V případě neúspěšné implementace, lze opakovat pouze implementační část zkoušky byl test hodnocen 13 více body.

- **Zápočet**: alespoň 35 bodů a odevzdáné všechny domácí úkoly **do 13.1.2024@23:59 CET!**
- Předmět lze úspěšné ukončit **zápočtem** a **zkouškou**.

  Zkouškové termíny (KOS) a `https://cw.fel.cvut.cz/b231/courses/b0b36prp/start`.
- Test a písemná zkouška – krátké stručné odpovědi prokazující porozumění problematice.

  `https://cw.fel.cvut.cz/wiki/courses/b0b36prp/resources/test`
- Implementační zkouška – prokázaní samostatně porozumět a napsat krátký program.

  `https://cw.fel.cvut.cz/wiki/courses/b0b36prp/resources/exam`

Klasifikace předmětu

Klasifikace	Bodové rozmezí	Hodnocení	Slovní hodnocení
A	≥ 90	1	výborně
B	80–89	1,5	velmi dobře
C	70–79	2	dobře
D	60–69	2,5	uspokojivě
E	50–59	3	dostatečně
F	<50	4	nedostatečně

- Včasné odevzdáním všech domácích úkolů s povinným a **volitelným** zadáním (45 bodů).
- Bonusová úloha a bonusové odevzdání HW10 (10^+ bodů).
- Test v semestru (5 bodů).
- Písemná zkouška (20 bodů) .

  15 a více bodů je velmi slušný výsledek!
- Implementační zkouška (20 bodů).
- **95 bodů** a více (A – výborně), **76 bodů** (C – dobře) – (20% ztráta).
- Body jsou indikátorem průběžných výsledků.

Komunikace související s PRP

- Obracejte se na svého cvičícího dle cvičení, na které chodíte (jste přihlášeni).

  Případně na **prp-teachers@fel.cvut.cz**
- Komunikovat můžete elektronickou poštou (e-mail).
  - Pište ze své fakultní adresy (odesílatel).
  - Do předmětu zprávy uvádějte zkratku předmětu PRP.
  - Kopii zprávy (Cc) posílejte též příslušnému vedoucímu cvičení (dle studijního programu).
  - V případě zásadních problému (např. týkajících se zápočtu) uvádějte do Cc též přednášejícího.
- Případně můžete využít **discord** kanálu.

  Time management - nečekejte okamžitou odpověď.

  Pracujte soustředěně a užijte si tvůrčí zápal.

Vývojové prostředí

- Počítačové laboratoře - Ubuntu se síťovým bootováním a domovskými adresáři (NFS v4).
  Přenos a synchronizace souborů – ownCloud, SSH, gdrive, sharepoint
- Doporučený operační systém - Ubuntu-based, Pop OS!, Win s WSL(2) ideální s nativní VS Code.
  Přímočará instalace potřebných programů.
- Překladače **gcc** a **clang**, sestavení **make** (GNU make). https://gcc.gnu.org a http://clang.llvm.org
- **Visual Studio Code** (VSC) - https://code.visualstudio.com/
- Editor – **gedit**, atom, **sublime**, **vim** – https://atom.io/, http://www.sublimetext.com/
  http://www.root.cz/clanky/textovy-editor-vim-jako-ide
  Pokud programovat umíte, investuje čas do efektivního ovládání editoru, např. **vim**.
- C/C++ vývojová prostředí – WARNING: Do Not Rely on an IDE.
  http://c.learncodethehardway.org/book/ex0.html
  - **CLion**, NetBeans 8.0 (C/C++), Eclipse-CDT – https://www.jetbrains.com/clion
  - Geany, Code::Blocks, CodeLite https://www.geany.org/, http://www.codeblocks.org, http://codelite.org
  - **Nejdříve porozumějte principům**, nakonfigurujte nástroje a programování zefektivněte.
- Odevzdávání domácích úkolů BRUTE https://cw.felk.cvut.cz/brute.
  BRUTE – Bundle for Reservation, Uploading, Testing and Evaluation.

Služby akademické sítě – FEL, ČVUT

- http://www.fel.cvut.cz/cz/user-info/index.html
- Diskové úložiště ownCloud – https://owncloud.cesnet.cz
- Zasílání velkých souborů – https://filesender.cesnet.cz
- Rozvrh a termíny – FEL Portal – https://portal.fel.cvut.cz
- FEL Google Account – autentizovaný přístup do Google Apps for Education
  Více viz http://google-apps.fel.cvut.cz/.
- Gitlab FEL – https://gitlab.fel.cvut.cz/
- Přístup k informačním zdrojům (IEEE Xplore, ACM, Science Direct, Springer Link).
- Akademické a kampusové licence. https://download.cvut.cz
- Národní Gridová Infrastruktura MetaCentrum. http://www.metacentrum.cz/cs/index.html
- RCI Cluster. https://login.rci.cvut.cz