

# IB111 Základy programování

## Úvod kurzu

Radek Pelánek

2018

# Úvodní dotazník

- 1 programátorské zkušenosti
  - A (téměř) žádné
  - B dílčí zkušenosti, ale nepříliš kvalitní
  - C dobré zkušenosti
- 2 programovací jazyk
  - Python
  - C, C++, C#
  - Java
  - Pascal
  - PHP
  - JavaScript
  - jiné

# Dnešní přednáška

- o předmětu
- organizace, ukončení
- pojmy – algoritmus, programování
- motivace, širší kontext
- představení Pythonu, rychlé demo

více o samotném programování až příště

„dobré **základy**“

- zvládnutí základních **programátorských konstrukcí** (proměnné, funkce, if, for, while, ...)
- obecné **principy** použitelné v řadě programovacích jazyků
- programátorský **styl**
- úvod do programátorského a algoritmického **stylu myšlení**

# Programovací jazyk Python

„Základy programování“

nikoliv

„Programování v Pythonu“

- Python je používán pro ilustraci pojmů a příkladů, na cvičeních
- důraz na obecné koncepty, cílem není detailní zvládnutí Pythonu
- záměrně **ne**probíráme specifika Pythonu
- zvládnutí konkrétního jazyka – trénink a praxe

# Programátorská kultura

Programy by měly být nejen korektní, ale i „pěkné“.

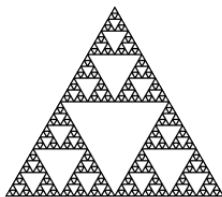
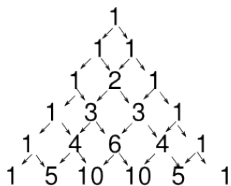
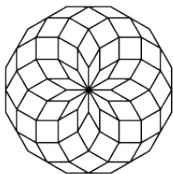
Programy by měly být nejen korektní, ale i „pěkné“.

- názvy proměnných, funkcí
- rozdělení funkcionality do funkcí
- zarovnání řádku
- dokumentace, komentáře
- nepoužívání „copy&paste“ kódu
- ...

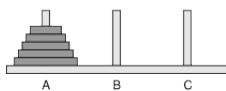
# Co čekat: příklady

- **výpočty**: faktoriál, prvočísla, odmocnina, náhodná čísla
- **obrázky**: želví grafika, „textová grafika“, bitmapy
- **jednoduché hry**: hádání čísla, variace na piškvorky
- **zpracování dat**: statistiky dat ze souboru

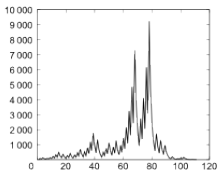




P J U Y B U  
 O D E O A R B A L O H E  
 K A S H N U N K I K R Z  
 L C Y I Z A



A -> C; A -> B; C -> B;  
 A -> C; B -> A; B -> C;  
 A -> C



# Forma předmětu

- 2h přednáška, nepovinná, ale velmi doporučená
  - hodně „příkladový styl“
  - slidy nemusí být pochopitelné bez komentáře, obsahují i záměrné chyby (pro ilustraci)
  - záznamy dostupné, ale plátno nepříliš čitelné
- 2h cvičení, povinné
  - programování v jazyce Python
  - více skupin, cvičících

Pozn. povinnost „být připraven na cvičení“

# Přednáška: zasedací pořádek

- D1 plná pozornost po celou dobu, žádné rušivé aktivity na notebooku
- D3 pozdější příchod, občas nesouvisející aktivita na notebooku a podobně
- D2 volnější sledování přednášky

řešení problémů, nestandardní situace, dotazy:

- cvičící spíše než přednášející
- osobně (po cvičení, přednášce) spíše než elektronicky
- diskuzní fórum spíše než mail

# Hodnocení předmětu

## 500 bodů

- **domácí úkoly:** 160 bodů
  - 5 za semestr
  - bodování: 25, 25, 30, 40, 40
- **2 vnitrosemestrální zkoušky u počítače:** 40+100 bodů
  - programování v rámci jednoho cvičení
  - úkoly – variace na příklady ze cvičení
- **závěrečná písemná zkouška:** 200 bodů
  - zkouší se principy, algoritmy, pojmy, „čtení kódu“
  - test s výběrem možností, podobné průběžným  
Odpovědníkům, Kahootům
- výjimečné bonusové body (kladné i záporné)

# Hodnocení předmětu

minimum pro ukončení:

- účast na cvičení (max. 2 neomluvené hodiny)
- odevzdání všech 5 domácích úloh (alespoň 5 bodů za každou)
- alespoň 100 bodů za domácí úlohy
- alespoň 70 bodů za vnitrosemestrální zkoušky
- alespoň 110 bodů ze závěrečné zkoušky

známka: hranice rovnoměrně mezi 280 a 500

# Nesplnění podmínek I

Nesplnění podmínek na účast na cvičení či nedostatečný zisk bodů z domácích úloh:

- nemáte nárok na opravu
- odůvodněné případy – bonusový (a náročný) domácí úkol
- plně v kompetenci cvičícího

# Nesplnění podmínek II

Nedostatečný zisk bodů z vnitrosemestrálních zkoušek:

- opravná zkouška v lednu
- binární hodnocení:
  - úspěch  $\Rightarrow$  součet bodů z vnitrosemestrálek se přepíše na konstantu 70
  - neúspěch  $\Rightarrow$  hodnocení F bez možnosti další opravy



# Nesplnění podmínek III

Nedostatečný zisk bodů ze závěrečné zkoušky:

- standardní závěrečná zkouška
- hodnocení F, můžete jít na opravný termín

# Domácí úkoly – organizace

- skupiny: rámcově stejné zadání, odlišné detaily
- přesné zadání, termín odevzdání – kompetence cvičících

# Opisování

- pracujte **samostatně**
- opisování se trestá velmi přísně (hodnocení F)
- neřešíme, kdo opisoval – **nesdílejte svoje řešení**
  - dát někomu opsat řešení je „danajský dar“
  - kdo nezvládne tento předmět samostatně, ztroskotá téměř jistě v dalším studiu

# Prohlášení

Autor: Jméno Příjmení, UČO

Prohlašuji, že celý zdrojový kód jsem zpracoval(a) zcela samostatně. Jsem si vědom(a), že nepravdivost tohoto tvrzení může být důvodem k hodnocení F v předmětu IB111 a k disciplinárnímu řízení.

# Prohlášení: varianta s asistencí

Autor: Jméno Příjmení, UČO

Prohlašuji, že zdrojový kód jsem zpracoval(a) samostatně, **kromě částí, které jsou explicitně označeny ASISTENCE a doprovozeny vysvětlujícím komentářem.** Jsem si vědom(a), že v nepravdivost tohoto tvrzení může být důvodem k hodnocení F v předmětu IB111 a k disciplinárnímu řízení.

# Domácí úlohy: Sebehodnocení

Součástí každé domácí úlohy je sebehodnocení:

- známé nedostatky
- programátorský styl

<https://www.fi.muni.cz/IB111/>

- harmonogram přednášek, cvičení, úkolů
- rozepsané podmínky ukončení
- výukové materiály
- doplňující informace

# Studijní materiály v ISu

Relevantní agendy z ISu pro tento předmět:

- *Učební materiály* – slidy z přednášek
- *Organizační pokyny* – archiv zaslaných mailů
- *Odpovědníky* – tréninkové testy
- *Odevzdáárny* – odevzdávání domácích úloh
- *Poznámkové bloky* – počet bodů z úloh
- *Diskuse* – nejasnosti, tipy na zajímavé zdroje, ...



# Sbírka příkladů

<https://www.fi.muni.cz/IB111/sbirka/>

- interaktivní webová stránka
- příklady ze cvičení
- procvičení nad rámec cvičení

# Odpovědníky, kontrolní otázky

- Odpovědníky

- objeví se v ISu v průběhu semestru
- otázky s výběrem možností
- dobrovolné, doporučené

- Kontrolní otázky

- otevřené otázky ke každé přednášce
- průběžně aktualizovaný dokument:

<https://docs.google.com/document/d/19VeL15P5s8rv-YoCMwpIiQD34ptn6bevJKsV7mBD-Uo>

- očekává se zvládnutí **před cvičením**
- nepřipravenost může znamenat i záporné body

# Umíme programovat

- `https://www.umimeprogramovat.cz/`
- neomezený přístup po přihlášení, viz instrukce v mailu
- především cvičení *Programování v Pythonu*

# Vybrané doplňkové zdroje – knihy

- *Python Programming: An Introduction to Computer Science*, J. M. Zelle.
- *Introduction to Computing and Programming in Python, A Multimedia Approach*. M. Guzdial, B. Ericson.
- *Programátorská cvičebnice*, R. Pelánek.
- *Jak to vyřešit*, R. Pelánek.

## Vybrané doplňkové zdroje – web

- **Učíme se programovat v jazyce Python**,  
<http://howto.py.cz/index.htm>
- <http://interactivepython.org> – interaktivní učebnice
- dokumentace k Pythonu
- <https://www.hackerrank.com/> – příklady, řešení (a opravování) v prohlížeči
- Coursera, Udacity kurzy
- sdílejte užitečné zdroje v diskusním fóru předmětu

# Předpoklady

- základní počítačová gramotnost
- středoškolská matematika (např. faktoriál, prvočíslo, logaritmus)
- logické spojky (and, or, ...)
- angličtina (alespoň pasivně, základní porozumění)

rychltest: Kahoot

# Náročnost předmětu

... závisí na vstupních dovednostech

**A žádné** programátorské zkušenosti

⇒ **náročné**, nezbytné věnovat průběžně čas nad rámec přednášek a cvičení (doporučeno: fixní 2 hodiny týdně v rozvrhu)

**B dílčí** programátorské zkušenosti

⇒ není těžké, pokud se předmětu **průběžně** poctivě věnujete

**C dobré** programátorské zkušenosti

⇒ celkem snadné, ale **nepodcenit** (především 2. polovinu kurzu a závěrečnou zkoušku)

# Motivační úloha

- převozník, loďka uveze jen 1 další kus nákladu
- náklad: vlk, koza, zelí
- bez dozoru:
  - vlk žere kozu
  - koza žere zelí
- jak dostat vše bezpečně na druhou stranu





# Motivační úloha

- převozník, loďka uveze jen 1 další kus nákladu
- náklad: vlk, koza, zelí
- bez dozoru:
  - vlk žere kozu
  - koza žere zelí
- jak dostat vše bezpečně na druhou stranu



Jak řešit úlohu algoritmicky? Co to znamená?

<http://www.fi.muni.cz/~xpelanek/IB111/vkz/>

- návod/postup, jak „mechanicky“ vyřešit určitý typ úlohy/problému
- příklady:
  - rozklad na součin prvočísel
  - nalezení nejkratší cesty mezi dvěma městy
  - vygenerovat zadání Sudoku

# Žádoucí vlastnosti algoritmu

- má jasný vstup a výstup
- obecný (nejen pro omezenou třídu instancí)
- deterministický (vždy jednoznačné, jak postupovat)
- konečný, efektivní

# Programování

- za **algoritmus** můžeme považovat i recept, návod
- **programování** – zápis algoritmů pro počítače
- počítače jsou „hloupé“ – zápis algoritmu musí být **opravdu přesný** (srovnej „osolíme přiměřeně“)
- nutnost vyjadřovat se přesně:
  - otrava – náročný zápis
  - bonus – nutnost myslet přesně

# Programování: motivace

Proč pořádně zvládnout základy programování?

- základ pro další studium
- užitečnost
  - profesní
  - občasná
- elegance, kreativita, „síla“

# Programování: způsoby využití

(příklady, rozhodně ne kompletní klasifikace)

- aplikace
- programování pro web
- vestavěné systémy
- vědecké výpočty
- skriptování

*každé důraz na něco jiného, sdílí ale základní principy  
„informatického myšlení“, námi probírané základní konstrukce  
jsou potřeba všude*

- „samostatné“ aplikace pro stolní počítače, mobilní zařízení
- příklady:
  - kancelářský software
  - editace grafiky, zvuku, videa
  - hry
- důraz na interakci s uživatelem
- využití knihoven, práce s operačním systémem

# Programování pro web

- příklady:
  - informační systémy
  - e-obchody
  - prezentace firmy
- široká škála:
  - drobné úpravy existujících systémů (CMS)
  - vytváření vlastních rozsáhlých systémů
- práce s databázemi, integrace různých prostředků (Python/PHP, JavaScript, CSS, HTML. . .)
- důraz na soukromí – přístupová práva v IS, elektronické platby



- příklady:
  - kuchyňské spotřebiče, GPS, mobil, foťák
  - dopravní prostředky
  - zdravotnické přístroje
- nízko-úrovňové programování, ovladače
- úzké propojení s konkrétním hardwarem
- bezpečnost, práce s limitovanými zdroji (paměť, energie)

- příklady:
  - simulace počasí, klimatu
  - bioinformatika (protein folding, analýza genomu, ...)
- vymýšlení algoritmů (urychlení výpočtu, distribuované výpočty)
- propojení informatiky a matematiky (příp. jiných disciplín)
- zpracování rozsáhlých dat
- uživatelské rozhraní a interaktivita jsou jen malá část

- příklady:
  - převod dat mezi různými formáty
  - rychlá analýza dat
  - prototypy, experimenty
  - drobné úpravy systému (např. správce sítě)
- malý rozsah, specifický účel
- často jednorázové aplikace
- „programování pro běžný život“

# Programování v malém / ve velkém

- programování v malém
  - desítky až stovky řádků kódu
  - nezávislé na „ostatních“
  - tento předmět
- programování ve velkém
  - tisíce až milióny řádků
  - závislosti, souvislosti, návrh, testování, ...
  - další předměty (OOP, softwarové inženýrství, ...)

# Programování v malém: motivace

- nutná prerekvizita pro kvalitní programování ve velkém
- trénink myšlení
- prakticky užitečné, i když nejste programátor na plný úvazek

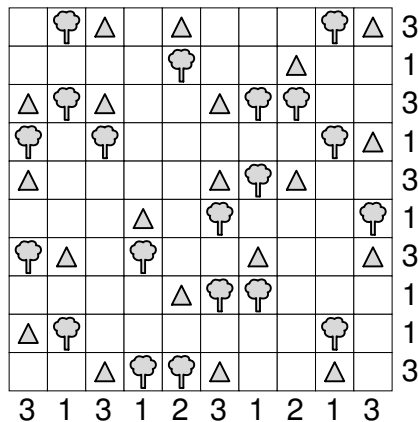
# Praktické programování v malém: příklady

- tvorba studijního katalogu
- vytvoření interaktivní výukové úlohy pro webový systém
- zpracování botanických dat
- obrázky do knihy Hlavolamikon
- vytváření šifer pro Tmou

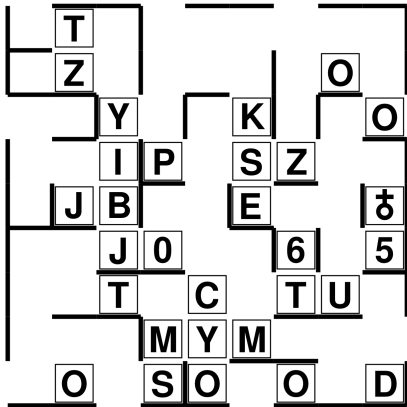
*Pokud neovládáte X, pak vás často ani nenapadne, že by se vám X mohlo hodit.*

# Obrázky do knihy

"0102010902050302030703080401040304090507060606100701...;  
3131313113:3131231213-;10"



**TMOLX** 7



**TMOLX** 7

**TMOLX** 7

**TMOLX** 7



# Doporučené cvičení

rutinní činnost (na počítači)



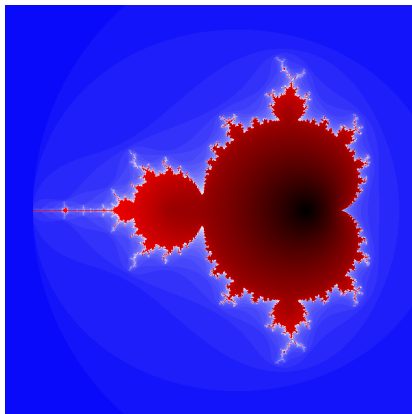
programátorské cvičení

# Nejen užitečnost. . .

programování je zajímavé i samo o sobě

- elegantní myšlenky
- radost z objevování, experimentování
- tvoření, kreativita
- „síla“ – pár stisků klávesnice a vytvoříte něco nového a zajímavého

# Elegance

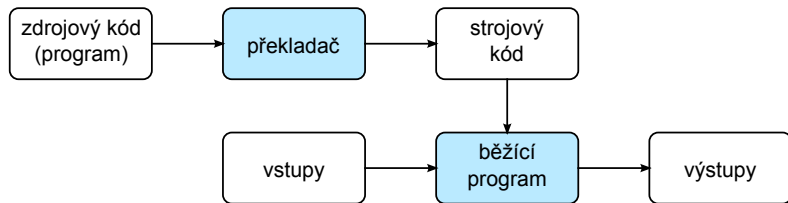


Mandelbrotova množina, 25 řádků kódu

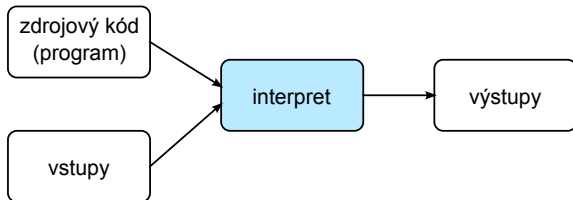
Zkuste YouTube: Mandelbrot set

# Programovací jazyky

## kompilovaný program



## interpretovaný program



# Programovací jazyky: klasifikace I

## nízko-úrovňové

- kompilované
- nutnost řešit specifika konkrétního systému
- explicitní práce s pamětí
- náročnější vývoj (nízká efektivita práce)
- vysoká efektivita programu

## vysoko-úrovňové

- interpretované
- nezávislé na konkrétním systému
- využití abstraktních datových typů
- snadnější vývoj (vysoká efektivita práce)
- nižší efektivita programu

nikoliv dvě kategorie, ale plynulý přechod; zjednodušeno

# Programovací jazyky: klasifikace II

zjednodušená klasifikace a použití

nízko-úrovňové C, FORTRAN, ...

vestavěné systémy, rychlé výpočty

objektové C++, Java, C#, ...

klasické aplikace, rozsáhlé systémy

skriptovací Python, PHP, JavaScript, Perl, ...

programování pro web, skriptování, prototypy

deklarativní Prolog, LISP, Haskell, ...

umělá inteligence

*více na samostatné přednášce na konci semestru*

- **vysoko-úrovňový** – velká míra abstrakce, „spustitelný pseudokód“
- **interpretovaný** – pomalejší než kompilovaný, ale větší volnost
- **pedagogický** – byl tak navržen, dnes již dominantní výukový jazyk
- **moderní a široce používaný** – patří mezi přibližně 5 nejpoužívanějších jazyků
- volně a snadno **dostupný** na všech platformách
- široká nabídka **knihoven**

# Původ jazyka a názvu

autor:

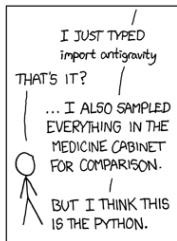
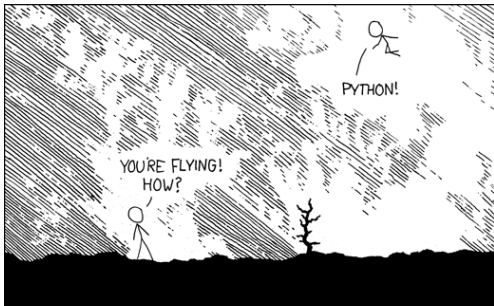
Guido van Rossum  
konec 80. let



název podle:  
Monty Python's  
Flying Circus





























# Python je používaný...

So what are the Top Ten Languages of 2018, as ranked for the typical IEEE member and *Spectrum* reader?

Language Rank	Types	Spectrum Ranking
1. Python	  	100.0
2. C++	  	99.7
3. Java	  	97.5
4. C	  	96.7
5. C#	  	89.4
6. PHP		84.9
7. R		82.9
8. JavaScript	 	82.6
9. Go	 	76.4
10. Assembly		74.1

<https://spectrum.ieee.org/at-work/innovation/the-2018-top-programming-languages>

# Programování v tomto kurzu

- důraz na obecné principy, nikoliv specifika Pythonu
- většina konceptů snadno a velmi podobně realizovatelná v jiných jazycích
- používáme Python 3 (zpětně nekompatibilní s Python 2)
- minimální použití rozšiřujících knihoven

# Osnova I: Základy

- Základní konstrukce (proměnné, výrazy, řídicí struktury, funkce)
- Programy pracující s čísly (číselné typy, jednoduché ukázky, dělitelnost, náhoda)
- Řetězce a seznamy (a kryptografické odbočky)
- Vyhledávání a řazení (práce se seznamem, binární vyhledávání, řadicí algoritmy, základy složitosti)

## Osnova II: Důležité věci detailněji

- Datové typy a jejich užití (seznam, zásobník, fronta, slovník, množina)
- Proměnné, paměť, soubory
- Rekurze
- Složené datové typy, objekty v Pythonu
- Práce s textem

# Osnova III: Aplikace, praxe

- Příklady aplikace datových struktur, práce s textem
- Obrázky (reprezentace, generování, úpravy)
- Vývoj programů (dokumentace, testování, moduly, konvence)
- Programovací jazyky (přehled jazyků a jejich užití), praktické postřehy

# Hlavní návaznosti

- IB002 Algoritmy a datové struktury I
- PB071 Principy nízkoúrovňového programování
- IB015 Neimperativní programování
- PB161 Programování v jazyce C++
- PB162 Programování v jazyce Java
- PB007 Softwarové inženýrství I

# Demo základních prvků

- proměnné
- typy (číslo, řetězec, bool)
- print (funkce pro výpis)
- výrazy, operátory (aritmetické, logické)
- podmínky (if/else)
- cykly (for, while)
- bloky kódu
- funkce



# Shrnutí: IB111 motivace I

- programování je zajímavé a zábavné
- důkladné základy jsou potřeba pro další studium
- Python je super (a přímo prakticky použitelný)

# Shrnutí: IB111 motivace II

Kdo nedá IB111, na FI (brzy) končí.

(trochu zjednodušeno, ale ne moc)