

# Programování a algoritmizace: úvod

2010

# Dnešní přednáška

- o předmětu, administrativa
- motivace

# Cíle předmětu

- Úvod do programátorského a algoritmického **stylu myšlení**
- Obecné **principy** použitelné v řadě programovacích jazyků
- Základní algoritmy a datové struktury
- Přehled programovacích jazyků a jejich výhod/nevýchod

# Forma předmětu

- 2h přednáška, nepovinné, ale velmi doporučené:
  - slidy nemusí být pochopitelné bez komentáře
  - na cvičení často algoritmy z přednášky
- 2h cvičení, povinné
  - programování v jazyce Python
  - důraz na algoritmy, nikoliv na prog. jazyk
- domácí úlohy – náročnost závisí na předchozích zkušenostech

# Ukončení předmětu

- **zkouška**
  - písemná, zkouší se principy, algoritmy, pojmy
  - nikoliv programování na počítači
  - alespoň 50% bodů
- bodované **úkoly** ze cvičení
  - alespoň 75% bodů
- účast na cvičení (max. 2 neomluvené hodiny)
- známka – podle součtu bodů ze zkoušky a úkolů ze cvičení

# Zdroje pro studium

- slidy z přednášek (budou v ISu)
- knihy, např.:
  - Introduction to the design and analysis of algorithms, A. Levitin.
  - Algoritmy (Datové struktury a programovací techniky), P. Wróblewski.
  - Python Programming: An Introduction to Computer Science, J. M. Zelle.
- internetové zdroje, např.:
  - Wikipedia články k probíraným tématům
  - dokumentace k Pythonu
  - MIT Open Courseware: Introduction to Computer Science and Programming (videa z přednášek)

# 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

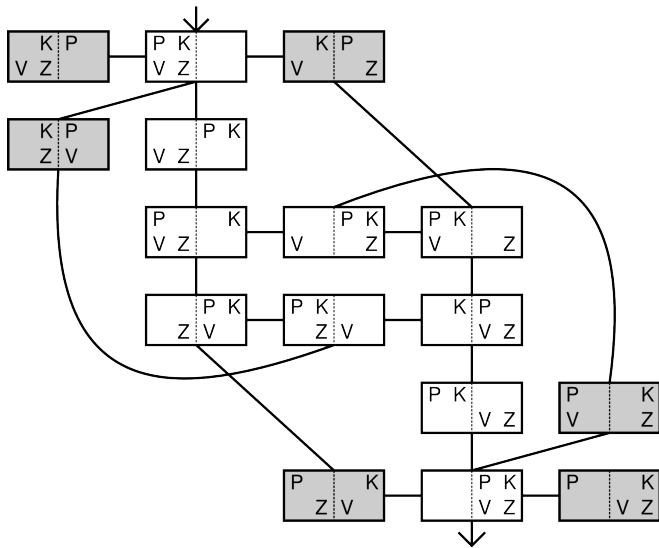
- převozník, loďka uveze jen 1 další kus nákladu
- náklad: vlk, koza, zelí
- bez dozoru:
  - vlk žere kozu
  - koza žere zelí
- jak dostat vše bezpečně na druhou stranu



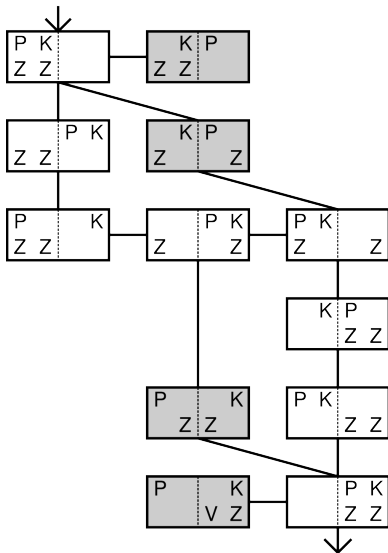
Jak řešit úlohy algoritmicky? Co to znamená?



# Vlk, koza a zelí



## Koza a dvě zelí



- návod/postup jak „mechanicky“ vyřešit určitý typ úlohy/problému
- typické 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)
- konečný
- deterministický (vždy jednoznačné, jak postupovat)

# 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: způsoby využití

Zkuste vymyslet (vzpomenout si) na co nejrozmanitější způsoby využití programování.

# Programování: způsoby využití

- „klasické“ aplikace
- programování pro web
- vestavěné systémy
- vědecké výpočty
- skriptování

*každé důraz na něco jiného, sdílí ale základní principy  
„informatického myšlení“*

# „Klasické“ aplikace

- příklady:
  - kancelářský, účetní software
  - editace grafiky, zvuku, videa
  - hry
- rozsáhlé projekty
- 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 (php, javascript, css, html, cgi, cookies, ...)
- důraz na soukromí – přístupová práva v IS, elektronické platby

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

# Vědecké výpočty

- příklady:
  - simulace počasí, klimatu
  - bioinformatika (protein folding, analýza genomu, ...)
- vymýšlení algoritmů (urychlení výpočtu, distribuované výpočty)
- 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

# Ilustrační příklad: Tmou

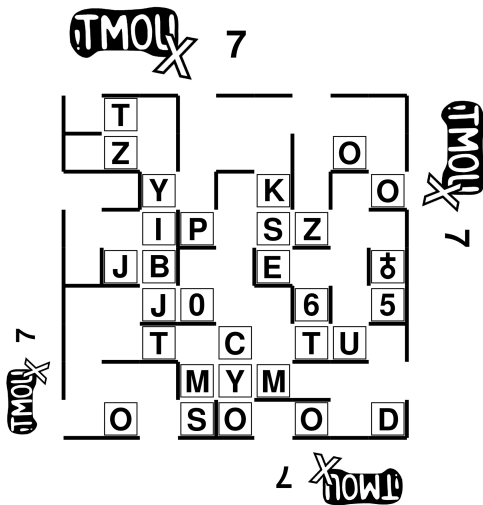
Ukázka z „praxe“:

- kde všude se může hodit programování (a inženýrské myšlení)
- programování se hodí, i když nejste programátor na plný úvazek

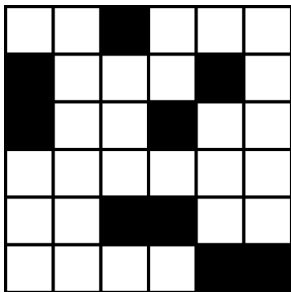
Tmou:

- šifrovací hra pro dospělé
- šifry často sofistikované
- konstrukce šifer občas náročná  $\Rightarrow$  využití programování

# Rotující gravitační bludiště

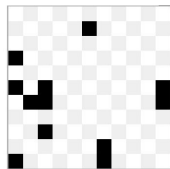


# Šifrovací mřížka

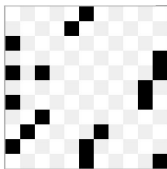


LEUCII  
KFSIAR  
ZMOKVR  
AIFZNR  
KOAZYV  
XIDAAS

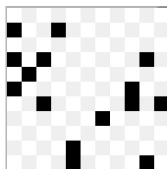
# Vývojová stádia šifrovací mřížky



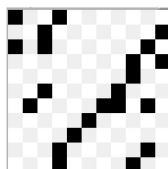
1



2

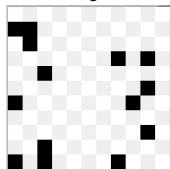


3

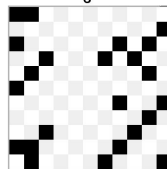


4

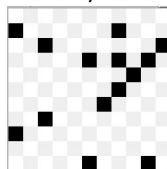
5



6



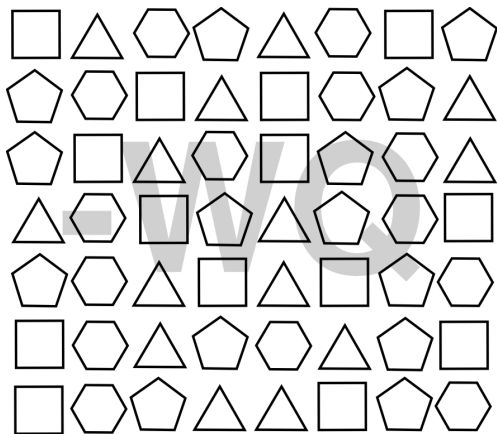
7



??



# Permutace tvarů



# Programovací jazyky: klasifikace I

## nízko-úrovňové

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

## vysoko-úrovňové

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

nikoliv dvě kategorie, ale plynulý přechod

# Programovací jazyky: klasifikace II

**zjednodušená** klasifikace a použití

nízko-úrovňové C, FORTRAN, ...

vědecké výpočty, vestavěné systémy

objektové C++, Java, C#, ...

klasické aplikace, rozsáhlé systémy

skriptovací Python, Perl, PHP, Javascript, ...

programování pro web, skriptování

deklarativní Prolog, LISP, Haskell, ...

umělá inteligence

- **vysoko-úrovňový** – velká míra abstrakce, „spustitelný pseudokód“
- **interpretovaný** – pomalejší než kompilovaný, ale větší volnost
- **pedagogický** – byl tak navržen
- **moderní a široce používaný** – cca 7 nejpoužívanější jazyk
- volně a snadno **dostupný** na všech platformách

# Programování v tomto kurzu

- důraz na obecné principy, nikoliv specifika Pythonu
- většina konceptů snadno a velmi podobně realizovatelná v jiných jazycích

# Motivační příklady na algoritmizaci

- jednorozměrné piškvorky
- Hanojské věže
- Sudoku
- vězni a karty

# Vězni a karty

- Albert dostane **5 karet** ze standardního balíčku 52 karet
- vybere jednu z nich
- zbylé čtyři poskládá do zvoleného pořadí a dá je Bedřichovi
- Bedřich musí určit, jaká je ta pátá odstraněná karta
- Jaký systém si mají Albert s Bedřichem domluvit?