Programovanie, algoritmy, zložitosť / ÚINF/PAZ1c

PAZ1C

Róbert Novotný
robert.novotny@upjs.sk

29. 9. 2009

Formality a byrokracie

- Teoretické cvičenie („prednáška")
  - streda, 15.20, P10
  - zlúčené s cvičením v P11
- Praktické cvičenia
  - pondelok, 10:45, P3
  - štvrtok, 8.55, P4
- Všetky inštrukcie na

http://ics.upjs.sk/~novotnyr/wiki/Java/PAZ1c

Každý musí byť zaradený na jedno praktikum a jedno teoretické cvičenie!

Vstupné vedomosti

- toto je tretí semester programovania
- od študenta sa očakáva
  - znalosť základného procedurálneho programovania
    - viem, čo je cyklus, podmienka, premenná, dátový typ, metóda (procedúra, funkcia)
  - znalosť OOP
    - pozri študijné texty z PAZ1a, PAZ1b, prípadne megaprezentácia z PAZ1c minulých rokov
  - idea o programovacích technikách a algoritmoch
    - rekurzia...

Témy a motívy predmetu

- Pokročilé princípy objektovo-orientovaného programovania
- Dôraz na dizajn a udržovateľnosť programov
- Zamerané na príklady z praxe
- Programovací jazyk Java

Požiadavky na hodnotenie

- **Účasť** na praktických cvičeniach (30%)
- Záverečný **projekt** (40%) – nutná podmienka
- Dva **testy**: v polovici semestra + na konci (15% + 15%)
  - jedna možnosť opravy jedného testu

Požiadavky na hodnotenie

- Extra požiadavka
  - domáca príprava

Nejdůležitější: vlastní praxe
Na přednáškách se nikdo nikdy programovat nenaučil

-- Filip Zavoral, MFF UK Praha

And now...

PAZ1C

...a teraz...

Ako vytvoriť rozumný objektový návrh?

- na PAZ1a/b sme sa venovali minimalistickým projektom
  - 1-2 triedy
- praktické projekty však majú rádovo vyšší počet tried
  - JDK 6: 7208 tried
  - Android 2.2: 2642 tried

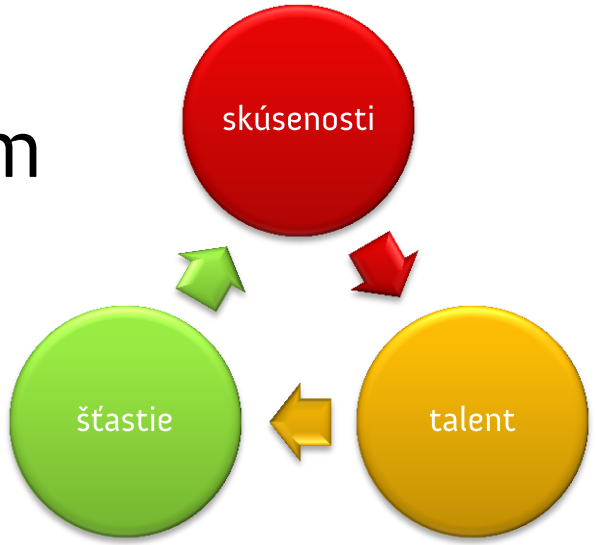
Dôraz na udržovateľnosť

- s narastajúcou veľkosťou projektu sa zvyšujú nároky na udržovateľnosť
  - udržovateľný (objektový) návrh je nutnosť
- programátor vyvíja softvér pre ostatných, nie pre seba
  - snaha predísť write-only kódu

```
#!/bin/perl -sp0777i<X+d*lMLa^*lN%0]dsXx++lMlN/dsM0<j]dsj
$/=unpack('H*',$_);$_=`echo 16dio\U$k"SK$/SM$n\EsN0p[lN*1
 lK[d2%Sa2/d0$^Ixp"|dc`;s/\W//g;$_=pack('H*',/((..)*)$/)
```

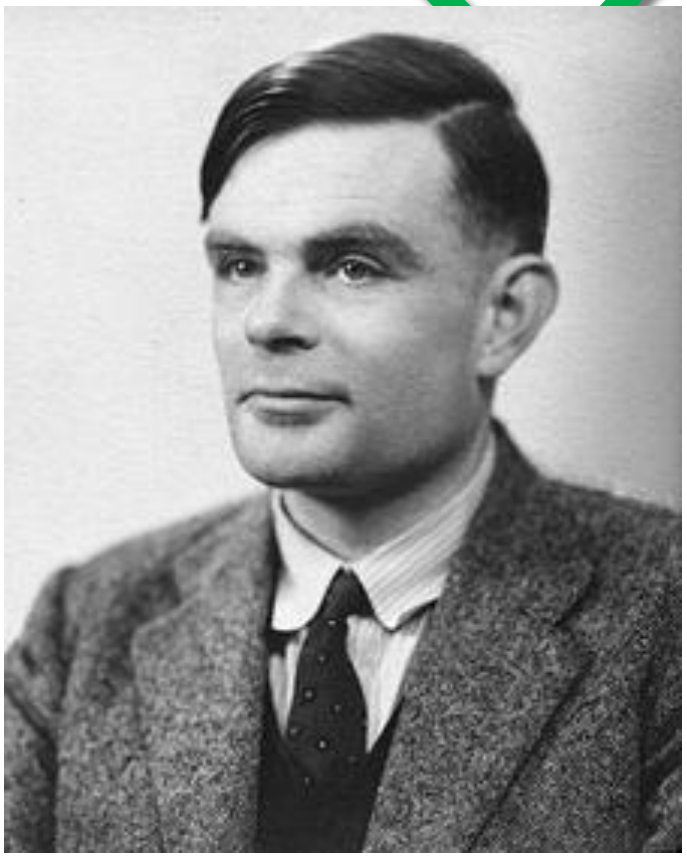
Ako identifikovať triedy?

- odhalenie tried je základom dobrého návrhu
- umenie toho, čo vybrať a čo zahodiť
  - **skúsenosti** využívajú *best practices* z minulých projektov
  - **talent** prechádzať chybám
  - **šťastie**, vďaka ktorému projekt odolá zmenám



Príklad z... praxe: Turingov stroj

...neohraničená kapacita pamäte, ktorá je reprezentovaná nekonečnou páskou rozdelenou na štvorčeky, pričom do každé z nich možno vytlačiť symbol. V danej chvíli je v stroji jeden symbol - nazýva sa načítavan symbol. Stroj môže upraviť načítavaný symbol a správanie tohto stroja je určené týmto symbolom, ale ostatné symboly na páske nijak neovplyvňujú správanie stroja. Páska však môže byť posúvaná dopredu i dozadu skrz stroj, čo je jedna zo základných operácií stroja, t. j. do stroja sa potenciálne môže dostať ľubovoľný symbol na páske.

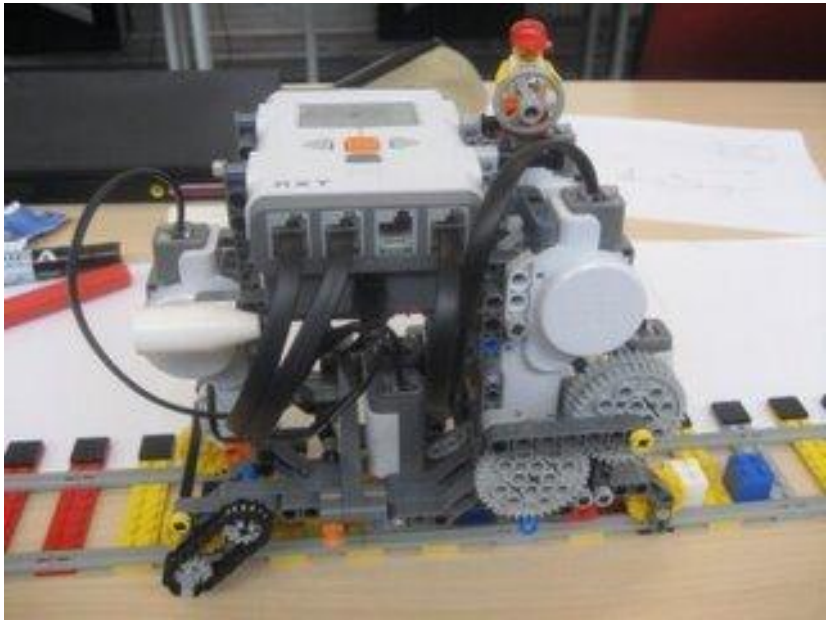


--- Alan Turing, 1937

Príklad z... praxe: Turingov stroj

- Nekonečná **páska** rozdelená na bunky. Každá z buniek obsahuje symbol abecedy.
- **Hlava**, ktorá sa pohybuje nad páskou a v každej chvíli sa nachádza nad jednou bunkou.
- **Stav**, v ktorom sa stroj aktuálne nachádza.
  - stroj sa vždy nachádza v jedinom stave, stavov môže byť konečne veľa.
- **Inštrukcie** (prechodová funkcia) qiXYqjD
  - ak je stroj v stave qi a pod hlavou je symbol X, do bunky sa zapíše Y, stroj sa prepne do stavu qj a pohne sa o jednu bunku doľava alebo doprava.

Príklad z... praxe: Turingov stroj



http://www.youtube.com/watch?v=cYw2ewoO6c4

Príklad z... praxe: Turingov stroj

Zamerajte sa na podstatné mená z opisu!

- Nekonečná **páska** rozdelená na **bunky**. Každá z buniek obsahuje **symbol abecedy**.
- **Hlava**, ktorá sa pohybuje nad páskou a v každej chvíli sa nachádza nad jednou bunkou.
- **Stav**, v ktorom sa **stroj** aktuálne nachádza.
  - stroj sa vždy nachádza v jedinom stave, stavov môže byť konečne veľa.
- **Inštrukcie** (prechodová **funkcia**) qiXYqjD
  - ak je stroj v stave qi a pod hlavou je symbol X, do bunky sa zapíše Y, stroj sa prepne do stavu qj a pohne sa o jednu bunku doľava alebo doprava.

Príklad z... praxe: Turingov stroj

Zamerajte sa na podstatné mená z opisu!

páska bunka symbol
abeceda hlava stav
stroj inštrukcia
prechodová funkcia

Umenie výberu, umenie zahadzovania

PAZ1C

páska bunka symbol
abeceda hlava stav
stroj inštrukcia
prechodová funkcia

- ktoré podstatné mená nemajú zmysel?
- obvykle:
  - tie, ktoré nie sú relevantné vzhľadom na systém
  - tie, ktorým ťažko nájsť správanie / vlastnosti
  - tie, ktoré sú pokryté ostatnými objektami

Objekty sú definované tým, čo s nimi môžeme vykonávať.

Umenie výberu, umenie zahadzovania

páska bunka symbol abeceda hlava stav stroj inštrukcia prechodová funkcia

- **páska** = množina buniek
- **bunka** = súčasť pásky, obsahuje symbol
- **abeceda** = množina symbolov
- **hlava** = ?
- **stav** = v teórii reprezentovaný číslom
- **Turingov stroj** = pozostáva z ostatných súčastí
- **inštrukcia** = usporiadaná pätica
- **prechodová funkcia** = množina inštrukcií

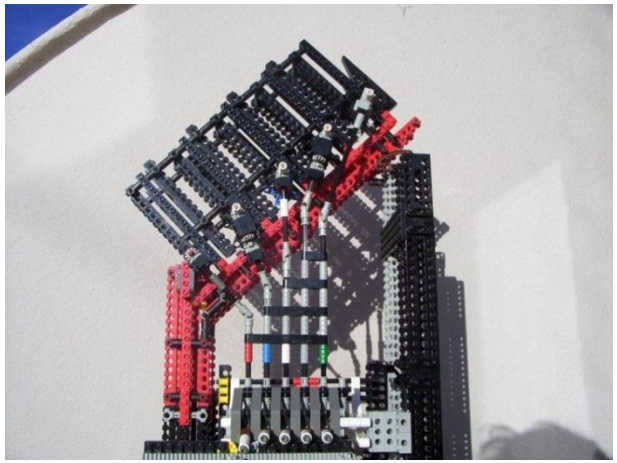
Umenie výberu, umenie zahadzovania

- **páska** = množina buniek = množina symbolov
- **hlava** = dôležité je len vedieť, ktorý symbol na páske načítavame = kde na páske sa nachádzame
- **stav** = číslo
- **inštrukcia** = usporiadaná pätica
- **prechodová funkcia** = množina inštrukcií
- **Turingov stroj** = pozostáva z ostatných súčastí

páska ~~bunka symbol~~ ~~abeceda~~ ~~hlava~~ ~~stav~~ stroj inštrukcia ~~prechodová funkcia~~

Umenie výberu, umenie zahadzovania

- **Turingov stroj**
  - **páska** – pokojne ju možno reprezentovať jednoduchým zoznamom symbolov
  - pozícia hlavy na páske
  - **stav** = číslo
  - **zoznam inštrukcií**
- **inštrukcia** = usporiadaná pätica



Trieda je charakterizovaná tým, čo robí

- pri návrhu triedy treba identifikovať
  - **správanie**, ktoré od triedy očakávame
  - na základe správania určíme **stavové premenné**
    - potrebné pre dosiahnutie funkcionality
- trieda má poskytovať požadované správanie
  - inak sa má tváriť ako čierna skrinka

Aké správania očakávame od TS?

PAZ1C

- zadaj vstup na pásku
- zadaj sadu inštrukcií
- vykonaj ho a vráť výsledok

V tejto chvíli máme dve možnosti návrhu!

Peter Boothe <peter@cs.uoregon.edu>

TS: variant Á


```
public class TuringMachine {
  public void setInput(char... input);

  public void setInstructions(
    Collection<Instruction> instructions);

  public int execute();
}
```

TS: variant Á

- stav, ktorý potrebujeme evidovať
  - páska
  - poloha na páske
  - číslo stavu
  - zoznam inštrukcií

TS: variant Á


```
public class TuringMachine {
   private int[] tape;
   private int currentState;
   private int position;
   private Collection<Instruction> instructions;
}
```

TS: použitie

- klient interaguje len s metódami
- vôbec ho nezaujíma vnútorný stav
- na najvyššej úrovni vníma klient TS ako **čiernu skrinku**, ktorá dostane vstup a inštrukcie a vráti výstup
- inštančné premenné sú preto privátne (skryté)