Pedagogika Výučby Programovania

Recepty na postup pri vývoji programov

Viera Krňanová Proulx

Northeastern University, Boston, MA, USA
vkp@ccs.neu.edu

Informatika: Veda Dizajnu

- Dizájn v kontexte informačných systémov
- Algebra a programovanie
- Informácia a kódované dáta
- Dizájn abstrakcií = softvérové knižnice
- Knižnice pre začiatočníkov

Informatika: Veda Dizajnu

Konkrétne komplexné systémy

- Automobily 
- Mrakodrapy 
- Slovenské železnice 

Informatika: Veda Dizajnu

Komplexné webové stránky

Systémy na predpovedanie počasia

Zdravotné informačné systémy

Wikipedia

Počítačové hry

Algebra a Programovanie

- Je súčet troch číslic deliteľný siedmimi?
- Vypočítaj pozíciu lopty kopnutej v určitom uhle a určitou rýchlosťou
- Kde sa nachádza sa dané písmeno v danom reťazci?

Funkcie s vchodnými dátami a výslednými dátami

Algebra a Programovanie

Funkcie s vchodnými dátami a výslednými dátami

Didaktika vývoja funkcie (procedúry)

1. Rozmýšľaj aké dáta sú vstupné a výstupné
2. Napíš komentár a hlavičku (kontrakt)
3. Urob príklady funkcie pre niekoľko daných vchodov
4. Inventúra: čo máme k dispozícii
5. Programuj funkciu
6. Over si správnosť na príkladoch z tretieho kroku

Algebra a Programovanie 1

- **Je súčet troch číslic deliteľný siedmimi?**
  - vstupy: tri číslice výstupy: boolean
- **Vypočítaj pozíciu lopty kopnutej v určitom uhle a určitou rýchlosťou**
  - vstup: uhol, rýchlosť, (príťažlivosť) výstup: pozícia (x,y)
- **Kde sa nachádza dané písmeno v danom reťazci?**
  - vstup: písmeno, reťazec výstup: int

Analyzuj problém, typy vstupov, výstupov

Algebra a Programovanie 2

- **Je súčet troch číslic deliteľný siedmimi?**
  - `// je súčet daných troch číslic deliteľný siedmimi`
  - `boolean delitelny7(int c1, int c2, int c3)`
- **Vypočítaj pozíciu lopty kopnutej určitou rýchlosťou**
- **... za daný čas, začínajúc na pozícii (0.0, 0.0)**
  - `// pozícia lopty pre danú rýchlosť v danom čase`
  - `CartPt pozicia(double dx, double dy, int t)`
- **Kde sa nachádza sa dané písmeno v danom reťazci?**
- **... vydaj -1 ak sa nenachádza**
  - `// kde sa nachádza sa dané písmeno v danom reťazci?`
  - `int kdeJe(char p, String slovo)`

Komentár, hlavička funkcie-procedúry

Algebra a Programovanie 3

- **Je súčet troch číslic deliteľný siedmimi?**
  - `delitelny7(1, 2, 3) -> false`
  - `delitelny7(2, 5, 7) -> true`
- **Vypočítaj pozíciu lopty kopnutej určitou rýchlosťou**
  - `pozicia(10.0, 20.0, 3) -> CartPt(30.0, 60.0)`
- **Kde sa nachádza sa dané písmeno v danom reťazci**
  - `kdeJe("a", "dedo") -> -1`
  - `kdeJe("a", "mama") -> 1`

Príklady použitia s očakávanými výsledkami

Algebra a Programovanie 4

- **Je súčet troch číslic deliteľný siedmimi?**
  - `inventúra: funkcia 'mod(int) -> int'`
- **Vypočítaj pozíciu lopty kopnutej určitou rýchlosťou**
  - `inventúra: dx, dy, t; začiatok: (0, 0)`
- **Kde sa nachádza sa dané písmeno v danom reťazci**
  - `inventúra: p, s,`
    `funkcia substring(s, i, k)`

-štruktúra -- vyber polia, eviduj ich typy
-varianty -- spracuj každú variantu osobitne
-zaeviduj funkcie ktoré sa dajú aplikovať

Algebra a Programovanie 5

- **Je súčet troch číslic deliteľný siedmimi?**
  [return (c1 + c2 + c3) mod 7 == 0]
- **Vypočítaj pozíciu lopty kopnutej určitou rýchlosťou**
  [return new CartPt(t * dx, t * dy)]
- **Kde sa nachádza sa dané písmeno v danom reťazci**

```
[for (int i = 0; i < length(slovo); i++)
   if (substring(slovo, i, i+1) == p)
   return i;
return -1;]
```

Len teraz vypracuj funkciu/procedúru - program

Algebra a Programovanie 6

- **Je súčet troch číslic deliteľný siedmimi?**
  - `delitelny7(1, 2, 3) -> false`
  - `delitelny7(2, 5, 7) -> true`
- **Vypočítaj pozíciu lopty kopnutej určitou rýchlosťou**
  - `pozicia(10.0, 20.0, 3) -> CartPt(30.0, 60.0)`
- **Kde sa nachádza sa dané písmeno v danom reťazci**
  - `kdeJe("a", "dedo") -> -1`
  - `kdeJe("a", "mama") -> 1`

Over si správnosť - testy sú definované v treťom kroku

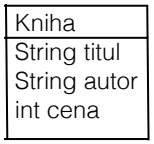
Informácia a kódované dáta

- **základné typy dát:** čísla, reťazce, images, bool
- **triedy/štruktúry:** niekoľko dát reprezentuje informáciu
- **referencia:** údaj v triede je objekt inej triedy
- **varianty:** niekoľko variantov patria spolu
- **kombinácie** týchto možností

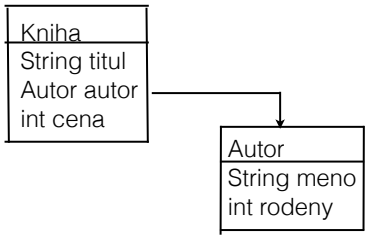
Analyzuj problém, definuj dáta ktoré potrebuješ, (podľa kategórií hore), urob niekoľko príkladov takýchto dát

Informácia a kódované dáta

- **základné typy dát:** int, String, boolean, image
- **triedy:**

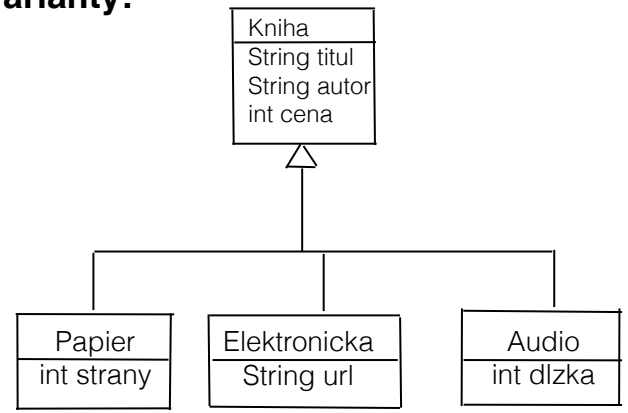


- **referencia:**



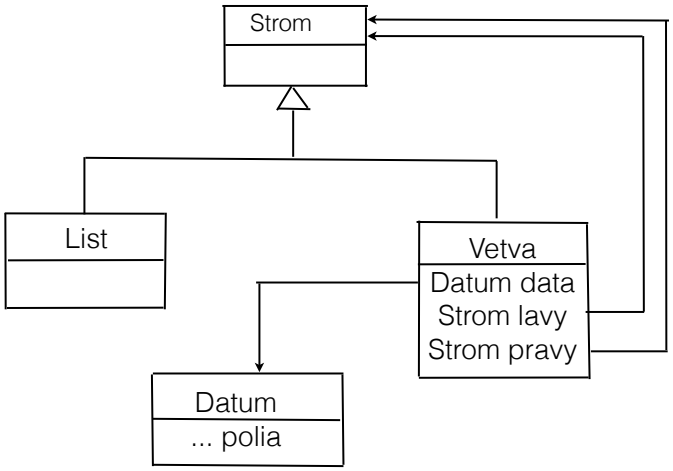
Informácia a kódované dáta

- **varianty:**

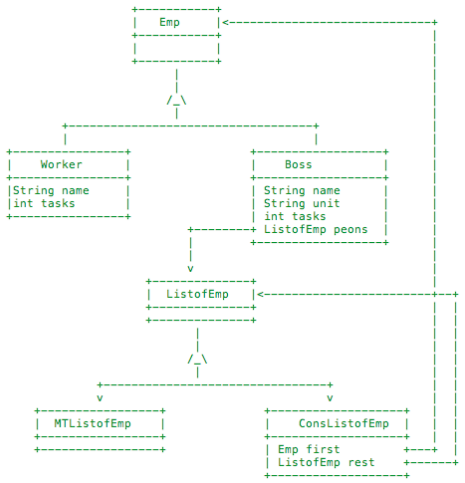


Informácia a kódované dáta

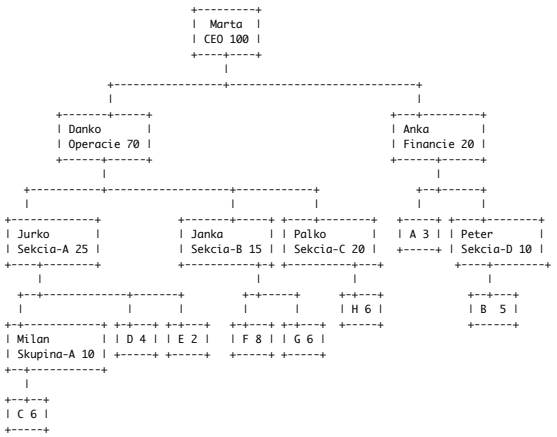
- **kombinácie:**



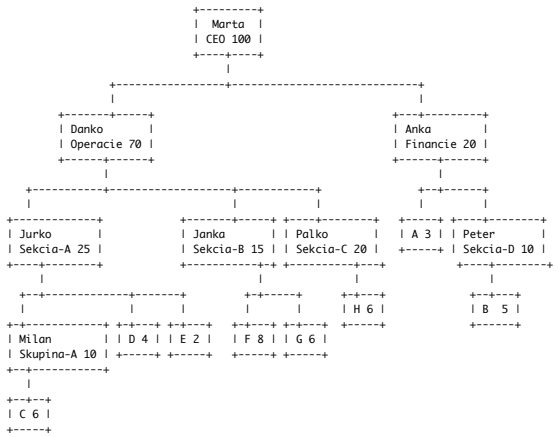
Informácia a kódované dáta



Informácia a kódované dáta



Informácia a kódované dáta



Koľko ľudí je v pracovnej jednotke s daným menom?

Čo je dôležité:

- Existujú rôzne typy dát, a môžeme ich kombinovať ak informácia má viac častí alebo variantov
- Úlohou programu je vytvoriť z daných dát nové dáta, ktoré reprezentujú nové informácie
- Každá funkcia alebo procedúra má mať len jeden úkol: použite pomocné funkcie a procedúry ak treba
- Input-output nie je dôležitý sám o sebe: programujeme osobitne spracovanie vstupov a prípravu dát na výstupy

Čo je dôležité:

- Dôležité je reprezentovať informácie ako dáta
- Dizajnovať každú funkciu a procedúru systematicky, s testovaním
- Budovať väčšie programy abstrakciami

  ak sa niečo opakuje (a len malá časť je rôzna), vyrobiť program kde varianty sú repreentované parametrami a spoločná časť sa vyskytne len raz

Dizajn abstrakcií = knižnice

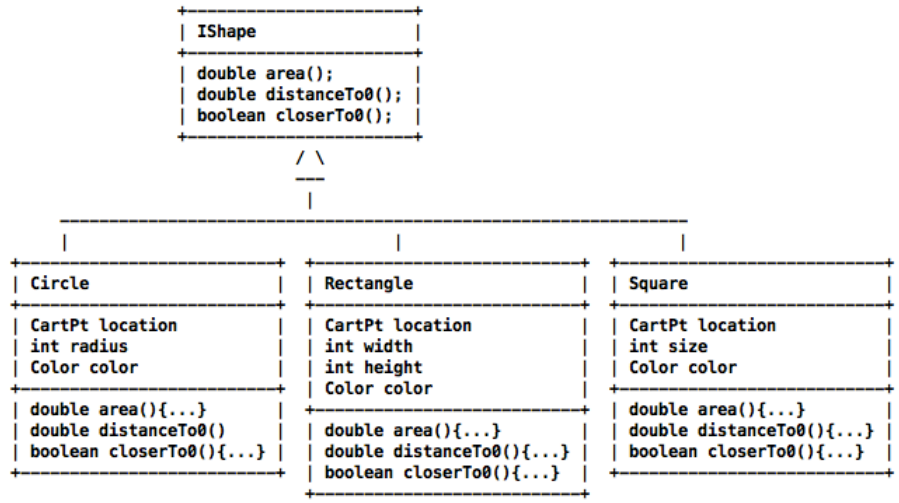
- Pozna**č** všetky miesta kde sa podobné programy líšia.
- Zme**ň** na parametre mista rozli**č**ností a prepíš program s nimi.
- Prepíš originálne programy tým že volajú nový program so správnymi argumentami.
- Over si, že všetky testy z predošlého programu prejdú úspešne.

Dizajn abstrakcií = knižnice

- rozhranie -- abstraktné triedy
- funkčné objekty (funkcie ako parametre)
- parametrizované typy
- traversals: poskytnutie členov súboru dát
- abstraktné dátové typy

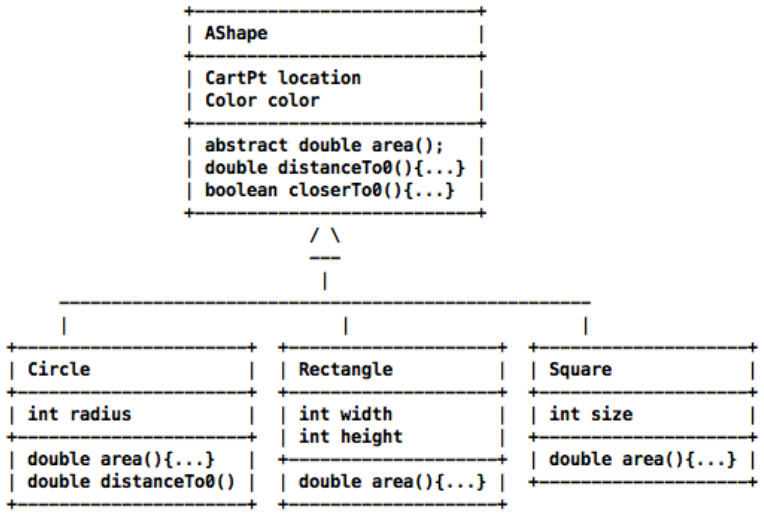
Kľúče k porozumeniu ako vytvoriť-použiť mnoho-úžitkové programy

Dizajn abstrakcií - rozhranie



Urob abstraktnú triedu: spoločné polia, metódy

Dizajn abstrakcií - rozhranie



Urob abstraktnú triedu: spoločné polia, metódy

Dizajn abstrakcií - funkčné objekty

- zorad' súbor typu T (v akom **poradí**?)

  porovnávacia funkcia: (T, T) -> int
- vyber všetky údaje s danou **vlastnosťou**

  výberový predikát: (T) -> boolean
- urob žiadaný **úkol**

  metóda úkolu ktorý sa ma konať

Ťažké ak jazyk nemá podporu pre funkcie prvej triedy

Dizajn abstrakcií - parametrizované typy

- binárne zorad'ovacie stromy čísiel, reťazcov, kníh

  Strom<T> --> Strom<Integer>, Strom<String>, ...
- zoznam ľudí, piesní, obrazov,

  Zoznam<T> --> Zoznam<Person>, Zoznam<Song>, ...

Nie je portrebné v jazykoch čo nemajú typy

Dizajn abstrakcií - prechody (traversals)

Niekroté algoritmy potrebulú použiť pojednom všetky elementy nejakej množiny údajov.

Iterátory, “návštevníci”, a špeciálne vytvorené triedy alebo jazykové štruktúry môžu poskytnúť takúto službu.

Algoritmy potom používajú tieto služby na prístup k členom množiny údajov, bez toho aby vedeli ako je tá monožina údajov reprezentovaná.

Iterátor produkuje údaje zo súboru alebo z poľa ...

Dizajn abstrakcií - abstraktné dátové typy

- Vector (ArrayList) -- štruktúry s indexovým prístupom
- Rad, Zásobník
- Uprednostňovaný rad (halda)
- Tabuľka, Hašovacia tabuľka -- (kľúč - hodnota)
- Graf

Vid' niekoľko realizácií -- porozumej výhody a nevýhody

Knižnice pre začiatočníkov: Java

- Typický programovací jazyk nie je vhodný pre začiatočníka
- Obsahuje prvky ktorým začiatočník nerozumie, ale chybové hlásenie sa na ne odvoláva
- Programovanie vstupov, výstupov a interakcií je zložité a treba dosť vedieť, kým to začiatočník môže programovať
- Programovanie testov vyžaduje znalosť rôznych metód na porovnávanie rovnosti medzi údajmi

Knižnice pre začiatočníkov: Java

Programovací jazyk pre začiatočníkov: FunJava

- Každá trieda môže implementovať len jedno rozhranie (interface)
- Hodnota každého poľa musí mať hodnotu ked' sa objekt zkonštruuje
- Hodnota poľa sa nezmení v programe
- Jazyk má len dva príkazy:

  return výraz

  if (podmienka) príkaz else príkaz

Typický programovací jazyk nie je vhodný pre začiatočníka

Knižnice pre začiatočníkov: Java

Programovací jazyk pre začiatočníkov: FunJava

- Každá trieda môže implementovať len jedno rozhranie (interface)
- Hodnota každého poľa musí mať hodnotu ked' sa objekt zkonštruuje
- Hodnota poľa sa nezmení v programe
- Jazyk má len dva príkazy:

  return výraz

  if (podmienka) príkaz else príkaz

Obsahuje prvky ktorým začiatočník nerozumie, ale chybové hlásenie sa na ne odvoláva -- tu to nie je problém

Knižnice pre začiatočníkov: Java

Knižnica pre začiatočníkov: World, Canvas

- jednoduché funkcie na kreslenie geometrických tvarov (kruh, disk, obdĺžnik, čiara, text)
- World: (programovanie interaktívnych hier)

    Canvas theCanvas -- pole ktoré je vidieť

    World onTick()

    World onKeyEvent(String ke)

    boolean endOfWorld()

    boolean draw()

Programovanie vstupov, výstupov a interakcií je zložité a treba dosť vedieť, kým to začiatočník môže programovať

Knižnice pre začiatočníkov: Java

Knižnica pre začiatočníkov: World, Canvas

- jednoduché funkcie na kreslenie geometrických tvarov (kruh, disk, obdĺžnik, čiara, text)
- World: (programovanie interaktívnych hier)

  svet začne animáciu funkciou

  bigBang(int šírka, int výška, double tick)

Programuje sa len model - je možné si overiť správnosť programu testami

Programovanie vstupov, výstupov a interakcií je zložité a treba dosť vedieť, kým to začiatočník môže programovať

Knižnice pre začiatočníkov: Java

- World: (programovanie interaktívnych hier)
  - pre pokročilejších aj myš
  - aj hudba
  - (programovanie postupností nôt a ich kombinácií)

Programuje sa len model - je možné si overiť správnosť programu testami

Programovanie vstupov, výstupov a interakcií je zložité a treba dosť vedieť, kým to začiatočník môže programovať

Knižnice pre začiatočníkov: Java

Knižnica pre začiatočníkov: Tester

- špeciálna knižnica na pripravovanie a hodnotenie testov
- porovnávanie dvoch objektov na základe ich hodnôt, nie referencie
- porovnávanie nepresných hodnôt s približnostiami
- špeciálne testy pre konštruktory, výnimky, iterátory, jednu z možností, hodnota v rozsahu

Programovanie testov vyžaduje znalosť rôznych metód na porovnávanie rovnosti medzi údajmi

Knižnice pre začiatočníkov: Java

Knižnica pre začiatočníkov: Tester

- vypracuje referát so všetkými výsledkami:
  - vytlačí hodnoty všetkých objektov (pretty-print)
  - vytlačí výsledky všetkých testov
  - ak test neprejde,
    - ukáže pri sebe očakávané a vypočítané hodnoty
    - poznačí kde sa výsledky líšia
    - poskytne linku k testu čo neprešiel

Programovanie testov vyžaduje znalosť rôznych metód na porovnávanie rovnosti medzi údajmi

Čo je dôležité:

- Namiesto púheho používania knižníc naučiť študentov ako sa budujú knižnice
  - funkcie ktoré definujú zorad'ovania
  - algoritmy ktoré používajú iterátory a pomocné funkcie (sort, filter, andmap)
- Abstraktné dátové typy a ich implementácia
- Základy hodnotenia zložitosti algoritmov a dátových štruktúr

Čo je dôležité:

- Princípy spracovania dát pri vstupoch a výstupoch:
  - preloženie reťazca na číselnú hodnotu
  - zakódovanie dát pri výstupe do súborov
  - event handling - princípy, použitie
  - prúdy vstupov a výstupov
- Princípy dizajnu:
  - test-first dizajn
  - jeden úkol, jedna procedúra (funkcia)

Na záver

Ďakujem za pozornosť

Program by Design:

http://www.programbydesign.org

Java knižnice:

http://www.ccs.neu.edu/javalib

Laboratóriá, materiály:

http://www.ccs.neu.edu/home/vkp/Teaching

Program pre druhý stupeň:

http://www.bootstrapworld.org