Co bude obsahem písemky

Navrhnout, jak by se úloha dala řešit.

Nemusí obsahovat žádný zdrojový kód

(i když popis (pseudo)kódem může být kratší a přesnější než slovy).

V odpovědi popište

0) upřesnění zadání, pokud je potřeba

1) postřehy

2) zdůvodněnou volbu algoritmu

3) reprezentaci dat

4) dekompozici programu

5) diskusi

VŠECHNY TYTO ČÁSTI JSOU DŮLEŽITÉ!

Příklad zadání - ANKETA

Navrhněte program, který zpracuje výsledky studentského hodnocení učitelů.

Vstup:

1) soubor obsahující na každém řádku
   učitel – předmět – hodnocení
2) seznam kateder
   učitel – katedra

Výstup:

1) soubor učitel – předmět – průměrné hodnocení – odchylka
   setříděný podle učitel – předmět
2) dtto, setříděný podle katedra – učitel – předmět

Příklad zadání - ANKETA

Velikosti a omezení:

Počet učitelů: stovky

Počet předmětů: tisíce

Počet hlasovacích lístků: desetitisíce

Velikost paměti: 1kB

Velikost disku: neomezeně

Čas: přiměřeně (minuty až hodiny)

Generické metody a třídy

- jeden a více typových parametrů
- možnost omezit pomocí where
- where:
  - interface
  - class
  - struct
  - rodičovská třída
  - ...apod.

Zvěřinec OOP jazyků

- Python (1991)
- C++ (1985)
- Java (1995)
- C# (2000)

Simula67 (1967), Smalltalk (1971),
Object Pascal (1989), Visual Basic (1991)
JavaScript (1995), PHP (1995)...a další

Zvěřinec OOP jazyků 2

Specifika, zvláštnosti, úchylky:

- C# a Java pouze objektově
- C++ pracuje přímo s pamětí
- C#, Java, PY: garbage collector
- PY nemá statické typování
- PY nemá private/protected
- PY, Java ani C++ nemají properties
- ...

Zvěřinec OOP jazyků 3

Specifika, zvláštnosti, úchylky:

- Java nemá var-parametry
- Java class = stejnojmenný soubor
- ...

Všechny jazyky se vyvíjejí a přebírají to, co se jinde osvědčilo.

Hygiena programování

. (praktické tipy)

1. Oddělovat různé činnosti

1. analýza
2. design
3. GUI design
4. kódování
5. texty, grafika, zvuky
6. testování (psaní testů)
7. ladění
8. dokumentace

   resources, šablony výstupů…

2. Pracovní deník

1. ujasnění si, CO budu dělat teď
2. rozhovor (kačenka)
3. záznam pro soustředění na jednu věc
   (viz ToDo-list)
4. pozdější dohledání (...a viz verse)
5. sledování, jak dlouho mi co trvá

3. ToDo-list, bug-list, issue tracker

* //TODO ve zdrojovém kódu
* deník, konec nebo vyhledatelná značka
* samostaný soubor
* software (GitHub, Trello, GitLab)

https://wac-cdn.atlassian.com/dam/jcr:20386fdc-1de9-43ab-8ae6-4564aa8e7e94/image4.png?cdnVersion=1605

4. Správa verzí

* CSV, SVN (Subversion),
  Hg (Mercurial), Git
* možnost návratu k minulým verzím
* sledování rozdílů

I při práci na jednom počítači.
I pro ne-programátorské projekty.

5. Automatické testy, TDD

* unit-testy
    1. Napsat testy
    2. Spustit testy a ujistit se, že selžou
    3. Napsat kód
    4. Ladit kód, dokud nesplní testy
    5. Refaktorovat kód
    6. Opakovat
* testování funcionalit
* výkonnostní testy
* regresní testy i dnes funguje, co fungovalo včera
* Průběžná integrace (CI)