ZÁPADOČESKÁ
UNIVERZITA
V PLZNI

Samostatná práce

z předmětu

Počítače a programování 1

Jméno a příjmení: *Martin Lipinsky*
Osobní číslo: ***A05450***
Studijní skupina: *Dálkové studium*
Obor: *INIB-VTB*
E-mail: *martin@lipinsky.cz*

Označení zadání: *Varianta A - Řazení*

Datum odevzdání: 9.1.2006

Zadání

Napište program v jazyku Java pro realizaci třech následujících algoritmů řazení (vzestupně i sestupně) metodami Insert sort(vkládáním), Select sort (výběrem) a Bubble sort (bublinkové řazení). Pracujte s celými nebo reálnými hodnotami prvků generovanými generátorem náhodných čísel. Program bude umožňovat i vstup dat z klávesnice (dle potřeby může být realizován i souborový vstup). Součástí metod bude zjištění počtu výměn (přesunů) prvků a počtu jejich porovnání.

Stručný popis

Při psaní programu jsem se znažil maximálně využít objektový přístup k programování. Vše obecné pro řazení je definováno v třídě **Sort** (interface) respektive v třídě **SortAdapter**. Ta definuje operace společné pro každé třídění (prohození dvou prvků, porovnání dvou prvků v závislosti na vzestupnosti či sestupnosti třídění) a přidává požadované sledování strojové náročnosti daných algoritmů tedy počet porovnání a počet výměn prvků.

Třídy **BubbleSort**, **InsertSort** a **SelectSort** potom implementují jednotlivé známé algoritmy řazení (viz přiložený soubor sort.pdf s detailním popisem algoritmů ze kerého bylo při řešení čerpáno) .

Třída **SortTest** obsahuje statické metody pro vygenerování náhodného pole celých čísel o délce daném konstantou RANDOM_LENGHT, převod pole na string pro možnost jeho vytištění a metodu která zavolá příslušné řazení na náhodně vygenerované pole vzestupně, potom na již vzestupně setříděné pole znovu vzestupně, a nakonec na vzestupně setříděné pole sestupně. Výsledek tiskne na konzoli ve formě seřazených dat + počtu porovnání a výměn.Z metody main je pak zavoláno generování pole + postupné zavolání všech druhů řazení na takto vygenerovaná data.

Ověření funkce řazení

Všechny tři způsoby třídění byli ověřeny. Přikládám výstup programu pro náhodně vygenerované pole:

```
Tridena data : [ 38, 92, 95, 44, 24, 58, 99, 7, 89, 70 ]

Otestuj narocnost BUBBLE sortu
Od nejmensiho: [ 7, 24, 38, 44, 58, 70, 89, 92, 95, 99 ] SC = 22, CC = 72
Jeste jednou : [ 7, 24, 38, 44, 58, 70, 89, 92, 95, 99 ] SC = 0, CC = 9
Od nejvetsiho: [ 99, 95, 92, 89, 70, 58, 44, 38, 24, 7 ] SC = 45, CC = 90

Otestuj narocnost SELECT sortu
Od nejmensiho: [ 7, 24, 38, 44, 58, 70, 89, 92, 95, 99 ] SC = 9, CC = 45
Jeste jednou : [ 7, 24, 38, 44, 58, 70, 89, 92, 95, 99 ] SC = 9, CC = 45
Od nejvetsiho: [ 99, 95, 92, 89, 70, 58, 44, 38, 24, 7 ] SC = 9, CC = 45

Otestuj narocnost INSERT sortu
Od nejmensiho: [ 7, 24, 38, 44, 58, 70, 89, 92, 95, 99 ] SC = 22, CC = 29
Jeste jednou : [ 7, 24, 38, 44, 58, 70, 89, 92, 95, 99 ] SC = 0, CC = 9
Od nejvetsiho: [ 99, 95, 92, 89, 70, 58, 44, 38, 24, 7 ] SC = 45, CC = 45
```

Strojová náročnost řazení pro vzrůstající počet prvků

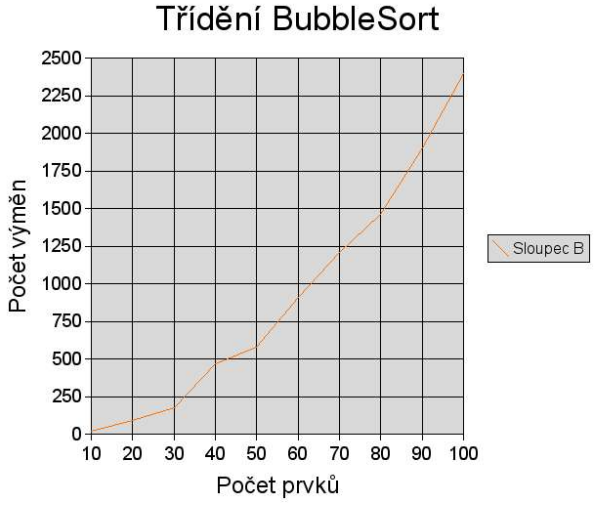
Z možných zadání jsem se rozhodl ověřit závislost počtu prvků na počtu výměn prvků při řazení metodou Bubble sort. Teoreticky by mělo jít o závislost o málo méně než kvadratickou což vyplývá z principu algoritku.

Třída **SortTest** proto obsahuje statickou metodu, která vygeneruje pole délky 10 až 100 s krokem 10 prvků a k příslušnému počtu náhodně generovaných prvků zavolá metody Bubble sort a doplní tabulku o příslušný počet výměn.

Při jednom ze spuštění této metody vypadal výstup takto:

```
Pocet prvku: 10
Pocet vymen: 19
------------------------
Pocet prvku: 20
Pocet vymen: 91
------------------------
Pocet prvku: 30
Pocet vymen: 178
------------------------
Pocet prvku: 40
Pocet vymen: 471
------------------------
Pocet prvku: 50
Pocet vymen: 581
------------------------
Pocet prvku: 60
Pocet vymen: 909
------------------------
Pocet prvku: 70
Pocet vymen: 1211
------------------------
Pocet prvku: 80
Pocet vymen: 1462
------------------------
Pocet prvku: 90
Pocet vymen: 1905
------------------------
Pocet prvku: 100
Pocet vymen: 2400
```

Závislost je vyjádřena v tabulce:



Poděkování

Chtěl bych poděkovat svému bratrovi Luboši Lipinskému za nemalou pomoc při mém zasvěcení do tajů objektového programování.