Opis

Zostałeś poproszony o implementację algorytmu do uniwersalnego programu sortującego dane w formacie tsv. Wykorzystując algorytm *QuickSort* w wersji **iteracyjnej, o stałej pamięci oraz bez użycia stosu,** posortuj wartości w tabeli według podanej kolejności. Algorytm należy wykonywać, dopóki wielkość podzadań jest większa lub równa **20**. Następnie, pozostałe podzadania należy wykonać przejściem *sortowania przez wstawianie*.

Wejście

Mimo, iż implementacja sortowania została pozostawiona Tobie, kierownictwo nałożyło sztywne ramy wejścia i wyjścia z programu. Dane wczytywane są ze standardowego wejścia zgodnie z poniższą specyfikacją.

Pierwsza linia wejścia zawiera liczbę całkowitą oznaczającą liczbę zestawów danych. Pierwsza linia każdego zestawu zawiera dwie liczby całkowite oddzielone spacją: liczbę kolumn oraz liczbę wierszy w tabeli. Następne linie zawierają tabelę w formacie tsv. Pierwszy wiersz zawiera listę kolumn (nagłówek) oddzielonych znakami tabulacji. Po niej podawane są wiersze danych, w których każda kolumna również oddzielona jest znakiem tabulacji. Wartość każdej komórki jest liczbą rzeczywistą większą od zera, a część dziesiętna oddzielona jest znakiem przecinka.

Po wczytaniu danych, następna linia zawiera liczbę całkowitą oznaczającą liczbę zapytań. Każde zapytanie podawane jest w osobnej linii i składa się z dwóch elementów:

<single/all> <nazwa_kolumny>

Wyjaśnienie:

- `single / all` – w przypadku „`single`”, należy na wyjście wypisać tylko wartości w podanej kolumnie, a w „`all`” wypisać wartości wszystkich kolumn oddzielone znakiem tabulacji.
- `nazwa_kolumny` – wybiera kolumnę, po której należy posortować logi.

Wyjście

Wyjściem z programu są zestawy kolumn posortowane według wymagań poszczególnych zapytań. Każdy zestaw powinien zaczynać się od wypisania zapytania poprzedzonego znakiem dolara ($) oraz spacją, np.:

```
$ all id
```

Kolejne linie powinny zawierać dane wraz z nagłówkiem, posortowane wg wymogów zapytania. W przypadku, kiedy zapytanie zawiera kolumnę, której nie ma w zbiorze danych, zapytanie powinno zwrócić wiadomość:

```
invalid column name: <nazwa_kolumny>
```

Wymagania implementacyjne

1. Jedynymi możliwymi importami są import skanera wczytywania z klawiatury oraz `java.text.DecimalFormat`. Potrzebne może być także dodanie `.useLocale(Locale.GERMAN)` do instancji skanera w celu czytania liczb zmiennoprzecinkowych z przecinkiem oddzielającym część dziesiętną. Można wtedy zaimportować także `java.util.Locale`.
2. Algorytm QuickSort musi zostać zaimplementowany iteracyjnie, bez wykorzystania stosu oraz mieć średnią złożoność czasową rzędu O(n log n) oraz złożoność pamięciową O(1).
3. Jako typ danych tabeli należy przyjąć `float`.
4. Podczas wypisywania, dane w posortowanej tabeli należy formatować za pomocą kodu: `DecimalFormat format = new DecimalFormat("0.####");`
5. Na końcu kodu przesyłanego submitu proszę dopisać w formie komentarza własne dane wejściowe.
6. Przypominam o komentowaniu aplikacji w formie opisanej w punkcie 3 Regulaminu zaliczania programów na BaCy z roku 2020/2021.

Przykład danych

Wejście (dla czytelności tabulatory zostały zamienione czterema spacjami!):

```
1
5 3
Id    hour    minute    temp    pressure
1    11    25    35,5    20,3
2    13    10    20,2    25,14
3    12    30    15,03    50,92
3
all temp
single minute
single volume
```

Wyjście:

```
$ all temp
id    hour    minute    temp    pressure
3    12    30    15,03    50,92
2    13    10    20,2    25,14
1    11    25    35,5    20,3
$ single minute
minute
10
25
30
$ single volume
invalid column name: volume
```