Soalnya Susah Banget

Worksheet 6
Kamis, 29 November 2018

Deskripsi

Anda seperti kebanyakan namun tidak semua orang pasti memiliki banyak teman. Suatu hari teman anda yang bernama "***Nama Teman Anda Yang Sama Sekali Tidak Panjang***" mempunyai suatu persoalan dalam suatu mata kuliah yang agak sangat susah dan meminta bantuan anda untuk menyelesaikannya.

Teman anda sudah mengerti betul tentang persoalan kali ini. Persoalan kali ini akan berfokus tentang menyusun urutan pengerjaan soal-soal pada setiap orang. Urutan pengerjaan setiap orang akan dimodelkan ke dalam suatu array yang bertindak mirip seperti priority queue dengan isinya adalah soal-soal. Setiap soal akan memiliki suatu poin yang akan didapatkan, serta waktu pengerjaan yang dibutuhkan jika mengerjakan soal. Setiap soal hanya akan diberikan ke satu orang.

Terdengar mudah? Setiap orang memiliki urutan prioritas pengerjaan yang berbeda-beda. Yaitu, mengerjakan soal dengan poin tertinggi terlebih dahulu, atau mengerjakan soal dengan waktu pengerjaan tersingkat. Terlebih lagi, terdapat pertanyaan yang akan ditanyakan kepada kedua orang tersebut berbeda, yaitu:

- Jika orang tersebut memilih mengurutkan berdasarkan poin terbanyak, maka akan ditanyakan berapa **waktu** yang mereka butuhkan untuk minimal mendapatkan suatu poin tertentu.
- Jika orang tersebut memilih mengurutkan berdasarkan waktu tersingkat, maka akan ditanyakan berapa **poin** yang mereka dapatkan setelah minimal telah mengerjakan soal-soal selama waktu tertentu.

Anda yang merasa kasihan dengan teman anda yang bernama "***Nama Teman Anda Yang Sama Sekali Tidak Panjang***", akhirnya memutuskan untuk membantu dia. Tunjukan kemampuan anda sebagai seorang mahasiswa yang jago dengan membantu dia dengan membuat program yang menyelesaikan masalah tersebut.

Catatan: Asisten akan mengecek kode yang anda buat. Jika anda tidak melakukan implementasi priority queue menggunakan array/arraylist (langsung menggunakan priority queue dari java collection) akan langsung mendapatkan nilai 0 (nol).

Format Masukan

Baris pertama terdiri dari dua angka **N**,**M**.

N baris berikutnya berisi nama, kode urutan prioritas, dan suatu *nilai minimal* setiap orang. Nama setiap orang maksimal 10 karakter lowercase alfabet tanpa spasi.

Kode urutan prioritas hanya terdapat dua kemungkinan yaitu 1 atau 2, angka 1 berarti berprioritas untuk poin tertinggi terlebih dahulu, dan angka 2 berarti berprioritas untuk menyelesaikan waktu tersingkat terlebih dahulu. Setiap orang memiliki suatu nilai indeks, orang pertama memiliki indeks 0 dan orang terakhir memiliki indeks N-1, urutan input setiap orang sesuai dengan urutan indeks.

Nilai minimal adalah suatu nilai yang akan digunakan untuk menjawab pertanyaan yang telah disebutkan tadi:

- Jika orang tersebut memilih mengurutkan berdasarkan poin terbanyak, maka akan ditanyakan berapa **waktu** yang mereka butuhkan untuk minimal mendapatkan poin `<nilai_minimal>`.
- Jika orang tersebut memilih mengurutkan berdasarkan waktu tersingkat, maka akan ditanyakan berapa **poin** yang mereka dapatkan setelah minimal telah mengerjakan soal-soal selama `<nilai_minimal>`.

```
<nama> <kode_prioritas> <nilai_minimal> //berindeks-0
<nama> <kode_prioritas> <nilai_minimal> //berindeks-1, dst
```

M baris berikutnya berisi tentang tiga angka mengenai suatu soal. Angka pertama adalah indeks orang yang akan mengerjakan soal ini. Angka kedua adalah poin dari suatu soal dan angka ketiga adalah waktu pengerjaan suatu soal.

```
<indeks_seseorang> <poin> <waktu>
```

Format Keluaran

M baris untuk setiap orang berdasarkan pertanyaan diatas. Urutan nama sesuai dengan input juga berurutan naik dengan indeks orang. Jika tidak dapat mendapatkan nilai minimal keluarkanlah `-1`.

```
<nama>: <nilai_yang_diminta>
```

Batasan

- $1 \leq$ **N** ≤ 100
- $1 \leq$ **M** ≤ 1000
- $1 \leq$ **nilai_minimal** ≤ 10000
- $1 \leq$ **poin_soal** ≤ 1000
- $1 \leq$ **waktu_soal** ≤ 1000
- $0 \leq$ **indeks_seseorang** $\leq$ N-1
- $0 \leq$ **indeks_persoalan** $\leq$ M-1

Contoh 1

Masukan

```
2 4
siapa 1 10
gitu 2 20
0 5 10
0 10 10
1 19 19
1 20 20
```

Keluaran

```
siapa: 10
gitu: 39
```

Penjelasan

“Siapa” memiliki prioritas untuk mengerjakan persoalan dengan poin tertinggi terlebih dahulu, sedangkan “**Gitu”** memiliki prioritas untuk mengerjakan persoalan dengan waktu pengerjaan tercepat. “**Siapa”** memasukkan soal pertama dan kedua ke dalam rencana pengerjaan soalnya. “**Gitu”** memasukkan soal ketiga dan keempat ke dalam rencana pengerjaan soalnya. Untuk mencapai nilai minimal yaitu 10 poin, **“Siapa”** hanya harus mengerjakan soal kedua dengan waktu 10, sedangkan “**Gitu”** harus mengerjakan kedua soal untuk mendapat nilai minimal 20 waktu, dan mendapat 39 poin.

Contoh 2

Masukan

```
2 4
ferguso 1 100
ferguson 2 10
0 10 45
1 20 35
0 30 25
1 40 15
```

Keluaran

```
ferguso: -1
ferguson: 40
```

Penjelasan

Orang dengan nama **“Ferguso”** tidak dapat memenuhi nilai minimal seberapa banyak pun dia mengerjakan soal. “**Ferguson”** hanya perlu mengerjakan soal dengan poin 40 untuk memenuhi nilai minimal.

Template

```java
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.IOException;

import java.util.Comparator;
import java.util.ArrayList;
import java.util.Collections;

public class SoalSusah {
   public static void main(String[] args) throws IOException {
       BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));

       String inputs[] = reader.readLine().split(" ");

       int N = Integer.parseInt(inputs[0]);
       int M = Integer.parseInt(inputs[1]);

       // TODO: Implement this
       ArrayList<CustomPriorityQueue> prioritasPerorangan = new ArrayList<CustomPriorityQueue>();
       for (int i = 0; i < N; i++) {
           inputs = reader.readLine().split(" ");

       }

       // TODO: Implement this
       for (int i = 0; i < M; i++) {
           inputs = reader.readLine().split(" ");

       }

       // TODO: Implement this
       for (int i = 0; i < N; i++) {
           CustomPriorityQueue current = prioritasPerorangan.get(i);

           // TODO: Implement this
           // System.out.println(current.getNama()+": "+???));
       }

   }
}

class Persoalan {
   private int poin;
   private int waktu;

   public Persoalan(int poin, int waktu) {
       this.poin = poin;
       this.waktu = waktu;
   }

   public int getPoin() {
       return this.poin;
   }

   public int getWaktu() {
       return this.waktu;
   }
}

class SoalPoinComparator implements Comparator<Persoalan> {
```

```
    // TODO: Implement this
    public int compare(Persoalan p1, Persoalan p2) {

    }
}

class SoalWaktuComparator implements Comparator<Persoalan> {

    // TODO: Implement this
    public int compare(Persoalan p1, Persoalan p2) {

    }
}

// Implementasi PQ menggunakan unsorted array method
class CustomPriorityQueue {
    private ArrayList<Persoalan> array;
    private int size;
    private String nama;
    private int type;
    private int nilaiMinimal;

    public CustomPriorityQueue(String nama, int type, int nilaiMinimal) {
        this.array = new ArrayList<Persoalan>();
        this.size = 0;
        this.nama = nama;
        this.type = type;
        this.nilaiMinimal = nilaiMinimal;
    }

    public boolean empty(){
        return size == 0;
    }

    // TODO: Implement this
    public void insert (Persoalan soal) {
        // size++;
    }

    public Persoalan remove(){
         if (empty()) return null;

         Persoalan prioritas;
         if (type == 1) {
         prioritas = Collections.max(array, new SoalPoinComparator());
         } else {
         prioritas = Collections.max(array, new SoalWaktuComparator());
         }

         size--;
         array.remove(prioritas);
         return prioritas;
    }

    // TODO: Implement this
    public int countMinimalValue() {

    }

    public String getNama() {
        return this.nama;
```

```
    }
}
```

https://gist.github.com/sangbijaksana/f6e3bd2e78da7b223b89a15983766a1f

Peringatan

- Implementasi method *insert* boleh mengikuti dari slide, atau sumber lain (referensi harus dicantumkan). Akan tetapi, jika mengutip atau menggunakan code yang sudah ada, nilai WS ini maks 85 (walaupun di Aren nilai 100).
- Jika menulis code sendiri untuk method insert, baru bisa dapat 100.
- Jika menggunakan code dari sumber lain tanpa referensi, dianggap plagiat.
- Kolaborasi hanya dibolehkan dalam diskusi ide, bukan dalam menulis code. Menulis code harus dilakukan sendiri.