Started on	Thursday, 16 March 2023, 10:40 PM
State	Finished
Completed on	Thursday, 16 March 2023, 11:18 PM
Time taken	37 mins 34 secs
Marks	400.00/400.00
Grade	**100.00** out of 100.00

Question **1**

Correct

Mark 100.00 out of 100.00

Time limit	1 s
Memory limit	64 MB

Apakah kalian masih ingat dengan kelas Kompleks yang pernah kalian kerjakan pada praktikum pertama?

Karena sekarang waktunya belajar Java, terjemahkanlah kelas Kompleks di bawah dari C++ ke dalam Java!

Berikut adalah header dari kelas tersebut

```
#ifndef KOMPLEKS_H
#define KOMPLEKS_H

class Kompleks {
  public:
    // ctor tanpa parameter
    // inisialisasi seluruh koefisien dengan nilai 0
    Kompleks();

    // ctor dengan parameter
    Kompleks(int real, int imaginer);

    //mengembalikan bagian riil
    int getReal() const;

    // mengembalikan bagian imaginer
    int getImaginer() const;

    // mengisi bagian riil
    void setReal(int);

    // mengisi bagian imaginer
    void setImaginer(int);

    // operator overloading

    // operator+ untuk melakukan penjumlahan dengan rumus berikut
    // (a + bi) + (c + di) = (a+c) + (b+d)i
    friend Kompleks operator+ (const Kompleks&, const Kompleks&);

    // operator- untuk melakukan pengurangan dengan rumus berikut
    // (a + bi) - (c + di) = (a-c) + (b-d)i
    friend Kompleks operator- (const Kompleks&, const Kompleks&);

    // operator* untuk melakukan perkalian dengan rumus berikut
    // (a + bi)(c + di) = ac + bci + adi + bd i^2 = (ac-bd) + (bc+ad)i
    friend Kompleks operator* (const Kompleks&, const Kompleks&);

    // operator* untuk mengkalikan bilangan kompleks dengan konstanta
    // (a + bi)(c) = (ac) + (bc)i
    friend Kompleks operator* (const Kompleks&, const int);

    // operator* untuk mengkalikan bilangan kompleks dengan konstanta (sifat komutatif)
    friend Kompleks operator* (const int, const Kompleks&);

    // mengembalikan jumlah instance yang pernah dibuat
    static int countKompleksInstance();

    // mencetak bilangan kompleks ke layar, diakhiri dengan end-of-line
    // contoh:
    // 3+5i
    // 0
    // 3i
    // -3
    // -5-4i
    void print();

  private:
    static int n_kompleks;
    int real;
    int imaginer;
};

#endif
```

Namun, karena operator overloading tidak bisa diimplementasikan pada Java, maka terdapat beberapa perubahan pada kelas tersebut. Untuk setiap operator, implementasikan ke method-method berikut.

```
// operator+ untuk melakukan penjumlahan dengan rumus berikut
// (a + bi) + (c + di) = (a+c) + (b+d)i
public Kompleks plus(Kompleks b)
// operator- untuk melakukan pengurangan dengan rumus berikut
// (a + bi) - (c + di) = (a-c) + (b-d)i
public Kompleks minus(Kompleks b)
// operator* untuk melakukan perkalian dengan rumus berikut
// (a + bi)(c + di) = ac + bci + adi + bd i^2 = (ac-bd) + (bc+ad)i
public Kompleks multiply(Kompleks b)
// operator* untuk mengkalikan bilangan kompleks dengan konstanta
// (a + bi)(c) = (ac) + (bc)i
public Kompleks operator* (const int);
```

Kumpulkan file dengan nama **Kompleks.java**

Java 8

Kompleks.java

Score: 160

Blackbox

Score: 160

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	10	Accepted	0.30 sec, 28.27 MB
2	10	Accepted	0.38 sec, 27.85 MB
3	10	Accepted	0.25 sec, 28.52 MB
4	10	Accepted	0.33 sec, 28.38 MB
5	10	Accepted	0.30 sec, 27.89 MB
6	10	Accepted	0.36 sec, 30.07 MB
7	10	Accepted	0.18 sec, 28.45 MB
8	10	Accepted	0.33 sec, 30.30 MB
9	10	Accepted	0.49 sec, 28.44 MB
10	10	Accepted	0.52 sec, 27.82 MB
11	10	Accepted	0.16 sec, 28.11 MB
12	10	Accepted	0.12 sec, 28.68 MB
13	10	Accepted	0.10 sec, 28.38 MB
14	10	Accepted	0.12 sec, 30.81 MB
15	10	Accepted	0.15 sec, 28.61 MB
16	10	Accepted	0.47 sec, 28.93 MB

Question **2**

Correct

Mark 100.00 out of 100.00

Time limit	1 s
Memory limit	64 MB

Java sudah menyediakan beberapa interface untuk dapat diimplementasikan. Salah satunya adalah interface Comparable. Dengan mengimplementasikan interface Comparable, Anda dapat menggunakan library Java untuk sort, dan lain-lainnya.

Untuk mengimplemen interface Comparable, sebuah kelas harus mendefinisikan method compareTo(). Sebagai contoh, ini adalah kelas MataKuliah yang mengimplementasikan Comparable.

```
import java.lang.Comparable;

class MataKuliah implements Comparable<MataKuliah> {
  private float rating;

  public MataKuliah(float rating) {
    this.rating = rating;
  }

  public int compareTo(MataKuliah m) {
    // compareTo mengembalikan:
    // 0 bila this sama dengan m
    // 1 bila this lebih dari m
    // -1 bila this kurang dari m
    if (this.rating == m.rating) {
      return 0;
    } else if (this.rating > m.rating) {
      return 1;
    } else {
      return -1;
    }
  }
}
```

Di soal ini, Anda diminta mengubah kelas MataKuliah di atas. Anda perlu menambahkan atribut nama (String), kodeJurusan (int), dan tahunPengambilan (int). Lalu, buatlah juga method berikut:

- constructor yang menerima 4 parameter dengan urutan (String nama, int kodeJurusan, int tahunPengambilan, float rating)
- getter untuk tiap atribut, dengan nama getNama, getKodeJurusan, getTahunPengambilan, dan getRating
- method compareTo(MataKuliah m) yang membandingkan dua MataKuliah.

MataKuliah A disebut kurang dari MataKuliah B bila:

- kode jurusan MataKuliah A lebih kecil dari MataKuliah B
- kode jurusan MataKuliah A sama dengan MataKuliah B, tapi tahun pengambilannya lebih kecil
- kode jurusan dan tahun pengambilan MataKuliah A sama dengan MataKuliah B, tapi rating nya lebih kecil

MataKuliah A sama dengan MataKuliah B bila kode jurusan, tahun pengambilan, dan rating kedua MataKuliah bernilai sama

Selain itu, MataKuliah A disebut lebih dari MataKuliah B

Kumpulkan MataKuliah.java

Java 8

MataKuliah.java

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	9	Accepted	0.23 sec, 28.91 MB
2	9	Accepted	0.20 sec, 27.94 MB
3	9	Accepted	0.17 sec, 28.20 MB
4	9	Accepted	0.27 sec, 28.20 MB
5	9	Accepted	0.40 sec, 28.22 MB
6	9	Accepted	0.30 sec, 28.37 MB
7	9	Accepted	0.19 sec, 27.97 MB
8	9	Accepted	0.31 sec, 28.37 MB
9	9	Accepted	0.34 sec, 27.88 MB
10	9	Accepted	0.18 sec, 27.92 MB
11	10	Accepted	0.09 sec, 28.86 MB

Question **3**

Correct

Mark 100.00 out of 100.00

Time limit	1 s
Memory limit	64 MB

Mas Nopek merupakan seorang komedian yang membutuhkan tempat tinggal, namun Mas Nopek sedang galau karena dia juga sering bepergian untuk tur komedi, sehingga takut jika langsung membangun rumah diatas tanah akan jarang ditempati. Sehingga Mas Nopek memiliki alternatif lain untuk tinggal di karavan. Bantu Mas Nopek untuk menghitung dan mensimulasikan memilih tempat tinggal di rumah atau di karavan.

Tuliskan tiga buah kelas dimana dua kelas merupakan turunan dari kelas TempatTinggal. Kemudian salah satu kelas turunan mengimplementasi sebuah interface.

1. **Spesifikasi kelas TempatTinggal :**
    1. **Memiliki atribut luas (integer; luas bangunan dalam meter persegi) dan hargaBahanBangunan (double; harga per meter persegi)**
    2. **Mengandung method:**
        1. **hitungBiayaBangun : Cara implementasi method hitungBiayaBangun adalah : biayaBangun = luas * hargaBahanBangunan**
2. **Spesifikasi kelas Rumah yang merupakan turunan kelas TempatTinggal :**
    1. **Mempunyai dua buah konstruktor dengan parameter:**
        1. **luas dan hargaBahanBangunan**
        2. **Lebar tanah, panjang tanah, dan hargaBahanBangunan. Pada konstruktor ini luas tanah akan dihitung dengan cara luas = lebar x panjang tanah.**
    2. **Memiliki method hitungBiayaBangun yang menghasilkan biayaBangun (biayaBangun = luas * hargaBahanBangunan)**
    3. **Di beberapa daerah, untuk membangun rumah terkadang ada "ormas" yang meminta setoran, sehingga akan dibuat method hitungBiayaBangun yang memasukkan variabel setoran ormas sebagai perhitungan. Sehingga ada method hitungBiayaBangun yang memiliki input parameter biayaOrmas: double. Rumus perhitungan akan menjadi, biayaBangun = (luas * hargaBahanBangunan) + biayaOrmas**
    4. **untuk setter luas, gunakan dua varian:**
        1. **Varian dengan input luas bangunan**
        2. **Varian dengan input panjang dan lebar bangunan.**
3. **Spesifikasi interface Kendaraan :**
    1. **Mempunyai sebuah method yaitu hitungJarakTempuh**
4. **Spesifikasi kelas Karavan yang merupakan turunan kelas TempatTinggal dan akan mengimplement interface Kendaraan**
    1. **Dalam kelas Karavan terdapat atribut tambahan yaitu:**
        1. **bensin (float) : jumlah bensin yang ada di tanki dalam satuan liter**
        2. **pemakaianBensin (float) : jarak tempuh yang dapat dilakukan per liter**
    2. **Berikut ini merupakan spesifikasi penerapan method pada kelas ini:**
        1. **hitungBiayaBangun : BiayaBangun = (luas * hargaBahanBangunan) * 3**
        2. **hitungJarakTempuh : JarakTempuh = (bensin * pemakaianBensin)**

Tugas Anda : Tuliskan semua kelas dan interface yang didefinisikan dalam persoalan di atas dalam 1 file bernama TempatTinggal.java. Semua nama-nama yang digunakan (termasuk nama file) harus sama persis seperti yang diminta dalam soal, termasuk type case-nya.

Selektor untuk semua kelas yang terdefinisi pada soal ini ditentukan dengan gaya penulisan camelCase dengan format get<attribut> atau set<attribut> dengan <attribut> diganti dengan nama atribut ybs.

Java 8

TempatTinggal.java

Score: 140

Blackbox

Score: 140

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	10	Accepted	0.11 sec, 30.51 MB
2	10	Accepted	0.10 sec, 30.73 MB
3	10	Accepted	0.21 sec, 32.48 MB
4	10	Accepted	0.10 sec, 28.54 MB
5	10	Accepted	0.09 sec, 29.75 MB
6	10	Accepted	0.09 sec, 29.29 MB
7	10	Accepted	0.09 sec, 29.32 MB
8	10	Accepted	0.10 sec, 29.55 MB
9	10	Accepted	0.09 sec, 29.39 MB
10	10	Accepted	0.10 sec, 29.39 MB
11	10	Accepted	0.09 sec, 28.09 MB
12	10	Accepted	0.17 sec, 30.48 MB
13	10	Accepted	0.11 sec, 28.39 MB
14	10	Accepted	0.11 sec, 29.84 MB

Question **4**

Correct

Mark 100.00 out of 100.00

Time limit	1 s
Memory limit	64 MB

Di dalam tugas besar OOP kali ini, masih banyak dari kalian yang tidak menerapkan best practice OOP atau masih "prosedural" banget lohh xixi. Nah agar kalian semakin paham pattern OOP seperti apa silahkan implementasikan skenario di bawah ini ya ^_^.

Terdapat interface Calculable.java, static class Comparator.java, dan abstract class Card.java. Implementasikanlah class YellowCard yang diturunkan dari class Card di dalam YellowCard.java. Begitu juga dengan class GreenCard , RedCard, dan BlueCard (bikin sendiri gaada template :p dan jangan lupa tiap kelas dipisah file). Ketentuan lebih detailnya silahkan simak poin-poin di bawah ini:

1. Class-class tersebut akan merepresentasikan kartu yang memiliki sebuah angka (anggap tidak ada batasan nilai angka).
2. Berat dari setiap kartu ditentukan dari angka dulu, baru warna. Dengan detail berat warna, Merah > Kuning > Biru > Hijau. Contoh, kartu berwarna **hijau** dengan angka **15** akan berbobot lebih besar dibanding kartu **kuning** dengan angka **7**. Namun, kartu berwarna **hijau** dengan angka **15** akan lebih kecil dibanding kartu **merah** dengan angka **15**.
3. Setiap kartu dapat memprint informasi (method printInfo()) dengan ketentuan sebagai berikut:
    1. Kartu biru => "Kartu Biru: <suatu_angka>".
    2. Kartu merah => "Kartu Merah: <suatu_angka>"
    3. Kartu hijau => "Kartu Hijau: <suatu_angka>"
    4. Kartu kuning => "Kartu Kuning: <suatu_angka>"
4. Silahkan implementasikan sendiri bagaimana rumus menghitung value agar kasus di atas terpenuhi :D.

*input output di bawah ini hanya contoh, silahkan fokus dengan implementasi yang diperintahkan pada soal tidak perlu memikirkan bagaimana perintah dijalankan dari command line.

**file yang perlu dikumpulkan cukup YellowCard.java, GreenCard.java, RedCard.java, dan BlueCard.java. Satukan keempat file tersebut ke dalam satu zip dan upload zip tersebut.

***untuk setiap operasi yang membutuhkan keluaran di command line, gunakan fungsi 'System.out.println()'

Input	Output	Keterangan
Kartu 1: Merah 9 Kartu 2: Kuning 10 isObj1BiggerThanObj2(Kartu 1, Kartu 2)	False	Karena angkanya lebih besar angka kartu kedua.
Kartu 1: Merah 9 Kartu 2: Kuning 9 isObj1EqualToObj2(Kartu 1, Kartu 2)	False	Karena walaupun kedua kartu tersebut memiliki angka yang sama tetapi warnanya berbeda.
Kartu 1: Merah 9 Kartu 2: Kuning 9 isObj1LowerThanObj2(Kartu 1, Kartu 2)	False	Karena walaupun kedua kartu tersebut memiliki angka yang sama tetapi Merah > Kuning.
Kartu 1: Merah 9 //call printInfo	Kartu Merah: 9.0	

Java 8

YellowCard.zip

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	10	Accepted	0.16 sec, 29.88 MB
2	15	Accepted	0.14 sec, 27.98 MB
3	15	Accepted	0.15 sec, 29.09 MB
4	15	Accepted	0.20 sec, 26.27 MB
5	15	Accepted	0.34 sec, 28.52 MB
6	15	Accepted	0.13 sec, 28.39 MB
7	15	Accepted	0.12 sec, 30.21 MB

◄ Pra Responsi 4 (Mystery Box)

Jump to...

Slide Responsi 4 ►