Started on	Saturday, 18 February 2023, 3:13 PM
State	Finished
Completed on	Saturday, 18 February 2023, 3:46 PM
Time taken	33 mins 14 secs
Grade	**600.00** out of 600.00 (**100**%)

Question **1**

Correct

Mark 100.00 out of 100.00

Time limit	1 s
Memory limit	64 MB

Interface - Vehicle

Anggaplah Anda sedang membuat program sederhana untuk melacak berbagai jenis kendaraan, seperti mobil, sepeda motor, dan sepeda. Setiap jenis kendaraan memiliki karakteristik tertentu

Untuk mengatasi masalah ini menggunakan prinsip OOP, Anda dapat membuat interface untuk semua kendaraan yang disebut "Vehicle". Interface ini akan mendeklarasikan method yang harus dimiliki setiap kendaraan yaitu:

Method Name	Parameter - Type	Return Type
getNumberOfWheels	Empty - Empty	int
getMaxSpeed	Empty - Empty	int
getFuelType	Empty - Empty	String
start	Empty - Empty	void
stop	Empty - Empty	void
turn	direction - String	void

Dengan menggunakan interface seperti ini, Anda dapat memastikan bahwa semua kendaraan memiliki karakteristik dan perilaku yang diperlukan, meskipun implementasi spesifik untuk setiap kendaraan berbeda.

Submitlah file **Vehicle.java** yang merupakan interface berisi method-method pada tabel diatas

Java 8

Vehicle.java

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	14	Accepted	0.07 sec, 28.96 MB
2	14	Accepted	0.07 sec, 28.94 MB
3	14	Accepted	0.06 sec, 28.29 MB
4	14	Accepted	0.07 sec, 28.75 MB
5	14	Accepted	0.07 sec, 28.45 MB
6	14	Accepted	0.07 sec, 27.79 MB
7	16	Accepted	0.07 sec, 28.93 MB

Question **2**

Correct

Mark 100.00 out of 100.00

Time limit	1 s
Memory limit	64 MB

Grouped Interface dan Implementasi

Buatlah sebuah class bernama `Point`, yang digunakan sebagai blueprint untuk lokasi. Class `Point` berisi atribut `latitude` dan `longitude`, keduanya memiliki tipe data `double` dan access modifier `private`. Kedua atribut `latitude` dan `longitude` diinisialisai pada konstruktor. Buatlah juga method getter untuk kedua atribut bernama `getLatitude` dan `getLongitude`.

Setelah itu, buatlah sebuah interface lain bernama `Trackable`, yang berfungsi sebagai pemberi informasi tambahan tentang kendaraan, seperti nomor plat dan posisi GPS. Interface ini akan mendeklarasikan beberapa method, diantaranya:

Method Name	Parameter - Type	Return Type
getPlateNumber	Empty - Empty	String
getGPSPosition	Empty - Empty	Point

Terakhir, buatlah interface lain bernama `TrackableVehicle`, yang merupakan Grouped Interface dari interface `Vehicle` dan `Trackable`.

Buatlah `Point.java`, `Trackable.java`, dan `TrackableVehicle.java` tersebut, dan kumpulkan dalam satu ZIP *file* **(tanpa dibuat folder)** dengan nama bebas. Submit file zip tersebut.

Java 8

Vehicle.zip

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	16	Accepted	0.07 sec, 30.42 MB
2	16	Accepted	0.07 sec, 28.97 MB
3	16	Accepted	0.07 sec, 29.14 MB
4	16	Accepted	0.07 sec, 28.75 MB
5	16	Accepted	0.07 sec, 30.37 MB
6	20	Accepted	0.07 sec, 28.79 MB

Question **3**

Correct

Mark 100.00 out of 100.00

Time limit	1 s
Memory limit	64 MB

Interface - Police Car

Buatlah sebuah class bernama `PoliceCar` yang mengimplementasikan interface `TrackableVehicle` yang telah dibuat pada soal nomor 2 yang memiliki atribut sebagai berikut

Nama Atribut	Tipe
plateNumber	String
gpsPosition	Point
speed	Integer
direction	String
wheel	Integer
fuelType	String
started	Boolean

Ketika kelas `PoliceCar` baru dibentuk, atribut `direction` akan secara otomatis bernilai "North" dan `started` akan secara otomatis bernilai false

Kelas akan berisikan **konstruktor** dan **getter** untuk semua atributnya sesuai dengan nama method pada interfacenya. Adapun penjelasan method tambahan sebagai berikut

Metode	Parameter	Return Type	Penjelasan
start	Empty	void	Mengganti started menjadi true
stop	Empty	void	Mengganti started menjadi false
turn	direction	void	Mengganti direction menjadi nilai pada parameter

Submit file bernama `PoliceCar.java`

Java 8

PoliceCar.java

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	20	Accepted	0.07 sec, 28.37 MB
2	20	Accepted	0.07 sec, 26.78 MB
3	20	Accepted	0.07 sec, 28.90 MB
4	20	Accepted	0.07 sec, 28.97 MB
5	20	Accepted	0.07 sec, 28.88 MB

Question **4**

Correct

Mark 100.00 out of 100.00

Time limit	1 s
Memory limit	64 MB

Interface - Calculator

Anggaplah Anda sedang membuat program sederhana untuk melacak berbagai jenis kalkulator, seperti scientific calculator, financial calculator, dll. Setiap jenis kalkulator memiliki karakteristik tertentu.

Untuk mengatasi masalah ini menggunakan prinsip OOP, Anda dapat membuat interface untuk semua kalkulator yang disebut "Calculator". Interface ini akan mendeklarasikan method yang harus dimiliki setiap calculator yaitu:

Method Name	Parameter - Type	Return Type
add	a,b - int,int	int
substract	a,b - int,int	int
multiply	a,b - int,int	int
divide	a,b - int,int	double
start	Empty - Empty	void
stop	Empty - Empty	void
checkBattery	Empty - Empty	int

Dengan menggunakan interface seperti ini, Anda dapat memastikan bahwa semua kalkulator memiliki karakteristik dan perilaku yang diperlukan, meskipun implementasi spesifik untuk setiap kalkulator berbeda.

Submitlah file **Calculator.java** yang merupakan interface berisi method-method pada tabel diatas

Java 8

Calculator.java

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	12	Accepted	0.07 sec, 27.99 MB
2	12	Accepted	0.07 sec, 27.80 MB
3	12	Accepted	0.07 sec, 28.91 MB
4	12	Accepted	0.07 sec, 29.85 MB
5	12	Accepted	0.07 sec, 30.60 MB
6	12	Accepted	0.06 sec, 28.15 MB
7	12	Accepted	0.07 sec, 28.17 MB
8	16	Accepted	0.07 sec, 28.07 MB

Question **5**

Correct

Mark 100.00 out of 100.00

Time limit	1 s
Memory limit	64 MB

Grouped Interface dan Implementasi

Buatlah sebuah class bernama `Point`, yang digunakan sebagai blueprint untuk point. Class `Point` berisi atribut `x` dan `y`, keduanya memiliki tipe data `int` dan access modifier `private`. Kedua atribut `x` dan `y` diinisialisai pada konstruktor. Buatlah juga method getter untuk kedua atribut bernama `getX` dan `getY` serta setter `setX` dan `setY`.

Setelah itu, buatlah sebuah interface lain bernama `GraphUpgrade`, yang berfungsi sebagai upgrade calculator dengam method-method untuk kalkulasi Graph, diantaranya:

Method Name	Parameter - Type	Return Type
shiftX	shiftCount - int	Void
shiftY	shiftCount - int	Void
distance	x - int, y - int	double

Terakhir, buatlah interface lain bernama `GraphCalculator`, yang merupakan Grouped Interface dari interface `Calculator` dan `GraphUpgrade`.

Buatlah `Point.java`, `GraphUpgrade.java`, dan `GraphCalculator.java` tersebut, dan kumpulkan dalam satu ZIP *file* **(tanpa dibuat folder)** dengan nama bebas. Submit file zip tersebut.

Java 8

Calculator.zip

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	16	Accepted	0.48 sec, 28.45 MB
2	16	Accepted	0.75 sec, 27.83 MB
3	16	Accepted	0.63 sec, 29.15 MB
4	16	Accepted	0.42 sec, 28.26 MB
5	16	Accepted	0.41 sec, 26.96 MB
6	20	Accepted	0.49 sec, 29.00 MB

Question **6**

Correct

Mark 100.00 out of 100.00

Time limit	1 s
Memory limit	64 MB

Buatlah sebuah class bernama `CasioGraph` yang mengimplementasikan interface `GraphCalculator` yang telah dibuat pada soal nomor 2 yang memiliki atribut sebagai berikut

Nama Atribut	**Tipe**
point	Point
batteryLevel	Integer
status	Boolean

Ketika kalkulator hidup, maka status akan bernilai true. Apabila kalkulator tidak hidup, maka status bernilai false.

Ketika kelas `CasioGraph` baru dibentuk, atribut `batteryLevel` akan secara otomatis bernilai 100 dan status yang otomatis bernilai false. Pada konstruktor, juga akan diminta 2 buah parameter x dan y untuk membuat point.

Kelas akan berisikan **konstruktor** dan **getter** untuk semua atributnya sesuai dengan interface yang membangunnya. Adapun penjelasan method tambahan sebagai berikut

Metode	**Parameter**	**Return Type**	**Penjelasan**
add	Int, Int	Int	Menambahkan nilai kedua parameter
substract	Int, Int	Int	Mengurangi nilai parameter pertama sebesar parameter kedua
multiply	Int, Int	Int	Mengalikan nilai kedua parameter
divide	Int, Int	Double	Membagi nilai parameter pertama dengan parameter kedua
start	Empty	void	Menghidupkan kalkulator. Apabila battery = 0, maka kalkulator tidak dapat hidup
stop	Empty	void	Mematikan kalkulator
checkBattery	Empty	Int	Memeriksa battery kalkulator
shiftX	Int	void	Menambahkan nilai X pada point sebesar parameter
shiftY	Int	void	Menambahkan nilai Y pada point sebesar parameter
distance	Int, Int	Double	Menghitung jarak titik pada kelas dengan titik pada parameter dengan rumus V(x1 - x2)^2 + (y1 - y2)^2
getStatus	Empty	Boolean	Mengembalikan status kalkulator
getPoint	Empty	Point	Mengembalikan nilai point yang disimpan oleh kalkulator
charge	Empty	void	Menambahkan batteryLevel hingga 100 dan secara otomatis menyalakan kalkulator

Perlu diperhatikan bahwa setiap menggunakan kalkulator (methods add, substract, multiply, divide, shiftX, shiftY, dan distance) battery kalkulator akan turun sebanyak 10. Apabila kalkulator tidak hidup, maka satu satunya method yang dapat digunakan adalah charge, start, stop, checkBattery, getStatus, dan getPoint. Method lain akan secara otomatis mengembalikan nilai -1 dan metods shiftX dan shiftY tidak akan merubah state kelas.

Submit file bernama `CasioGraph.java`

Java 8

CasioGraph.java

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	15	Accepted	0.07 sec, 28.91 MB
2	15	Accepted	0.07 sec, 28.79 MB
3	30	Accepted	0.07 sec, 28.50 MB
4	40	Accepted	0.07 sec, 28.68 MB

◄ Feedback Praktikum 3

Jump to...