Merrifield Preparatory School & College

SBA Object Oriented Programming Practical Test 2023

Time: 50 mins
Marks: 40

Examiner: M. Boon
Moderator: D. Emery

SCENARIO

RaceFit is a company that makes running shoes for runners. To promote its brand, the company organizes running races in cities around the country. It needs an application that can store the data of upcoming races and completed races.

The application will use two object classes:
1.The **Race** class will store the information for an upcoming race.
2.The **CompletedRace** class will store the information for a completed race.

A **Race** object contains the following information:

- name : the name of the race
- location : the location of the race
- raceDateTime : the date and time that the race will take place
- distance : the distance that must be run to complete the race

An example **Race** object:

- Name: Colour Run
- Location: Nahoon East London
- Date and Time: 2023-05-25T08:30
- Distance (km): 5

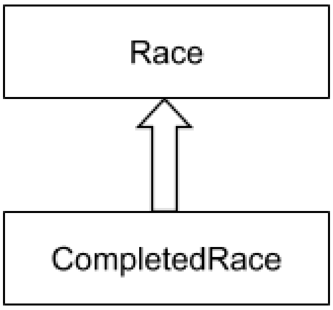
A **CompletedRace** contains all of the information of a **Race** object, and also includes the following:

- winners[] : a list of all winners of a particular race. A race might have winners in different age groups e.g. a junior and senior category. Each row in the array will contain text about a single winner including their age group, name and finish time.
- finishers: the number of participants that finished the race.

An example **CompletedRace** object:

- Name: Colour Run
- Location: Nahoon East London
- Date and Time: 2023-05-27T08:30
- Distance (km): 5
- Winners: Junior, Moleboheng Leeto, 18:55
  Senior, Michael Maruza, 19:12
- Finishers: 5340

You will need to create two object classes including the parent **Race** class and the child **CompletedRace** class as shown in the diagram below:



QUESTION 1

1.1 Create a simple user interface named **RaceEventUI**.

(1)

[1]

QUESTION 2

Use the class diagram below to create a class called **Race**. This class will be used to create objects to store the details of a race. The diagram indicates the required fields and methods.

```
Race
----------------------------------------------------------
Fields:
- name : string
- location : string
- raceDateTime : date time
- distance : real
----------------------------------------------------------
Methods:
+ Constructor(inNM : string, inLOC : string,
              inDT : date time, inDST : real)
+ getName() : string
+ getDateTime() : date time
+ formatDate() : string
+ formatDistance(system : string) : string
+ toString() : string
```

2.1 Create a new class called **Race** with name, location, raceDateTime and distance fields as indicated above.

(4)

2.2 Code a constructor method for the class that accepts a string **inNM** representing the name field, a string **inLOC** representing the location field, a date time **inDT** representing the raceDateTime field and a real number **inDST** that will represent the distance field. Use these parameters to assign the values for the fields of the class.

(3)

2.3 Create **accessor/getter** methods for the **name** and **raceDateTime** fields of the class.

(2)

2.4 Code a method named **formatDate** that will return a string containing the **raceDateTime** formatted as follows:

DayName<space>Date<space>MonthName<space>
Year<space>-<space>Time

For example:

```
Sat 27 May 2023 - 08:30
```

(2)

2.5 Code a method named **formatDistance** that will return a string containing the **distance** in either kilometres or miles. The formatDistance method will accept a string parameter ("metric" or "imperial") and then convert the distance to that measurement system accordingly.

If the parameter is "metric" then the distance is already in kilometres and no conversion is necessary. Add " kilometres" after the distance.

For example, if the distance is 10 the "metric" output will be:

```
10 kilometres
```

If the parameter is "imperial" then the distance needs to be converted to miles. The formula for converting kilometres to miles is to divide the kilometres by 1.609.

Round the conversion to 2 decimal places and add " miles" after the distance.

For example, if the distance is 10 the "imperial" output will be:

```
6.22 miles
```

(6)

2.6 Add a **toString** method to the class that will return a string containing all the information of the class in the following format (use **imperial** for the distance):

Name<space>(formatted distance in imperial)<space>-<space>
location<space>-<space>formatted date<space>start

For example,

```
Colour Run (6.22 miles) - Nahoon East London - Sat 27 May
2023 - 08:30 start
```

(3)

[20]

QUESTION 3

Use the class diagram below to create a class called **CompletedRace**. This class will inherit from the **Race** class and will be used to create objects that will store the details of a completed race. The diagram below indicates required fields and methods.

```
CompletedRace
-----------------------------------------------------------
Fields:
- raceWinners : string[]
- finishers : int
-----------------------------------------------------------
Methods:
+ Constructor(inNM : string, inLOC : string,
              inDT : date time, inDST : real,
              inRW : string[], inF : integer)
+ formatWinners() : string
+ toString() : string
```

3.1 Write code to create a new class called **CompletedRace** that extends the **Race** class.

(1)

3.2 Add a **raceWinners** field and a **finishers** field as indicated in the class diagram.

(2)

3.3 Code a constructor method that will initialise the fields **name**, **location**, **raceDateTime** and **distance** of the **Race** parent class and the **raceWinners** and **finishers** of the **CompletedRace** child class.

(4)

3.4 Code a method named **formatWinners** that will return a string containing all data from the **raceWinners** array.

The string stored in each element of the array is as follows:

Age group, winner name, race time

For example:

Junior, Moloeboheng Leeto, 18:55

Format the string as follows:

Age group<space>winner:<tab>winner name, race time<new line>

For example, in a raceWinners array containing two rows the output would appear as follows:

```
Junior winner: Moleboheng Leeto, 18:55
Senior winner: Michael Maruza, 19:12
```

(4)

3.5 Code a **toString** method that will override the parent class's **toString** method. It must add the finishers, and the raceWinners (using the **formatWinners** method in Question 3.4) to the string returned by the **toString** method of the **Race** class in the following format:

Name<space>(formatted distance *in imperial*)<space>-<space>
location<space>-<space>formatted date<space>start<new line>
Finishers:<tab>finishers<new line>
raceWinners formatted

For example,

```
Colour Run (6.22 miles) - Nahoon East London - Sat 27 May
2023 - 08:30 start
Finishers:     5340
Junior winner: Moleboheng Leeto, 18:55
Senior winner: Michael Maruza, 19:12
```

(2)

[13]

QUESTION 4

4. 1 Create a 1D string array named **rw1** that contains the following information:

```
Junior, Moleboheng Leeto, 18:55
Senior, Michael Maruza, 19:12
```

Create a 1D string array named **rw2** that contains the following information:

```
Junior, Ashwell Clooco, 28:52
Senior, Tayla Kavanagh, 24:17
```

(1)

4.2 Create a **CompletedRace** object named **cr1** using the **rw1** array and the following data:

```
Colour Run,Nahoon East London,
2023-05-25T08:30,10,rw1,5340
```

Create a **CompletedRace** object named **cr2** using the rw1 array and the following data:

```
Beach Run,Gonubie East London,
2023-03-18T10:00,5,rw2,9731
```

(1)

4.3 Write code to output the details of **cr1** and **cr2** as follows:

```
Colour Run (6.22 miles) - Nahoon East London - Sat 27 May
2023 - 08:30 start
Finishers:     5340
Junior winner: Moleboheng Leeto, 18:55
Senior winner: Michael Maruza, 19:12

Beach Run (3.11 miles) - Gonubie East London - Sat 18
March 2023 - 10:00 start
Finishers:     9731
Junior winner: Ashwell Clooco, 28:52
Senior winner: Tayla Kavanagh, 24:17
```

(1)

4.6 Write code to determine if the **cr1** race happens before or after the **cr2** race.

If the **cr1** race happens first, use the name fields from **cr1** and **cr2** to output the following:

```
The Colour Run will take place before the Beach Run.
```

If the **cr2** race happens first, use the name fields from **cr1** and **cr2** to output the following:

```
The Beach Run will take place before the Colour Run.
```

(3)

[6]

Total Marks: 40