IB Diploma Program Computer Science

Semester 2 Internal Test

Name _____

Part A - Option D Questions
Part B - N/A

26

PART A - Option D : Recommended Time 50 Mins

A Team has 5 Players. The UML diagrams for the Team and Player classes are shown below:

Player
- String name - String position - int pointsScored
+ Player(String, String) + getName():String + getPosition():String + getPoints():int + setPoints(int):void

Team
- String name - Player[] players - int totalPoints
+ Team(String) + calcPoints():int + addPoints(String, int):void + addPlayer(Player):void + getPlayers(): Player[ ]

Note: when a `Player` is constructed, `pointsScored` is assigned 0.

a) Complete the UML diagram by showing the relationship between the 2 classes. [1]

b) The `setPoints` method of the `Player` class takes an integer as a parameter and adds it to the current `pointsScored` for that player. Write code to implement the `setPoints` method of the `Player` class.

[2]

c) the `setPoints` method of the `Player` class is a mutator method. Explain the purpose of a mutator method.

[1]

d) A main driver program runs in real time and allows a user to enter the name of a player and the number of points the player scored. The program then finds the player in the team and assigns the points to that player. Here is a sample driver program:

```
Team t1 = new Team(”Dodgers”);

t1.addPlayer(”Jessie”);
t1.addPlayer(”Karen”);
t1.addPlayer(”Lee”);
t1.addPlayer(”Noel”);
t1.addPlayer(”Barley”);

while(gamePlaying){

     //code to enter a player name not shown
     //code to enter the player’s points not shown
     t1.addPoints(playerName, playerPoints)

}
```

[1]

The `addPoints` method of the Team class searches the `Player` array and when it finds the correct `Player`, it assigns the points to that `Player`.
Write code to implement the `addPoints` method of the `Team` class.

[5]

e) A `Defender` IS-A `Player`!
Write the entire `Defender` class, including any necessary accessor/mutator methods.

[3]

f) An `Attacker` is also a `Player`. However, an `Attacker` has additional property that other Players don't have. An `Attacker` has an `aggressionIndex` which is a number between 1 and 10 inclusive. When constructing a new `Attacker`, this value is set to 5. It can be changed during the game.

Write the entire `Attacker` class, including any necessary accessor/mutator methods.

[5]

g) Outline one advantage of using inheritance as a strategy in programming.

[2]

h) A Game consists of two Teams. Here is the entire Game class:

```
public class Game {

      private Team[] teams;

      public Game(Team t1, Team t2) {
            teams = new Team[2];
            teams[0] = t1;
            teams[1] = t2;
      }

      public Team getTeam(int index) {
            return teams[index];
      }

      public void displayScore() {
            int t1Score = 0;
            int t2Score = 0;

            t1Score = teams[0].calcPoints();
            t2Score = teams[1].calcPoints();

            System.out.println("Team1 "+t1Score+" "+t2Score+" Team2");
      }

      public String getHighestScorer(){
            //yet to be implemented
      }
}
```

Implement the `getHighestScorer()` method of the `Game` class. It returns the name of the player in the game with the highest score.
pre-condition: both teams have 5 players
post-condition: if there are multiple players with the same highest score, only 1 needs to be returned by the method