**CS 1302 – Sample Test 5 Handout**

**For problem 1**

Consider the class diagram below. **You will write only the (complete) *Game* class.** However, you will need some of the methods defined in the *Player* class.



Some notes about the *Game* class:

1. The *Game* class maintains a map of *Player* objects where the key is the *id*.
2. *addPlayer* – accepts a *Player* object and adds it to the map, provided there doesn’t already exist a player with the same *id*, and returns *true*. If the *id* already exists, then return *false.*
3. *getPlayer* – accepts an *id* and returns the corresponding player if they exist; otherwise, it returns *null.*
4. *removePlayer* – accepts an *id* and removes and returns the corresponding player (assume the room exists)
5. *lowestLevelPlayer* – returns the *Player* with the lowest level.
6. *playersBelowList* – accepts a *level* and returns a list of players at or below *level.*
7. *playersBelowMap* – accepts a *level* and returns a map of players at or below *level*, where *id* is the key.
8. *idsBetween* – accepts two integers, *from* and *to* and returns a list of *id*s of the players whose *id* is between *from* and *to*, inclusive.



|  |  |  |
| --- | --- | --- |
| **CS 1302 – Sample Test 5** | **Name:** |  |
| No electronics, notes, smart watches, *etc.* allowed. |  | Print: FirstName LastName |

1. (55 points) Write the entire *Game* class (see the handout provided)
2. (10 points) Pick a problem from: (a) Ch 9, Sec 2.1, Exercises 1-4. (b) Ch 9, Sec 7.1, Exercises 5, 6, 9
3. (10 points) Pick another problem from: (a) Ch 9, Sec 2.1, Exercises 1-4. (b) Ch 9, Sec 7.1, Exercises 5, 6, 9

There will be 2 or 3 recursion problems chosen from different types:

1. (10 points) Pick a problem from: (a) Ch 10, examples in Sections 4.1-4.5, (b) Ch 10, Sec 4.6, Exercises 3-5.
2. (10 points) Pick a problem from Ch 10, examples in Sections 5.2-5.3
3. (10 points) Pick a problem from: (a) Ch 10, examples in Sections 6.1-6.3, or (b) Ch 10, Sec 6.4, Exercise 11
4. (10 points) Pick a problem from: (a) Ch 10, examples in Sections 7.1-7.3, or (b) Ch 10, Sec 7.4, Exercises 12-13
5. (10 points total) Given the skeleton of a binary search method and recursive helper method, that searches for an integer in an array (or ArrayList) of integers, fill in the blanks of the missing code. **The best way to study for this is to be able to write the code for the example in Ch 10, Sec 8.5.**
6. (10 points total) Given the skeleton of a binary search method and recursive helper method, that searches for an *Employee* in an array (or ArrayList) of *Employee* objects, given that the *Employee* class implements the *Comparable* interface, fill in the blanks of the missing code. **The best way to study for this is to study the changes we made to example in Ch 10, Sec 8.5, when I presented a similar problem in class. Hint: the main thing you need to change is:** key < vals[mid] and key > vals[mid].