COMP20010

Data Structures and Algorithms I

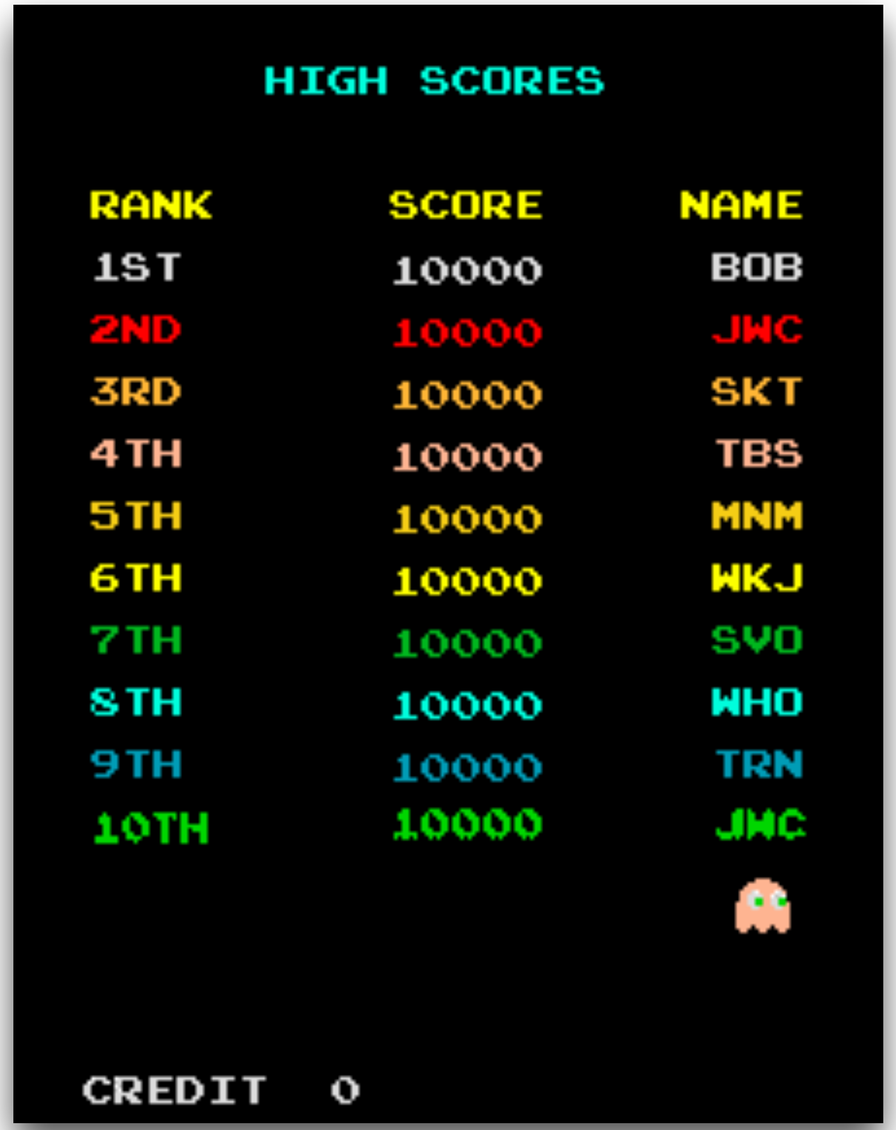
Dr. Aonghus Lawlor
aonghus.lawlor@ucd.ie

Tutorial

The starter code is on github

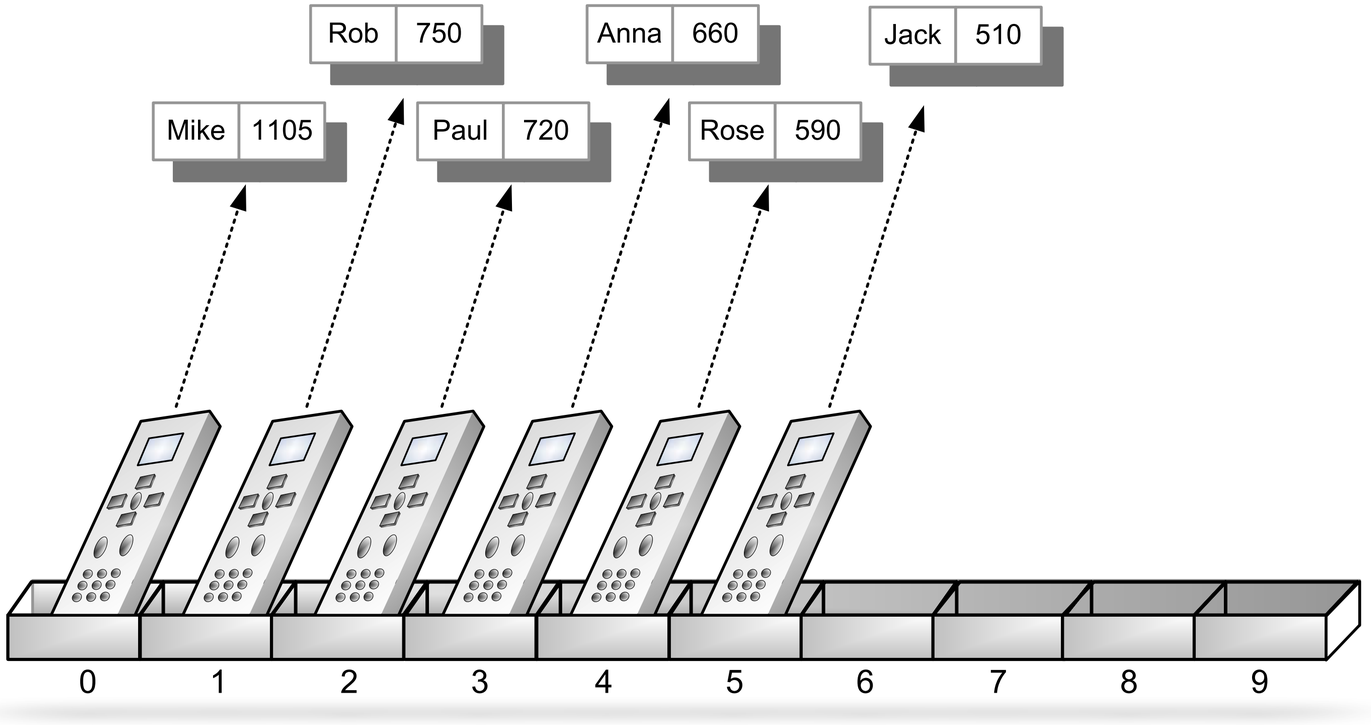
Game Scoreboard

- Store high score entries for a game in an array
- Fixed number of entries we wish to store: **`maxEntries`**
- Store the entries sorted by their (integer) score value (highest to lowest).



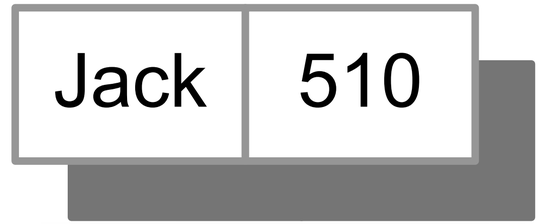
Game Scoreboard

- Fixed array size 10
- Storing 6 game entries
- Other elements are null
- Each game entry has a name and a score



Class Design

```
public class GameEntry {
   /** name of the person earning this score */
   protected String name;

   /** the score value */
   protected int score;

   /** Constructor to create a game entry */
   public GameEntry(String name, int score)…

   /** Retrieves the name field */
   public String getName() …

   /** Retrieves the score field */
   public int getScore() …

   public void setName(String name) …

   public void setScore(int score)…

   public String toString()…
}
```



Class Design

```
public class ScoreBoard {
   private int maxEntries;
   private int numEntries; // number of actual entries
   private GameEntry[] board; // array of game entries (name and scores)

   public ScoreBoard(int capacity) {
      // TODO
   }

   /** Attempts to add a new score to the collection (if it is high enough). */
   public void add(GameEntry e) {
      // TODO
   }

   /** Attempts to remove an existing score from the collection */
   public GameEntry remove(int i) throws IndexOutOfBoundsException {
      // TODO
   }

   public String toString() {
      // TODO
   }
```

constructor

Add new entries to the scoreboard

Remove entry at i

String representation of scoreboard

Scoreboard

```
public void add(GameEntry e) {
```

- Insert a GameEntry into the collection of high scores
- If the collection is full then ***e*** is added only if its score is higher than the lowest score.
- The GameEntry ***e*** should be inserted in the correct position according to its score

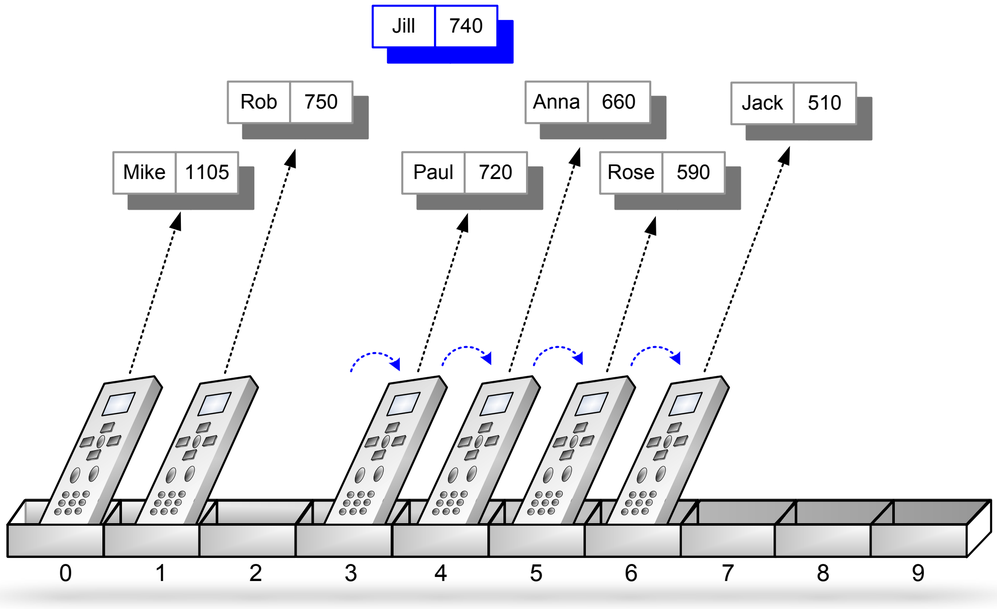
Scoreboard

```
public void add(GameEntry e) {
```

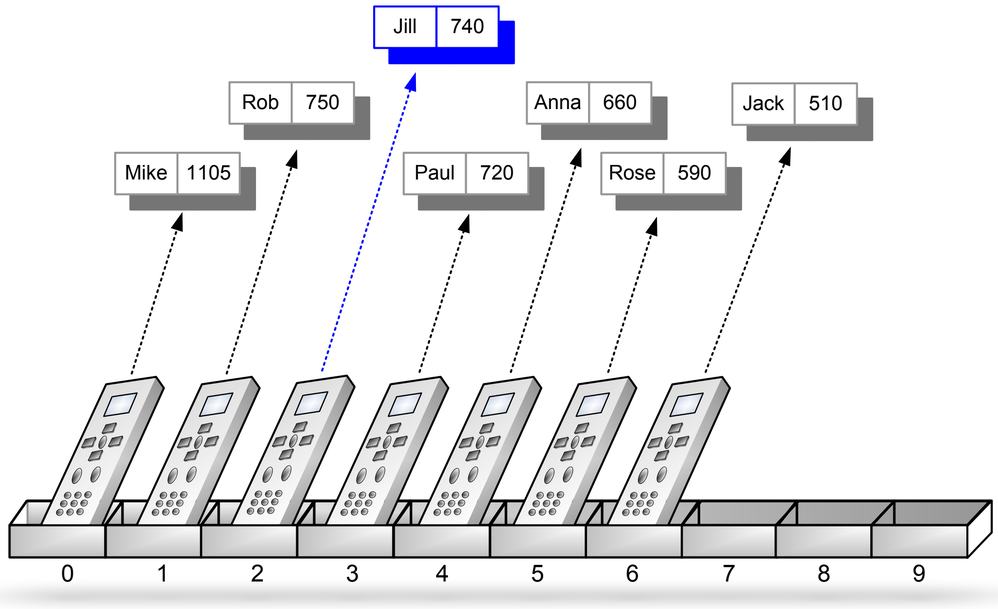
- Insert a GameEntry into the collection of high scores
- If the collection is full then ***e*** is added only if its score is higher than the lowest score.
- The GameEntry ***e*** should be inserted in the correct position according to its score

Scoreboard

```
public void add(GameEntry e) {
```



To make room for a new entry we move the existing entries to the right by one

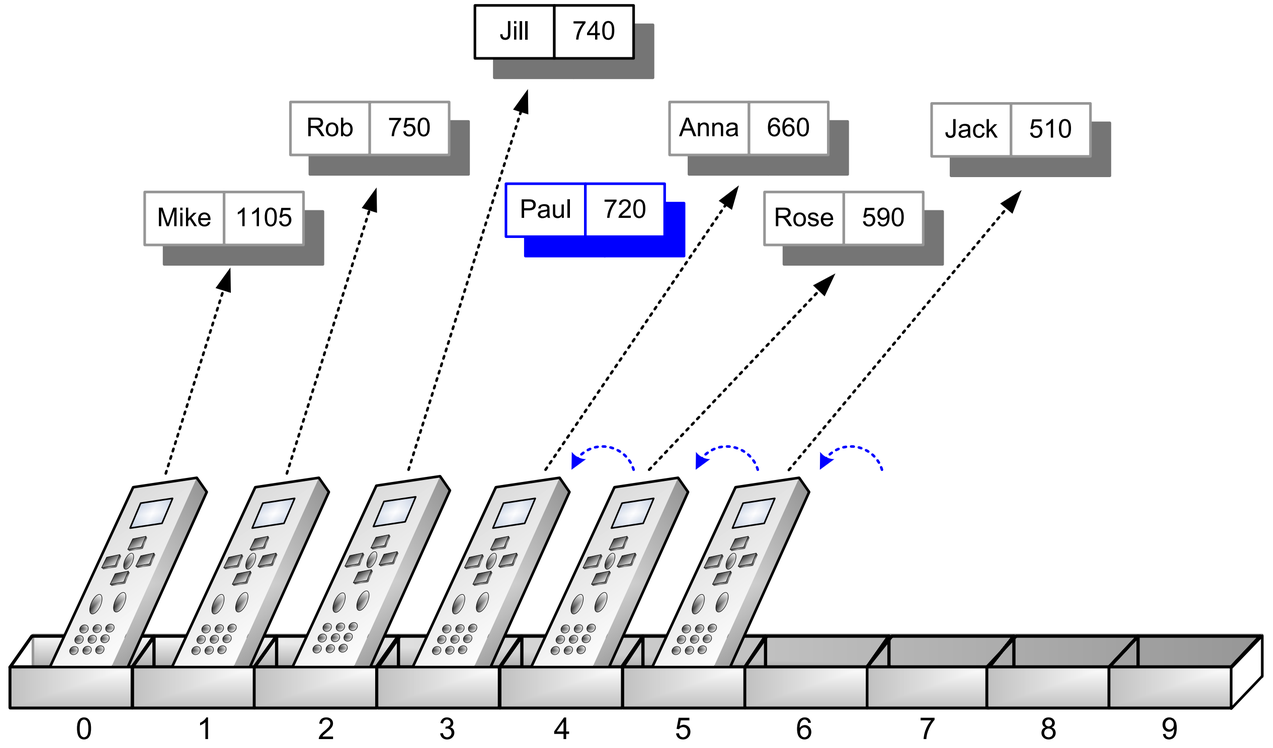


Now we insert the new entry at the right position

Scoreboard

```
public void remove(int i) {
```

- Remove and return the game entry e at index I.
- If I is outside the bounds of the array then the method throws an exception.
- Otherwise, the entries array is update to remove the object at index I and al objects previously stored at indices higher than I are moved one space to the left



Task

1. Implement the classes GameEntry, Scoreboard
2. Write a main function which reads the scores from “scores.txt”
3. Print out the resulting score board

Things to think about:

Can you think of other (better?) ways to implement the add() function?

What other kinds of exceptions should your code handle?