Name: Luke O'Brien Select Section: (MW(1:40)) TR (1:40)

Worksheet #1: Stacks (15 pts)

Download and import the file **Worksheet1.java.** Use the file to answer the following questions.
Please **write your answers in the worksheet.** Add code to the file to see if it works but do not turn in the code.

1) Notice **Worksheet1.java** contains the generic stack class from assignment 5 and a **Book** class. If we wanted to create a stack of books, give a **short explanation** of what changes, if any, are needed to the generic class to create a stack of books. Give a **short reason why**. **(2pts)**

No changes would need to be made. The "stack" would just have to be created & the books added

2) Note that the file contains a **Book** class. What code is needed to create a stack of books? **(2 pts)**
   a. In main, find the comment "*Worksheet1 Question #2*"
   b. At this point, what code is needed to create a stack of books called **bookStack**? Write code here:

```
GenericStackSpec<Book w1> bookStack = new GenericStackSpec<>();
```

3) What code is needed to add new book object to the **bookStack**? **(2 pts)**
   c. In main, find the comment "*Worksheet1 Question #3*"
   d. At this point, what code is needed to add a book to the stack? Write that code here:

```
bookStack.push(book);
// Everything is already in for loop
```

4) What code is needed to print the name of each book in the **bookStack**? Use a **while loop**. **(6 pts)**
   e. In main, find the comment "*Worksheet1 Question #4*"
   f. At this point, write a **while loop** to print the name of each book. Write that code here:

```
while(!bookStack.isEmpty()){
    System.out.println(bookStack.pop().getName());
}
```

5) What code is needed to examine the top element on the **bookStack**? **(3 pts)**
   g. In main, find the comment "*Worksheet1 Question #5*"
   h. At this point, add one line of code to display the book on top of the stack. Write that code here:

```
System.out.println(bookStack.peek());
```

   i. What error occurs when you add the line in 5h and run the file? Why does this occur?

Index out of bounds.

This happened because there where no elements left in the array to look at.

Worksheet #2: Nested Objects (12 pts)

Download and import the file **Worksheet2.java.** Use the file to answer the following questions.

1) Let's create a queue in main. This means, the queue is <u>not nested</u> inside a class.
   In main, find the comment "*Worksheet2 Question #1*"
   a. At this point, write the declaration for a queue of books and place the books that have been created for you into this queue using the ***offer*** method. Write that code here: **(2 pt)**

```
Queue<Book> queueNotNested = new LinkedList<>();
queueNotNested.offer(javaBooks);
queueNotNested.offer(cBooks);
queueNotNested.offer(pythonBooks);
```

2) Next, let's move the queue **inside a class** called **BookQueue**. Complete the **BookQueue** class below by writing on the worksheet the code required for each method. Find *Worksheet2 Question #2.* **(6 pts)**

```
class BookQueue {
    private Queue<Book> queue = new LinkedList<>();

    public int size() {
        return queue.size();
    }

    public void offer(Book book) {
        queue.offer(book);
    }

    public Book remove() {
        return queue.remove();
    }
} // BookQueue
```

3) Finally, test the **BookQueue** class using the code in **Worksheet2.java**
   a. Write the declaration for an object of type **<u>BookQueue</u>**. Place the three books that were created into the object (it acts like a queue!) Find comment *Worksheet2 Question #3a.* **(2 pt)**

```
BookQueue aQueue = new BookQueue();
aQueue.offer(javaBooks);
aQueue.offer(cBooks);
aQueue.offer(pythonBooks);
```

   b. Write the code to print the names of the books in the **<u>BookQueue</u>** object. *Worksheet2 Question #3b.* **(2 pts)**

```
int qSize = aQueue.size();
for(int x=0; x<qSize; x++){
    System.out.println(aQueue.remove().getName());
}
```

Worksheet #3: Complicated Nested Objects (20 pts)
Download and import the file **Worksheet3.java.** Use the file to answer the following questions.

1) Now that you've seen how to nest a queue of books inside a class, let's nest a queue of **Scores** inside the **Player** class which will then be placed inside an array. Complete the **Player** class below by writing on the worksheet the code required for each method. Find comment *Worksheet3 Question #1.* **(6 pts)**

```
class Score  {
   private int value;
   private String dateOfScore;

   public Score (int value, String dateOfScore) {
      this.name = name;
      this.dateOfScore = dateOfScore;
   }

   public int getValue() {
      return value;
   }
} // Score

class Player {
   // Queue of scores for this player
   private Queue<Score> scores = new LinkedList<>();

   public int getScoresSize() {

      return scores.size();

   }

   public void addScore (Score score) {

      scores.offer(score);

   }

   public Score getScore () {

      return scores.remove();

   }
} // Player
```

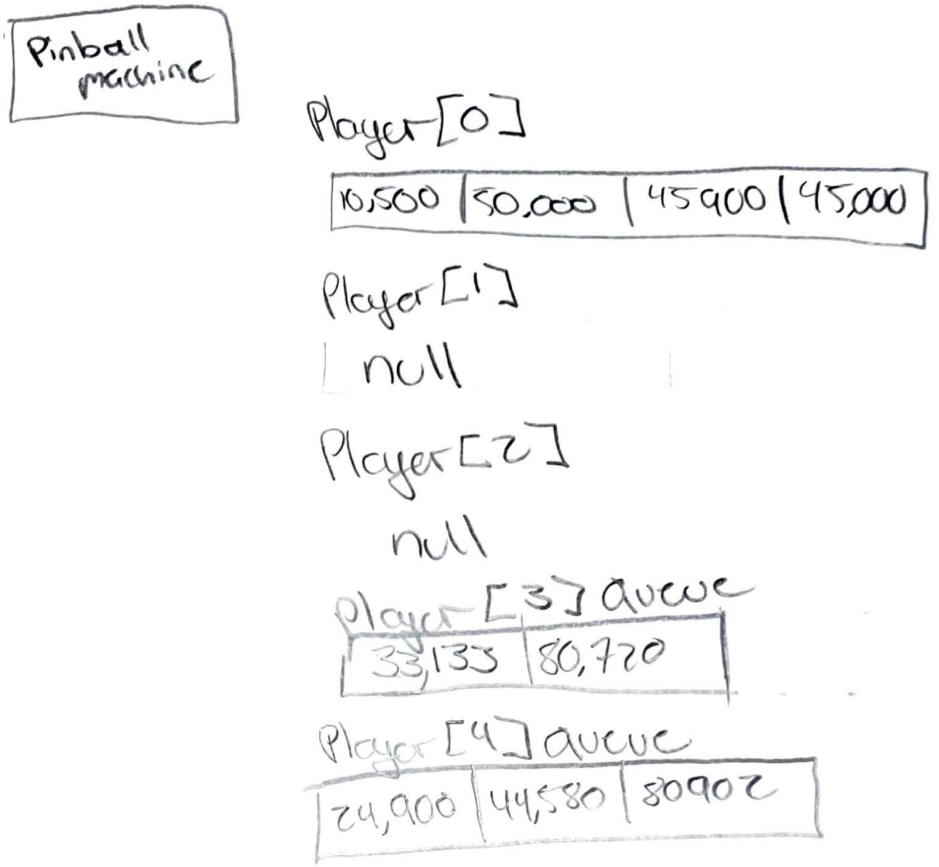
2) Assume we have a modified **PinballMachine** class from Assignment 4 and the **Player** class shown above. An array of player objects has been added. Each player in the array contains a queue as shown above.

```
class PinballMachine {

   private Player[] players;      // Array of players whose scores for this
                                  // pinball machine are saved

   public void addPlayer (Player player, int slot) {
      players[slot] = player;
   }
} // PinballMachine
```

Draw a picture of a **PinballMachine** object. Show the **array of players** and for each player in the array the **queue of scores**. Assume the pinball machine can store scores for only a small number of players so the player array will have 5 slots (0-4) with the players shown below. Show the following in your picture:

1. Show all slots in the player array, including the ones that **do not** contain players. **(10 pts)**
2. Be sure to label the different pieces!

- Slot 0: Frank who has a queue with 4 scores (10,500, 50,000, 45,900, 45,000)
- Slot 3: Paul who has a queue with 2 scores (33133, 80720)
- Slot 4: who has a queue with 3 scores (24900, 44580, 80902)



3) In question 2 above, in order to add a player to the pinball machine's list of players, we use the *addPlayer* method. Now, what if we want to add a **score** to a specific player's **queue of scores**?
This process requires thinking through several layers of nested objects, that is:

- Pinball machine contains a player array which contains players and each player contains a queue

Write a PinballMachine method called ***addScoreToPlayer*** which takes a score object and a slot number. Use the picture above to visualize what needs to be done to add **one score** to the player's queue of scores. Find comment *Worksheet3 Question #3* and write that code here: **(4 pts)**

```
// Add a score to the player in location (slot) in the player array
public void addScoreToPlayer(Score score, int slot) {

    Players[slot].addScore(score);

}
```

Gustavo Bravo

Worksheet #4: Priority Queues and Comparable Interface (23 pts)
Download and import the file **Worksheet4.java.** Use the file to answer the following questions in the worksheet.

1) When the remove method on a priority queue is called, how does it decide which element to remove? **(2 pts)**
it will look for the variable with the highest priority, then remove that first.

2) Write the declaration for a priority queue of integers. Write that code here: **(2 pts)**
```
PriorityQueue<Integer> pQueue = new PriorityQueue<>();
```

3) Write the declaration for a priority queue of player objects. Name this priority queue **results**. Find comment *Worksheet4 Question #3.* Write that code here: **(2 pts)**
```
PriorityQueue<PlayerW4> results = new PriorityQueue<>();
```

4) Let's do some experimenting in code. In **Worksheet4.java** you'll see the **Player** and **Scores** classes from **Worksheet #3: Complicated Nested Objects** with some minor changes to the Player class. After importing the java file, fix the errors by doing the following:

   a. In the **Player** class, complete the ***getScoresSize, addScores*** and ***getScores*** methods.
      i. In Player class, there are 3 comments for "*Worksheet4 Question #4a*"
      ii. Copy the answers from question #1 in **Worksheet #3 Complicated Nested Objects** to complete these 3 methods.
      iii. No need to re-write that code in worksheet for this step.

   b. Now, let's add players to the priority queue called **results. (4 pts)**
      i. In main, find the comment "*Worksheet4 Question #4b*"
      ii. At this point, write code to add **player1** & **player2** to the priority queue called **results**. Write that code here:

```
results.offer(player1);
results.offer(player2);
```

   c. Run the code. What is the result? Show the exact output. **(2 pts)**
Error: The class can not be "casted" to comparable

d. The **Player** class is missing some code that will fix the issue in 4c. Describe the pieces of code that are missing in the **Player** class? **(2 pts)**

The peice of code that is missing is an implimantation of Comparable. thatway the Computer knows how to sort them

e. Explain why the code in 4d is necessary. **(3 pts)**

With the Absence of this code, the Program has no way to know how good like to prioritize the queue. therefore you must tell it.

f. Using the *number of scores* for a player as the comparison factor, add the necessary code to the **Player** class in the java file and rerun it. Did the issue in 4c go away? Write that code here: **(6 pts)**

```
Class PlayerW4 implements Comparable<PlayerW4> {

    .
    .
    .
    .
    .

Public int compareTo(PlayerW4 other) {
    return Integer.compare(this.Scores.size(), other.Scores.size());
}

} // The Error went away
```