COMP 1006B In-Class Test #1 (Winter 2018)

Name: _____ **Student#:** _____ **Tutorial:** ___

(1) [10 marks] Complete the code below so that it determines whether or not a given input string represents a valid password or not. Here are the conditions for a password:

- All passwords must contain one or more comma `','` characters
- All passwords must have exactly two plus `'+'` characters.
- All comma characters must appear in between the two `'+'` characters.

The program should output: "Invalid" to the System console if the input is not a valid password. It should output "Valid " if it is valid. You can use a String's **charAt(i)** method to access the character (type char) at position **i** in the string. You can also obtain the length of a string by using the **length()** method. (hint: find the locations of the comma/plus characters.)

```
public static void main(String[] args){
    System.out.println("Enter password:");
    String password = new java.util.Scanner(System.in).next();

    int plus1 = password.length(), plus2 = -1, plusCount = 0;  // some counters
    int comma = -1;                                            // and position holders
    boolean commaBeforePlus = false;                           // a flag

    for(int index=0; index<password.length(); index+=1){      // 1 – loop OK
      char c = password.charAt(index);                         // 1 – access char
      if( c == '+' && plusCount == 0){                         // 1 – find first +
          plus1 = index;
          plusCount += 1;
      }else if( c == '+'){                                     // 1 - find last +
          plus2 = index;
          plusCount += 1;
      }else if(c == ','){
        comma = index;                                         // 1 – position of comma
        if(comma < plus1){                                     // 1 – check if comma
          commaBeforePlus = true;                              // before first +
        }
      }
    }
    if(   plusCount != 2               // 1 – check if more or less than 2 +s
       || commaBeforePlus              // 1 – check if comma appears before first +
       || (comma > plus2) ){           // 1 – check if comma appears after last +
      System.out.println("Invalid");
    }else{                             // 1 – proper output
      System.out.println("Valid");
    }
}
```

Consider **Book** and **Library** classes defined as follows:

```
public class Book {
   public String   title;
   public int      numPages;
}
```

```
public class Library {
   public Book[]  books;
   public int     numBooks;
}
```

(2) [3 marks] Write a public class method for the **Library** class called **biggerBook()** that takes two **Book** objects as parameters and returns the **Book** that has more pages in it (return the second input book if they have the same number of pages).

```
public static Book biggerBook(Book b1, Book b2) {
   if (b1.numPages > b2.numPages){    // parentheses not needed
       return b1;
   }else{                              // else keyword is not needed
       return b2;
   }                 // 1 - proper method signature,
                     // 1 - proper access of numPages attributes
                     // 1 – proper comparison of attributes
}
```

(3) [5 marks] Complete the **Library** method below which returns the fraction of books that have pages between minPages and maxPages (inclusively) compared to the total number of books in the library. The output will be a number between 0.0 and 1.0 (inclusive).

```
public double fractionOfBooks(int minPages, int maxPages) {
   int  inRange = 0;                          //  1/2 – proper init

   for (int i=0; i<numGames; i+=1) {          // 1   - proper loop

      if (books[i].numPages >= minPages  // 1/2 – correct array access
           && books[i].numPages <= maxPages)   // 1 – correct logic

         inRange += 1;                        // 1/2 – increment inRange


   return ((float)inRange)/numBooks;  // 1 - proper return value
                                      // 1/2 – proper casting

   }
```

(4) [2 marks] Write a proper **toString()** method for the **Book** class that returns a String representation of the book in this format: `"Animal Farm [112 pages]"`

```
public String toString() {  // 1 – proper signature and return type
   return this.name + " [" + title + "pages]";   // this not needed
   }          // 1/2 – correct attribute  used

              // 1/2 – correct format
```