STUDENT NAME ______________________________ **STUDENT ID** ________________________

QUESTION PACKET

CS 207, Fall 2016, Programming II FINAL EXAM
Saturday, December 10, 2016
11:00 a.m. to 1:00 p.m.
Room LWH-1001

1. **Do not turn this page until told to do so.**

2. This exam is **closed book** and **closed notes**.

3. **Write your STUDENT ID Number on every page**. If you do not write your ID Number on a certain page, you **will not receive credit for the question on that page!**

4. **Your name should be on THIS page only**. If you write your name on any other pages, **you will not receive credit for the question on that page!**

5. **You MUST turn in the Question Packet along with the Answer Packet.**

6. There are **6** questions on the exam. Once you start, verify you have all 6 questions.

7. You must place your answer to a question on the **specified** page in the **ANSWER PACKET**. If you place your answer anywhere else, **you will not receive credit for it!**

8. For problems that ask you to write a **method**, you must write the method header **exactly** as shown, but you do not need to write `main()`.

9. For problems that ask you to write a **program**, you should write the `main()` method **only.** You can assume the following import statement and keyboard declarations.

```
        import java.util.*;

        Scanner keyboard = new Scanner(System.in);
or      Scanner kb = new Scanner(System.in);
or      Scanner kbd = new Scanner(System.in);
```

10. For tracing problems, assume that any appropriate import statements are provided.

11. You may use "`SOP`" as an abbreviation for "`System.out.print`" and "`SOPln`" (or "`SOPL`") for "`System.out.println`".

12. You do not need to do any error checking of input values, **unless the problem specifically asks you to do so.**

13. If you are caught looking at other papers or communicating with other students in any way, you will receive an **F** for the course.

STUDENT ID ONLY: ______________________________

Question 1 (20 pts)

Write a method named `maxHourglassSum` that receives one parameter, a 2-dimensional (2D) integer array, named `arr` that contains only positive integers. The method should do the following:

1. Find and return the largest hourglass sum in the 2D array `arr`.
2. An hourglass is defined as a subset of 7 values from a 2D array that have the pattern of an hourglass (note that the letters below represent numbers):
   ```
   a b c
     d
   e f g
   ```
3. An hourglass sum is the sum of the numbers in the hourglass. There can be multiple hourglass patterns in one 2D array and they can overlap. For example, sample array parameters `a` and `b` each have 2 hourglass patterns, whereas `e` has 4 hourglass patterns. Note that these are just examples and the array `arr` could have more or fewer hourglasses than the examples provided.
4. If `arr` does not contain at least one hourglass pattern, throw an `IllegalArgumentException` with the message "`No hourglass.`" Do not worry about handling (i.e. catching the exception).
5. Sample usage is provided below. Do not worry about ragged arrays.
6. Complete and place your code in the **Question 1** box in the Answer Packet. **Any code on this page will not be considered as part of your solution.**

Array Parameter	Sample Method Usage	Return Value

`int[][] a = { {  4, 4, 9, 3 },`
`{ 10, 3, 5, 1 },`
`{ 15, 8, 6, 5 } };`
	`maxHourglassSum(a)`	`49`

`int[][] b = { { 20,  2,  4 },`
`{  4, 11,  5 },`
`{ 15,  0,  3 },`
`{ 17,  5, 12 } };`
	`maxHourglassSum(b)`	`55`

`int[][] c = { {  0,  9 },`
`{  4, 10 },`
`{  1,  8 } };`
	`maxHourglassSum(c)`	`No return value. Throws an IllegalArgumentException`

`int[][] d = { {  6, 19,  2 },`
`{ 14,  5,  3 } };`
	`maxHourglassSum(d)`	`No return value. Throws an IllegalArgumentException`

`int[][] e = { {  8,  1,  0, 15 },`
`{  4, 17,  3,  6 },`
`{  1,  7, 12,  9 },`
`{  0, 20,  2, 13 } };`
	`maxHourglassSum(e)`	`73`

Question 2 (20 pts)

Create a class named `Skier` that has the following:

1. A properly encapsulated `int` instance variable named `id`.
2. A properly encapsulated `String` instance variable named `scores`, which contains each of the scores for the events in which a Skier has participated. You can assume that the scores are decimals separated by spaces. Note that not all Skiers participate in the same number of events.
3. A constructor that accepts an `int` parameter for `id` and sets the corresponding instance variable correctly.
4. A method named `printSkierId` that does not take any parameters and prints out the value of the `id` instance variable. Your output format should match the sample output format **exactly**.
5. A setter method for the `scores` instance variable.
6. A method named `sumScores` that does not take any parameters and returns a `double`. This method should calculate the sum of all the values in the `scores` instance variable.
7. An overridden `equals` method that returns `true` if the sum of the `scores` instance variable of the object passed in and that of the object calling the method are equal.
8. Sample usage is provided below.
9. Complete and place your code in the **Question 2** box in the Answer Packet. **Any code on this page will not be considered as part of your solution.**

Sample usage:

```
Skier s1 = new Skier(2831);
s1.printSkierId();
s1.setScores("80.2 100.3 60.4 32.2");
double d1 = s1.sumScores();
System.out.println(d1 + "\n");

Skier s2 = new Skier(30184);
s2.printSkierId();
s2.setScores("77.3 90.2 54.2");
double d2 = s2.sumScores();
System.out.println(d2 + "\n");

System.out.println(s1.equals(s2));
s2.setScores("133.5 139.6");
d2 = s2.sumScores();
System.out.println(d2);
System.out.println(s1.equals(s2));
```

Output:

```
Skier id: 2831
273.1

Skier id: 30184
221.7

false
273.1
true
```

STUDENT ID ONLY: ______________________________

Question 3 (15 points)

What is the **exact** output of the `main` method in the `FoodApp` class? Place your output in the **Question 3** box in the Answer Packet. **Any output on this page will not be considered as part of your solution.**

```java
public class Food
{
   public static String vit;
   private String name;
   private int cals;

   public Food(String n)
   {
     this(n, 0);
     System.out.println(vit);
   }

   public Food(String n, int c)
   {
     this.name = n;
     this.cals = c;
     vit += n.charAt(0);
     System.out.println(vit);
   }

   public void norf()
   {
     System.out.println(this.name + " "
                                + this.cals);
   }

   public int bar(Object o)
   {
     System.out.println("Food!");
     int boo = 5;
     if (!(o.getClass().equals(this.getClass())))
       boo -= 4;
     else
     {
       Food f = (Food) o;
       boo = f.cals - this.cals;
     }
     return boo;
   }
}
```

```java
public class Vegetable extends Food
{
   private int fiber;

   public Vegetable(String n, int c, int f)
   {
     super(n, c);
     this.fiber = f;
     Food.vit += f;
   }

   public int bar(Object o)
   {
     System.out.println("Veggie!");
     int boo = 3;
     if (!(o.getClass().equals(this.getClass())))
       boo = 6;
     else
     {
       Vegetable v = (Vegetable) o;
       int b1 = super.bar(v);
       int b2 = v.fiber - this.fiber;
       boo = b1 + b2;
     }
     return boo;
   }
}
```

```java
public class FoodApp
{
  public static void main(String[] args)
  {
    Food f1 = new Food("Chicken");
    Food f2 = new Vegetable("Squash", 300, 25);
    f1.norf();
    f2.norf();

    Food f3 = new Food("Corn", 300);
    f3.norf();
    Vegetable v1 = new Vegetable("Kale", 300, 25);
    v1.norf();

    System.out.println(f3.bar(f2));
    System.out.println(f2.bar(f3));
    System.out.println(v1.bar(f2));
  }
}
```

STUDENT ID ONLY: ____________________________

Question 4 (15 points)

Given the below interface and abstract class, create the non-abstract class `TicTacToe` as follows:

1. `TicTacToe` class inherits from the `Game` class and implements the `Board` interface.
2. `TicTacToe` should have a properly encapsulated 2D `char` array instance variable named `b`.
3. The `TicTacToe` constructor should take one `String` parameter and one `int` parameter and set the superclass instance variables.
4. The `create` method should create a 2D character array of size `r` rows and `c` columns, composed of all hyphens (`-`), and set the `b` instance variable to that array.
5. The `play` method takes an `int` parameter representing the player (player 1, player 2, etc). The method should prompt the user to enter the row and the column number, matching the sample output. You can assume that the `Scanner` object has been created (see instructions on first page). Modify the `b` instance variable to contain an `X` or an `O` at the specified row and column. Player 1 is `X` and player 2 is `O`.
6. Override the `printGameInfo` method to print the `Game` name and the number of players on the first line, and then each row of `b` on the following three lines, matching the sample output.
7. **You should write the code to produce the output in the same format provided below. Do not add any additional methods other than the ones described in the instructions.**
8. Complete and place your code in the **Question 4** box in the Answer Packet. **Any code on this page will not be considered as part of your solution.**

Game.java

```
public abstract class Game
{
  private String name;
  private int players;

  protected Game(String n, int p)
  {
    this.name = n;
    this.players = p;
  }

  public abstract void play(int p);

  public void printGameInfo()
  {
    String s = this.name + " Players: "
                         + this.players;
    System.out.println(s);
  }
}
```

Board.java

```
public interface Board
{
  public abstract void create(int r, int c);
}
```

Sample Usage

```
TicTacToe t = new TicTacToe("TicTacToe", 2);
t.create(3, 3);
t.printGameInfo();
t.play(1);
t.play(2);
t.printGameInfo();
System.out.println();
t.play(1);
t.play(2);
t.printGameInfo();
```

Sample Output

```
TicTacToe Players: 2
- - -
- - -
- - -
Enter row: 1
Enter column: 1
Enter row: 0
Enter column: 2
TicTacToe Players: 2
- - O
- X -
- - -

Enter row: 0
Enter column: 0
Enter row: 2
Enter column: 2
TicTacToe Players: 2
X - O
- X -
- - O
```

STUDENT ID ONLY: ______________________________

Question 5 (15 points)

Complete the `divisibleby10()` method below so that it does the following:

1. Reads the input from the `numbers.txt` file (sample provided below). You do **not** know how many lines are in the file. The sample provided is just an example of the file format.
2. Determines whether the number on each line ends with a zero. If it does, print out the result of dividing the number by `10`. Otherwise, print out "`Not divisible by 10`". **Your code should work for integers of any size.**
3. Keep a count of the numbers that have been divided by `10` and display that value (matching the sample output format exactly) after processing the entire file. Note that the sample output displays this value as "`xxx`" since only a sample of the file has been provided.
4. Handles a `FileNotFoundException` by printing out "`File not found`".
5. You can assume that the `Scanner` and `File` classes have been imported as well as the following import statement: `import java.math.*;`
6. Sample usage is provided below. Your output format should match the sample output format **exactly**.
7. Complete and place your code in the **Question 5** box in the Answer Packet. **Any code on this page will not be considered as part of your solution.**

Sample of `numbers.txt`:

```
2300000010
125
579102988759337294783392731113827OO
250
...
```

Sample output:

```
230000001
Not divisible by 10
57910298875933729478339273111382 70
25
...
Count: xxx
```

STUDENT ID ONLY: ______________________________

Question 6 (15 points)

Write a method called `mostFrequentChar` that takes one parameter, a `String s`, and does the following:

1. Returns the character that occurs the most number of times in the `String s`. If there is a "tie" for the most frequently occurring character, return the character that occurs the earliest in the `String`.
2. See sample usage below.
3. Complete and place your code in the **Question 6** box in the Answer Packet. **Any code on this page will not be considered as part of your solution.**

Sample Method Usage	Return Value
`mostFrequentChar("banana")`	`a`
`mostFrequentChar("g8ueixbo927v")`	`g`
`mostFrequentChar("brookkeeps")`	`o`