CS 113 FINAL EXAM
SPRING 2018

There are 13 questions on this test. The value of each question is:

1-10 multiple choice (total 40 pts)
11-13 coding problem (total 60 pts)

You may get partial credit for questions 11-13. If you finish early, use the extra time to double check your work. You may not use notes, books or electronic devices of any sort. All cell phones and other mobile devices must be turned off during the exam. You may not leave the room before you turn in your exam.

By signing below you acknowledge that you have read and understood the instructions above.

Good luck!

Name ___________________________ Student ID _______________________
Section _________

Answers for Questions 1 to 10

<table>
<thead>
<tr>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
<th>Q4</th>
<th>Q5</th>
<th>Q6</th>
<th>Q7</th>
<th>Q8</th>
<th>Q9</th>
<th>Q10</th>
</tr>
</thead>
</table>

EXAM SCORES

<table>
<thead>
<tr>
<th>Q1-Q10</th>
<th>Q11</th>
<th>Q12</th>
<th>Q13</th>
<th>TOTAL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
MULTIPLE CHOICE QUESTIONS 1-10. (4 points each)
For questions 1-4 below, use the following partial class definitions:

```java
public interface A {
    int a=5;
    public void doThis();

    //...
}

public class B implements A {
    protected int b;
    private int c;

    public void doThis(){
        b=2*a;
        c=a*a;
        System.out.println(a+","+b+","+c);
    }
    public int getB(){
        return b;
    }
}

public class C extends B {
    private int z;

    public void doThis(){
        super.doThis(); // line 4
        b+=a;
        z=a;
        System.out.println(a+","+b+","+z);
    }
}

public class Tester {
    public static void main(String[] args) {
        B c=new C(); // line 1
        c.doThis(); // line 2
        System.out.println(c.getB()); // line 3
    }
}
```

1. Which of the following is true with respect to A, B, C and Tester?
   a. C is a subclass of B and B is a subclass of A
   b. B and C are both subclasses of A and implement interface A
   c. C is a subclass of B and B is a subclass of Tester
   d. C is a subclass B but B is not a child of A
   e. None of the above
2. Which of the following data are accessible in class C?
   a. a, b, c, z
   b. b, c, z
   c. a, b, z, doThis
   d. a, b, z
   e. None of the above

3. What is True about the lines commented as line 1, line 2, line 3, respectively, in class Tester:
   a. Line 1 will not compile because type of c does not match the type of object
   b. Line 1 will compile but line 2 will throw an exception
   c. Line 1 and line 2 will run and print the values of a, b, and c
   d. Line 3 will not compile because class C does not contain a method `getB()`
   e. None of the above

4. The statement marked by comment “// line 4” does which of the following?
   a. calls the constructor as defined in the parent of the current class
   b. calls the constructor as defined in the current class
   c. calls the method doThis() defined in class B
   d. calls the method doThis() defined in class C
   e. None of the above

5. Show the output of the following code:
   ```java
   public class Test5 {
       public static void main(String[] args) {
           int[] first = {1, 0, -1}; int[] second = {1, 2, 1};

           bigSwap(second,first[1]);
           System.out.println(first[1] + " " + second[1]);
       }
       public static void bigSwap(int[] first, int second) {
           int temp=first[1];
           first[1]=second*2;
           second=temp;
       }
   }
   ```
   a. 1 2
   b. 1 1
   c. 0 2
   d. 0 0
   e. None of the above

6. If you instantiate an abstract class, the class or object you wind up with
   a. is also an abstract class
   b. is a reference to an object
   c. is an interface
   d. you can’t instantiate an abstract class
   e. None of the above
7. Given the classes below, what is the output of the following program execution:

```java
public class BadStringException extends Exception {
    public BadStringException (String message) {
        super (message);
    }
}

public class TestNames {
    public static void main (String[] args) {
        String[] names={"one", "two", "three"};
        BadStringException problem=new BadStringException("x");

        try {
            for (int i=0;i<=names.length;i++) {
                if (names[names.length-i].charAt(i) !='o') throw problem;
                System.out.print(names[names.length-i].charAt(i));
            }
            catch (BadStringException e){ System.out.print(e.getMessage()); }
            catch (Exception e){ System.out.print("y"); }
        }
    }
}
```

a. twe  
b. yxxy  
c. yxx  
d. y  
e. None of the above

8. If the method is invoked as `recursive(5, 8)`, what is returned?

```java
public static int recursive(int x, int y) {
    if (x == y)
        return 0;
    else
        return recursive(x, y-1) + 1;
}
```

a. This is an infinite recursion  
b. 5  
c. 8  
d. 3  
e. None of the above
9. Given the following code, what is the output?

```java
public class Figure {
    public void display() {
        System.out.print("Figure");
    }
}

class Line extends Figure {
    public void display() {
        System.out.print("Line");
    }
}

public class Inherit {
    public static void tryme(Figure f) {
        f.display();
    }

    public static void main(String[] args) {
        Figure f = new Figure();
        Line ln = new Line();
        f = ln;
        tryme(f);
    }
}
```

a. Line  
b. Figure  
c. FigureLine  
d. Compilation error  
e. None of the above

10. Why is the following method one which has infinite recursion?
    ```java
    public int infiniteRecursion(int n) {
        if (n > 0) return infiniteRecursion(n) - 1;
        else return 0;
    }
    ```

   a. Because there is no base case  
b. Because the base case will never be true  
c. Because the recursive call does not move the parameter closer to the base case  
d. Because the recursive call moves the problem further away from the base case  
e. None of the above
11. (20 points) Write a recursive method called `findSmallest` that finds the largest value among integers in an array. The method takes 2 parameters:
- array, a non-empty array of integers
- low, a positive integer, no more than size of array, representing number of elements to analyze in array.

For example,

```java
... int [] a={13, 22, 3, 44, 25}; System.out.println(findSmallest(a,5)); //will print 3 System.out.println(findSmallest(a,2)); //will print 13 ...
```

12. (20 points) Given the `Movie` class representing a Movie and its ranking:

```java
public class Movie
{
    private int stars;
    public Movie(int stars)
    {
        this.stars=stars;
    }
    public void setStars(int f){
        stars = f;
    }
    public int getStars(){
        return stars;
    }
}
```

Write ONLY the constructor for a class `MovieTheatre` that has two attributes:

- MovieList, an array of Movie objects showing in the theater
- rankings, an array of integers containing the number of movies in each star category, 0-5, inclusive.

The constructor takes an integer parameter named size representing the library’s size. It initializes all the elements of the array to Movie objects with stars value a random number between 0 and 5, inclusive and populates the rankings array accordingly. For example, if the movie theatre is showing:
Movie 1 – 3 stars
Movie 2 – 3 stars
Movie 3 – 5 stars
Movie 4 – 2 stars
The rankings array will contain 1, 0, 2, 1, 0, 0
13. (20 points) An irregular polygon is a polygon that does not have all sides and angles equal, as shown below (image from tutors.com):

![Image of irregular polygons](image from tutors.com)

Given, the definition of the IrregularPolygon class, write the definition of a child class of class IrregularPolygon named Pentagon. The Pentagon class represents polygons with 5 sides and 5 internal angles attributes defined by class Point below. The class should also have these members:
- a constructor for the class that initializes all attributes. The constructor also updates all sides with values from the user.
- A getter for the attribute sides
- a method isRegular() that returns true if the pentagon is regular and false otherwise.
- a method compareTo() that compares two pentagon objects and returns -1/0/+1, if the first pentagon has smaller/same/larger, respectively, perimeter than second one.

**HINT:** Remember to define/redefine any other methods needed in the child class.

```java
public abstract class IrregularPolygon{ 
    protected String name; 
    protected int[] sides; 
    public IrregularPolygon(String name, int numSides){ 
        this.name=name; 
        sides=new int[numSides]; 
    } 
    public abstract int perimeter(); 
    //returns the perimeter of the polygon 
    public String toString(){ //returns the name of the polygon 
        return "polygon"; 
    } 
}
```
Question 11 – Solution
Question 13 – Solution
Quick Reference

Scanner Class
Scanner(InputStream source)
Scanner(File source)
Scanner(String source)
String next()
String nextLine()
int nextInt()
double nextDouble()
float nextFloat()

String Class
String(String str)
int length()
int compareTo(String anotherString)
char charAt(int index)
boolean equals(String anotherString)
String substring(int beginIndex, int endIndex)
String substring(int beginIndex)

Random Class
Random()
float nextFloat()
int nextInt(int num)
int nextInt()

Math Class
static double random()
static final double PI