CSE 131 Introduction to Computer Science **Fall 2016**

Exam II

Given: 10 November 2016 Due: End of Exam Session

This exam is closed-book, closed-notes, no electronic devices allowed. The exception is the "sage page" on which you may have notes to consult during the exam. Answer questions on the pages of the exam. Do not unstaple the pages of this exam nor should you attach any other pages to the exam. You are welcome to use the blank space of the exam for any scratch work.

Your work must be legible. Work that is difficult to read will receive no credit. Be sure code is indented to show its structure. Comments are only needed if you think you need to clarify your work.

You must sign the pledge below for your exam to count. Any cheating will cause the students involved to receive an F for this course. Other action may be taken. If you need to leave the room for any reason prior to turning in your exam, you must leave your exam and any electronic devices with a proctor.

YOU MUST FILL IN YOUR IDENTIFYING INFORMATION CORRECTLY. Failure to do so is grounds for a zero on this exam.

Print clearly the following information:
Name (print clearly):
Student 6-digit ID (print *really* clearly):
Your answers below tell us where to return your graded exam.
What time do you actually attend studio/lab? 11:30am 1pm 2:30pm 4pm
What room? 222 , 218, 216, or 214

Problem Number	Possible Points	Received Points
1	20	
2	25	
3	10	
4	25	
5	20	
Total	100	

Pledge: On my honor. I have neither given nor received any unauthorized aid on this exam.

Signed:_____

1. (20 points) In the next 2 problems, you will write a static method **reverse** that takes in an array of type int and reverses the array. You should change the original input array to it's reverse, i.e. the reversed array should end up in the original input array.

For example given input:

```
int[] array = { 1, 2, 3, 4, 5 };
```

After running your method:

`array` should contain `{ 5, 4, 3, 2, 1 }`

Your solution should <u>NOT</u> create a new array even temporarily.

Write the method below using iteration

```
public static void reverse( int[] array )
{

}
```

2. (25 points)
(a)(20 points) Now rewrite the method from the previous problem using recursion. Use comments to document your input parameters, and indicate the base case and recursive step.
As in the previous problem, your solution should <u>NOT</u> create a new array even temporarily.

(b) (5 points) Below write the call to your method to reverse **`array`**

```
int[] array = {1, 2, 3, 4, 5};
//call your method to reverse array
```

3. (10 points) What does the following code print out

```
public static String foo(int x)
{
     if  (x <= 0) {
         return "a";
      }
          if (x == 1) {
        return "b";
      }
      else {
        return foo(x - 2) + "c" + foo(x - 1);
      }
}
public static void main(String[] args) {
  System.out.println(foo(4));
}
```

4. (25 points) Your task for this problem is to design a **Song** class. A **Song** has-a:

- title
- artist
- length (in seconds)

Follow the instructions below to complete the **Song** class:

```
public Song {

     //Put your instance variables here, making sure to use
     //appropriate types (6 points)









     //Put your constructor below (5 points)









     //Write a getter for the length of a song (4 points)









     //Create a toString() method for the song that includes all of
     the instance variables (5 points)









}
```

(5 points) Write a single line of code that will create an instance of **Song** and store it in a variable. You

can make up the artist, title, and length (or use your favorite song!)

5. (20 points) Your task for this problem is to design a **Playlist** class. A **Playlist** has-a:

- Title
- List of **Songs** (**Note**: do not use module 9 constructs like ListNode or ListItem for this!)

In terms of behavior, the Card object should offer the following functionality:

- It should be possible to add a **Song** to the list
- It should be able to tell us the total playing time of the list (in seconds)

Follow the instructions below to complete the **Playlist** class:

```
public Playlist {

     //Put your instance variables here, making sure to use
     //appropriate types (4 points)









     //Put your constructor below (6 points)









     //Put any other methods you need to complete the class below (10
     //points)









}
```