CMSC131
Spring 2004
Midterm #2

Name: _____

Student ID: _____

Section: _____

Grader Use Only:

#1	a	(2)	
	b	(2)	
	c	(2)	
	d	(2)	
	e	(2)	
	f	(2)	
	g	(2)	
	h	(2)	
	i	(2)	
	j	(2)	
#1			(20)
#2	a	(8)	
	b	(7)	
#2			(15)
#3			(25)
#4			(40)
Total			(100)

Problem 1 (20 points) General Questions.

1. Is it possible in Java to create an array of size 0?

   Circle Yes or No.

2. Suppose you have the following comment:

```
/*
 *
 The function computes the average
 *
 */
```

Modify the comment and turn it into a javadoc comment.

3. What is the difference between `System.out.println()` and `System.out.print()`?

4. Write the line of Java code that designates that the code in a specific file belongs to the 'oopi' package.

5. Write the line of code that converts the integer variable `i` to a String that represents that integer?

6. Write the statement that will allow us to access any class present in the java.util package within a Java program.

7. Which of the following situations will occur when an exception occurs and it is not handled? Circle your choice(s).

a. The debugger is invoked in order to take care of the exception.
b. The program is terminated (stops executing), and the call stack is displayed.
c. An infinite loop will occur.

8. Describe two exceptions discussed in lecture. Describing two situations causing those exceptions will be considered a valid answer (i.e., you do not have to know the precise names of the exceptions.)

9. Can you have more than one class per file in Java? If there are any restrictions, what are they?

10. Given the following code fragment:

```
String name1 = null;
String name2 = "";
```

Is the expression `(name1 == name2)` true or false?

Problem 2 (15 points)

Write the output generated by the following code segments.

1.
```
for (int i=0; i<5; i++) {
      switch (i) {
      case 0:
            System.out.println("m");
            break;
      case 1:
      case 2:
            System.out.println("o");
            break;
      case 3:
            System.out.println("d");
      case 4:
            System.out.println("!");
            break;
      }
}
```

2.
```
for (int i=0; i<3; i++) {
      for (int j=0; j<3; j++) {
            System.out.println("i=" + i + ", j=" + j);
            break;
      }
}
```

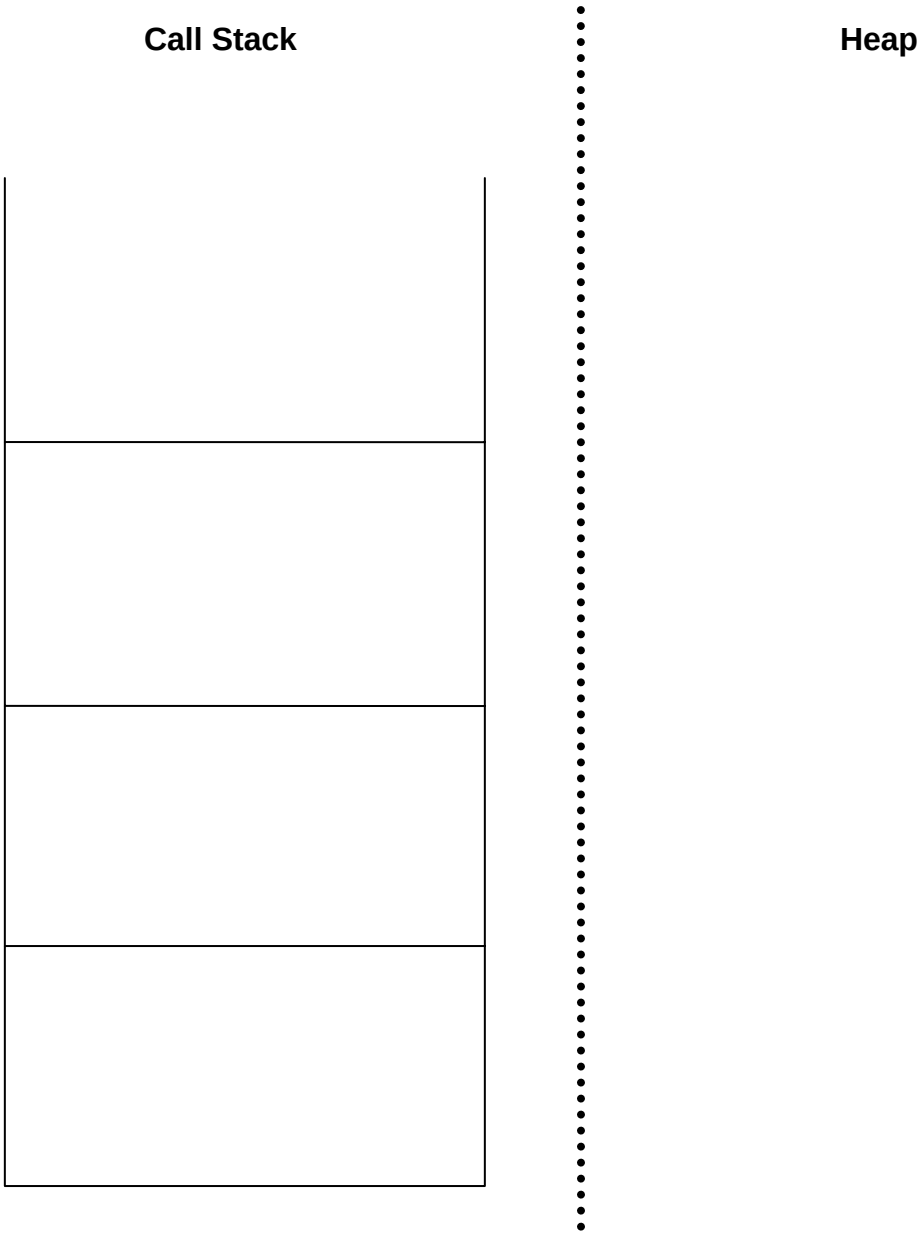
Problem 3 (25 points)

On the following page write a memory diagram representing the variables, and objects in the callstack and heap at the point where execution reaches the point marked as “STOP HERE”. Notice this point is before the method returns (that is, we also want to see the values of the local variables within “process” before it exits.)

```
public class Prob3 {
	public void process(String[] data, int theIndex, String theInfo) {
		data[theIndex] = theInfo;
		theIndex = 1;
		// STOP HERE
	}

	public Prob3() {
		String[] strValues = new String[4];
		strValues[1] = new String("House");
		strValues[2] = new String("Permit");
		String info = new String("Warning");
		int index = 0;

		process(strValues, index, info);
	}

	public static void main(String[] args) {
		Prob3 prob3 = new Prob3();
	}
}
```

Fill in the memory diagram template on the next page

Fill in memory diagram here



Problem 4 (40 points)

For this problem you must implement a class called PasswordVerifier. The class has the following instance variables:

```
String password;    // Represents a password value
int maxAttempts;    // Maximum number of attempts allowed
                    // during a password reading process
```

The class has the following constructor and method:

1. **Constructor** - It takes two parameters: A String object representing the password and an integer representing the maximum number of attempts. The constructor will initialize the corresponding instance variables.

2. **readPassword** - This method takes no parametes and returns boolean that should be true if the password was successfully entered, and false otherwise. It asks the user for a password value as long as the user provides an incorrect value and as long as the number of maximum attempts has not been exceeded. The message “Enter Password” must be shown to ask the user for a password.

   For each attempt that results in an incorrect password, the method will display the message “Incorrect password, try again". If the user provides the correct password the method will display "Welcome" and it will end.

   If the user does not provide the correct value and has exhausted the maximum number of attempts the method will display the message "Access Denied" and it will end. Notice that the method will stop only if the user provides the correct password or if the maximum number of attempts has been exceeded. A valid password input is considered an attempt.

   You must use JOptionPane methods in order to read, and display information (see below). You must use a do-while iteration statement (you will lose credit if you use any other iteration statement, of if you attempt to solve this without an iteration statement.) You do not need to provide a main() method.

Some JOptionPane methods:

showInputDialog – Shows a dialog asking the user to type in a String. For example:

```
String inputValue = JOptionPane.showInputDialog("Enter Password");
```

showMessageDialog – Shows a dialog that displays a provided String.

```
JOptionPane.showMessageDialog(null, "Incorrect Password");
```

Extra page for code writing

Extra page for code writing