CMSC131
Spring 2005
Midterm #1

Grader Use Only:

#1		(40)
#2		(15)
#3		(15)
#4		(30)
Total		(100)

First Name: _____

Last Name: _____

Student ID: _____

Section time _____ TAs: _____

I pledge on my honor that I have not given or received any unauthorized assistance on this examination.

Your signature: _____

General Rules (Read):

- This exam is closed-book and closed-notes.
- If you have a question, please raise your hand.
- Total point value is 100 points.
- **WRITE NEATLY**. If we cannot understand your answer, we will not grade it (i.e., 0 credit).

Problem 1 General Questions (40 pts)

a. (4 pts) Which of the following are valid Java identifiers? (Circle the valid ones.)

```
Sugar          wall&8          U_k          3Abc
```

b. (2 pts) Is null the same as the empty string ("")? Yes/No (Circle one.)

c. (2 pts) Can two or more reference variables refer to the same object at the same time? Yes/No (Circle one)

d. (2 pts) How many distinct String object instances are created in the following code segment?

```
String value = "Baseball";
value = "GoodBye";
String valueTwo = value;
```

Answer:_____

e. (2 pts) The following code fragment does not compile. Why? Briefly explain (in one or two sentences).

```
String k;
System.out.println(k.length());
```

Answer:

f. (3 pts) What is the output generated by the following code segment?

```
int y = 20;
int x = 40;
double d = 30.5;
if (x > 5 && d < 30 && (++y > 5)) {
          x++;
        }
System.out.println("X: " + x);
System.out.println("Y: " + y);
System.out.println("D: " + d);
```

_____

_____

_____

g. (6 pts) The Test and TestDriver classes are defined as follows:

```
public class Test {
  public int p;
  private void print() { }
  public void printAll() {
      print();
  }
  private float k;
}
```

```
public class TestDriver {
  public static void main(String[] args) {
      /* CODE SEGMENT */
  }
}
```

Indicate whether the following code segments are valid or invalid. Each code segment will replace /* CODE SEGMENT */ in TestDriver. Consider each code segment individually.

i.
```
Test t1 = new Test();
t1.p = 100;
```
VALID/INVALID

ii.
```
Test t2 = new Test();
t2.print();
```
VALID/INVALID

iii.
```
Test t3 = new Test();
t3.printAll();
```
VALID/INVALID

iv.
```
Test t4 = new TestDriver();
t4.printAll();
```
VALID/INVALID

v.
```
Test t5 = new TestDriver();
t5.k = 10;
```
VALID/INVALID

vi.
```
Test.printAll();
```
VALID/INVALID

h. (2 pts) What is the relationship between a class and an object? Do both terms refer to the same entity? Briefly explain (in one or two sentences).

Answer:

i. (2 pts) Write the file name extension associated with a file containing bytecode.

j. (6 pts) **Add parentheses** to specify completely the order in which the following expressions will be evaluated. You **do not** need to specify the final value of the expressions. The variables associated with the expressions are:

```
int w = 10, x = 20, y = 30;
boolean b;
```

Note: Add parentheses only. DO NOT evaluate.

i. `b = w < x && y < 30 ;`

ii. `x = w *= y ++ - 2 ;`

iii. `y = - x + - y * 5 % 3 ;`

k. (3 pts) Which of the following statements are valid?

i. `long x = 20;`
`int y = x;` VALID/INVALID

ii. `float m = 20.50f;`
`double k = m;` VALID/INVALID

iii. `long m = 5L * 4;` VALID/INVALID

l. (2 pts) What will occur if the following code segment is executed? Briefly explain (in one or two sentences).

```
String music = null;
boolean b = music.equals(null);
```

Answer:

m. (4 pts) Complete the following trace table.

```
int a = 10, b = 21;
int c = b % 2;
b += a;
a *= 2;
boolean d = true && !(a > 10);
```

a	b	c	d	

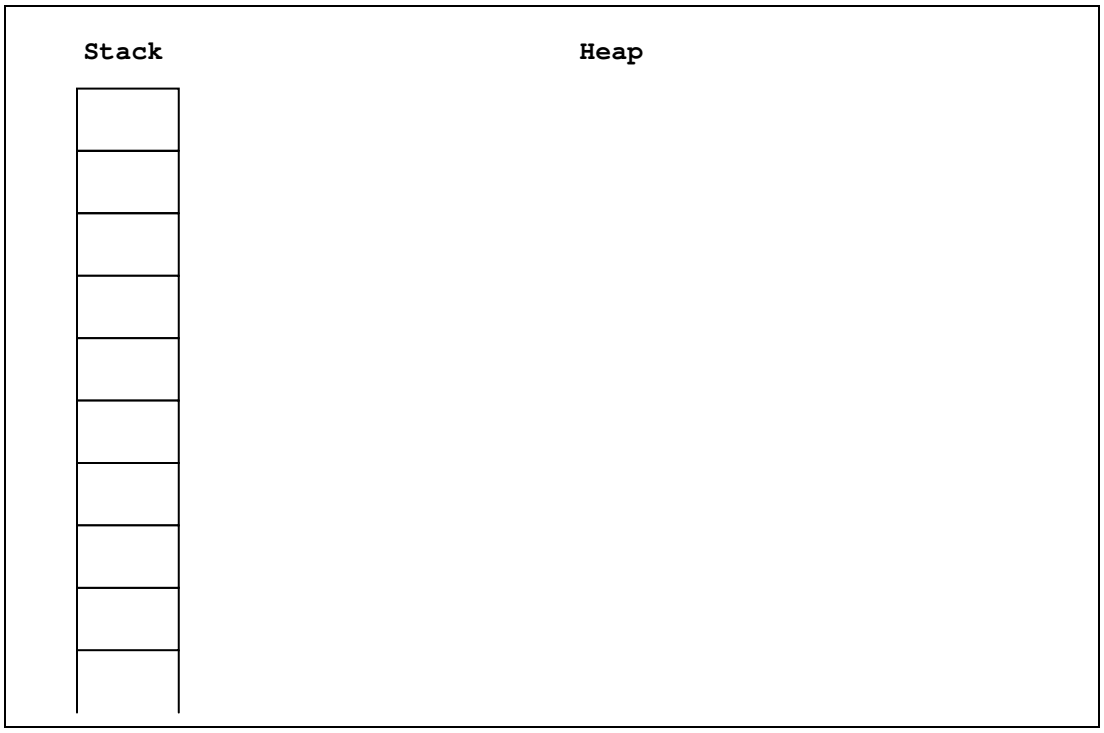
Problem 2 Memory Map (15 pts)

Draw a memory diagram for the following program up to point where the comment "STOP HERE" appears. Specify any objects that have been created, and the values of the variables m, b, w, x, s, d and t. Remember that we represent references (addresses) with arrows, objects with rectangles, and the null value with a line ended with a perpendicular line.

Note: As in class, the memory diagram must include all arguments and local variables associated with the main method.

```
public class MemMap {
    public static void methodOne(int x, String s){
        String t = "Sky";
        String d = new String(s + t);
        x = 200;
        s = t;
        t = null;
        // STOP HERE
    }

    public static void main(String[] args) {
        String m = "Blue";
        boolean b = true;
        int w = 10;
        methodOne(w + 5, m);
    }
}
```



Problem 3 Pseudocode (15 pts)

Write **pseudocode** (NO IMPLEMENTATION) for a method that reads a sequence of user-provided integer values and returns the minimum value in the sequence. The method has one formal parameter, an integer called **size**, that specifies the size of the sequence. You may assume that **size** is a valid integer value (> 0). You should return the first minimum value if there is more than one. As a simplification, you may read a value and store it in some variable x as follows: `x = read()`. Also you may print a value x or a message as follows: `print(x)` or `print("Hello")`.

Important: Keep in mind that pseudocode is **not** an English prose-like description of a particular task. (You will receive 0 credit if you provide a text paragraph that describes the task.) Rather, it is a code-like step-by-step procedure that describes how to accomplish the given task. Your pseudocode should be detailed enough that it can be converted by a competent programmer into Java, but should not contain Java-specific details.

WRITE YOUR PSEUDOCODE IN THE SPACE BELOW.

Problem 4 Class Implementation (30 pts)

For this problem you will implement a class called **Utilities**. The class has two non-static public methods.

a. **minimum** - It is the implementation of the **pseudocode** you defined above. Use the message "Enter Value" to prompt the user for values. Each input value will be read using a separate call to `JOptionPane`. You may assume that all inputs are **valid.**

b. **components –** The method's prototype is:

```
public Picture components(Picture picture, String colors);
```

This method is basically the one you implement in homework #3. It will return a Picture with only the color components specified in the **colors** parameter. The colors parameter can be any combination of the lowercase letters r, g, and b. If you don't recall how to use the `selectComponents` class then read the description at the end of the page. Remember that to retrieve a character from a String we use the method `charAt(x)` where `x` is an integer index value. The following are examples of some valid values for the **colors** parameter: `"rrgb"`, `""`, `"rbbbrb"`.

.

Restrictions for both methods

i. You cannot define instance variables (variables that are part of the class). Of course, local variables are fine.
ii. You don't need to use meaningful variable names.
iii. You must include necessary import statements.

WRITE YOUR PROGRAM ON THE NEXT PAGE.

Read only if you don't recall how to use selectComponents

We use the `selectComponents` class to generate a Picture with a particular set of color components. The constructor for this class takes a source picture reference followed by three boolean values where each one indicates whether we want the red, green and blue color components, respectively.

WRITE YOUR PROGRAM HERE.