CSC 106 Exam #2 (Stages 6 through 8)

**Name:** Andrew Januszko

## CSC 106 Exam #1

## Part 1: Written

### Short Answer

1. (8 pts.) Fill in the following table with the definition, description of the scope, and a description of the lifetime for each kind of variable:

| Variable definition | Scope | Lifetime |
|---|---|---|
| Instance Variable: | Depends on the visibility modifier | Same as the life time of the object |
| Method Parameter: | the method body | from beginning of method to the end of it |
| Local Variable: | declared point to the end of the block containing the declaration | Parallel to the scope but reflects runtime of the system |
| For Loop Variable: | includes the block controlled by the for loop | the for loop being executed |

# Reading Code

1. (4 pts.) Draw the memory diagram of the array built by the following code:

```
String[] x = new String[6];
for (int i=0;i<x.length;i++)
{
     x[i] = “Contents: ”+i;
}
```

x[Contents:0, contents:1, contents:2, contents:3,
Contents: 4, contents:5]

2. (1 point each) If x contains the String “NK#Sfo!7845U” what would each of the following evaluate to?

0 1 2 3 4 5 6 7 8 9 10 11

a. x.charAt(4);

f

b. x.substring(6,10);

!7845

c. x.length();

12

d. x.contains(“W&”);

0 / false

e. x.contains(“K#S”);

1 / true

f. x.substring(3);

Sfo!7845U

g. (x == “NK#Sfo!7845U”)

false, strings cannot be assessed
using == .matches

3. (3 pts.) What is the output from the following code?

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18

19

```
String x = "This Is Not A TEST!";
int count = 0;
for (int i=0;i<x.length; i++)
{
      if ((x.charAt(i)>= 'L') && (x.charAt(i) <= 'T'))
      {
            count++;
      }
}
System.out.println(count);
```

0

## Paper Practice

4. (3 pts.) What are the border cases that this test verifies?

```
@Test
public void testEarlyWarningGrade()
{

    assertEquals("PA", g.earlyWarningGrade(70));
    assertEquals("D", g.earlyWarningGrade(69));
    assertEquals("D", g.earlyWarningGrade(60));
    assertEquals("F", g.earlyWarningGrade(59));
 }
```

The numbers on the edge of a grade change ie 60=D but 59=F

5. (1 pt.) In Lab 5, we changed some, but not all, of our variables to longs in an unsuccessful attempt to avoid overflow. However, the instructions were very explicit that you should not change the variable i in this loop to a long. What would have happened if you had?

```
System.out.println("");
System.out.println("Adding one . . .");
int i;
for (i = 2147483645; i > 0; i++)
{
      System.out.print(" " + i);
}
System.out.println(" " + i);
```

It would run for a long time until it overflowed

overflow

so

6. (6 pts.) The following code is a method in a class named PaperClass and a test to verify that method's behavior. Mark all of the errors in the code.

```
/**
 * Create an array that contains even negative numbers
 * down to -x in sequential order
 *     two errors
 * @return the array
 */
public int[] buildDecreasingByTwo(int x)
{
      int[] y = new int[x];
      int number = -1;
      for (int i = 0; i < y.length; i++)
      {
            y[i] = number;
            number = number - 2;
      }
      return y;
}
```

ex x=5

y[-1, -3, -5, -7, -9]

Creates odd Numbers

make number = 0.

return → int type

**This code is from the test of PaperClass.**

```
/**
 *    one error
 * @param results
 * @param size
 */
private void checkDecreasingArray(int[] results, int size)
{
      assertEquals(size/2, results.length);
      for (int i=0;~~i<results.length; i++)~~
      {
            assertEquals(-2*i, results[i]);
      }
}

/**
 * - one errors
 * This is the test and it uses another method named
 * checkDecreasingArray
 */
public void testDecreasing()
{
      PaperClass p = new PaperClass();
      int[] results = p.buildDecreasingByTwo(6);
      checkDecreasingArray(results, 6)

      p = new PaperClass();
      checkDecreasingArray(p.buildDecreasingByTwo(8), 8);
}
```

Comes after for loop

missing Semicolon

7. (6 pts.) What is the output from the following code?

```
public class Paper3
{
    public static void main(String[] args)          (32,57)
    {                                                 p   q
        int x = 32;
        int y = 57;
        Class3 c = new Class3(x,y);
        System.out.println(c.playWith());    -25


        y = 32;
        x = 57;
        c = new Class3(y, x);                      (32, 57)
        System.out.println(c.playWith()); -25        p   q
    }
                                   -25
}                                  -25

public class Class3
{
    private int x;
    private int y;

    public Class3(int p, int q)
    {
        x = p; 32
        y = q; 57
    }

    public int playWith()
    {
        return x - y;
    }
}
```

## Writing Code

8. (4 pts.) Write a loop that does not use an early exit condition that encodes the same behavior as this loop:

```
~~int y = 103;~~
~~double x = 2;~~
~~while (true)~~
{
    ~~y = y - 4;~~
    System.out.println("y = "+ y +" x = "+ x);
    if (y%6 == 5)
    {
        return y+3;
    }
    ~~x++;~~
}
```

```
for (double x = -2, int y = 103; true; x++){
    y=y-4;
    System.out.println("y= "+y+" x= "+x);
    if(y%6==5){
        return y+3;
}   }
```

## Bonus (4 points)

For each of the methods below identify the scope of the variable **x** by drawing a box around the portion of code within the scope.

```
public void sillyMethod1()
{
        int x;
        x = 32;
}
```

```
public void sillyMethod2()
{
        int y = 3;
        while (y > 0)
        {
                int x = 42;
                y = y - 1;
        }
        y = 42;
}
```

```
public void sillyMethod3()
{
        int y = 3;
        while (y > 0)
        {
                y = y - 1;
                int x = 42;
                x = x - 1;
                y = y - 1;
        }
        y = y - 1;
}
```

```
public void sillyMethod4()
{
        int y = 3;
        while (y > 0)
        {
                y = y - 1;
                int x = 42;
                x = x - 1;
                for (int i = 0; i < 3; i++)
                {
                        System.out.println(i);
                }
                y = y -1
        }
        y = y -1;
}
```