Name_________________________ Date ___________

COMP 110 Midterm Practice Test

UNC Honor Pledge: I certify that no unauthorized assistance has been received or given in the completion of this work

Signature _______________________________

You may NOT use any course materials in completing this test.

The test consists of three sections: first, a section of short answer questions; second, a section requiring you to trace a short piece of code and write the output produced by the code; finally, a section requiring you to write code. The second and third sections each provide four questions of different point values. You can choose THREE questions from section two, and TWO questions from section three to be graded. It is possible to attempt greater than or fewer than 100 points, however your final grade will be calculated out of 100 points. Point values are given for each question.

Illegible answers will not receive credit.

Part 1. (40 points) Answer ALL questions

1.) (2 points) What is the value of *var*?

```
double var = (3 / 2) * 2.0;

     2.0
```

2.) (4 points) List 4 primitive types.

Byte, short, int, long, float, double, char

3.) (3 points) Declare a variable to hold the amount of money you have saved for spring break and initialize it to 150.75. Name your Java variable so any programmer would know what value it holds. Your variable name should be at least two words.

```
double breakMoney = 150.75;
```

4.) (3 points) Declare and calculate a double, averageAge, if you have (as integers) sumAges and totalPeople.

```
double averageAge = (double)sumAges / (double)totalPeople;
```

5.) (2 points) Give me two *good* reasons why you should always comment your code.

- Easier for other coders to read and understand
- Easier for you to understand when you look at your code a year later

6.) (2 points) Circle the expression you would use? Why?

`if (myVar == 15)` [✓ `if(15 == myVar)`]

The compiler will catch your error if you accidently write `15 = myVar` (assignment) instead of `15 == myVar`

7.) (2 points) Why is *information hiding* a good thing?

- Methods can be used without understanding details of code
- Programmer's job becomes simpler
- Code is easier to read

8.) (2 point) The two values of a bit are _0_ and _1_ .

9.) (3 points) Explain how and when to use each of the following?

a.) = assignment

b.) == equality test for primitive types

c.) equals() equality test for class types

10.) (2) Suppose you have the variable *s* of type String that has been initialized to "Spring break! Woo!" Write the data stored in *s* and place the index of each character below the string.

S	p	r	i	n	g		b	r	e	a	k	!		W	o	o	!
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17

11.) (5) Give the return *type* and *value* for the following method calls on *s* as defined in problem 10.

a.) `s.length();`

`int, 18`

b.) `s.charAt(8);`

`char, 'r'`

c.) `s.indexOf("!");`

`int, 12`

d.) `s.lastIndexOf("r");`

`int, 8`

e.) `s.substring(14);`

`String, "Woo!"`

12.) (10 points) In the code that follows, underline all Classes, circle all objects, draw boxes around all methods, and draw a line through all ~~arguments~~.

```
Scanner sc = new Scanner(~~System.in~~);

System.out.println(~~"Do you think exams are fun?"~~);

String s = sc.nextLine();

if(s.equals(~~"yes"~~))
{
   System.out.print(~~"Really?"~~);
}
else if(s.equals(~~"no"~~))
{
   String s2 = new String(~~"me neither"~~);

   System.out.print(~~s2~~);
}
```

Part 2. (30 points) Choose any THREE questions.
For questions 13-16, write the output for the piece of code if it compiles. If it does not compile, tell me what error I made.

13.) (8 points)

```
int i = 4;
int total = 0;
while (i < 7)
{
   int j = i;
   j = (j * 4) % 15;
   total += j;
   System.out.print(j + " + ");
   i++;
}
total += j;
System.out.print( j + "= " + total);
```

Error: the variable *j* is local to the *while* loop and does not exist outside of the *while* loop

14.) (10 points)

```
for(int i = 1; i < 8; i += 3)
    for(int j = 8; j > i; j -= 2)
       System.out.print(j + " ");
```

```
8 6 4 2 8 6 8
```

15.) (12 points) Given the input order:

4 10 17 1 2 7 20 100

```
          Scanner sc = new Scanner(System.in);
          int sum = 0;
          int a = sc.nextInt();
          do
          {
             if (a % 3 == 1)
             {
                sum += a;
                System.out.print(a + " ");
             }
             else
             {
                sum += 2*a;
                System.out.print(2*a + " ");
             }
             System.out.print("+ ");
             a = sc.nextInt();
          }while (!(sum > 50));

          System.out.println("= " + sum);
```

```
4 + 10 + 34 + 1 + 4 + = 53
```

16.) (15 points) Note: methodFun can be called from main without an object because the method is in the same class as main.

```
public static void methodFun(int x, int y)
{
   String s = null;
   switch ((x + y) % 5)
   {
      case 4: s = " compile again";
         break;
      case 3: s = " big bugs";
         break;
      case 1: s = "fix one bug,";
         break;
      case 2: s = " little bugs";
         break;
      case 0: s = "fix a bug";
         break;
      default:
         break;
   }
   System.out.print(s);
}
```

```
public static void main(String[] args)
{
   int x = 3, y = 4, z = 4;

   if ( (z < x) && (y < z) ) System.out.print("Go Heels");
   else System.out.print((x * y * z + 52));

   while (z != 0)
   {
      if ((x - y) >= 0)
       {
          if (x < 9)
          {
             System.out.print(" in the code\n" +(25*x-25));
             methodFun(z-1, x);
             System.out.println("\n");
           }
           else
               methodFun(x,z);
            x += 4;
       }
       else
       {
             methodFun(z,x);
             x += 2;
       }
       z --;
    }
    System.out.print("\n" + ((y * 25) + (z + 1)));
    methodFun(z,x);
    System.out.print(" in the code...");
 }
```

```
100 little bugs in the code
100 little bugs
fix one bug, compile again
101 little bugs in the code...
```

Part 3. (30 points) Choose any TWO questions

17.) (10 points) Complete the method *pow2* (using a loop) to calculate and return the value 2^n. Assume n is a non-negative number.

```
public int pow2(int n)
{
    int p = 1;
    for(int i = 0; i < n; i++)
    {
        p = p*2;
    }
    return p;
}
```

18.) (15 points) Write the code for removing the first and last word of any string. (Example: "Spring break! Woo! Hoo!" would return "break! Woo!") Do not worry about error checking. I will not take off for simple syntax or off-by-one errors. You should only need methods from problem 11.

```
int space1 = s.indexOf(" ");
int space2 = s.lastIndexOf(" ");
System.out.print(s.substring(space1+1, space2));
```

19.) (20 points) The Fibonacci sequence is defined such that a number in the sequence is the sum of the previous two number in the sequence such that the first two numbers are 0, 1.

The Fibonacci sequence: 0, 1, 1, 2, 3, 5, 8, 13, 21, …

Write a function, fib that takes an integer as input and displays the Fibonacci sequence to the first value greater than the input.

Example: `fib(6)` would display `0, 1, 1, 2, 3, 5, 8`

```
public static void fib(int n)
{
    if(n < 0)
        System.out.print("0");
    else
    {
        int f0 = 0;
        int f1 = 1;
        System.out.print(f0 + ", " + f1);
        while(n >= f1)
        {
            int temp = f0 + f1;
            f0 = f1;
            f1 = temp;
            System.out.print(", " + f1);
        }
    }
}
```