CS312 Fall 2018 Exam 1 Solution and Grading Criteria.
Grading acronyms:
BOD - Benefit of the Doubt. Not certain code works, but, can't prove otherwise
Gacky or Gack - Code very hard to understand even though it works. (Solution is not elegant.)
LE - Logic error in code.
NAP - No answer provided. No answer given on test
NN - Not necessary. Code is unneeded. Generally no points off
OBOE - Off by one error. Calculation is off by one.
RTQ - Read the question. Violated restrictions or made incorrect assumption.

1. Expressions: 1 point each, 15 points total

A.	`5 + 3 * 7 / 2`	**`15`**	
B.	`142 / 10 + 17 / 34`	**`14`**	
C.	`7.0 / 2.0 + 7.5 / 2.5`	**`6.5`**	
D.	`37.5 / 10.0 + 5 / 10 + 7 / 2.0`	**`7.25`**	
E.	`4 - 2 + "4" + 2 + 2 * 4`	**`"2428"`**	**Must have quotes**
F.	`3 * -4 + "DHF" + (6 - 10)`	**`"-12DHF-4"`**	**Must have quotes**
G.	`40 % 5 - 35 % 6 + 15 % 30`	**`10`**	
H.	`2.9 % 1.2 + 1635 % 5`	**`0.5`**	
I.	`"" + -3 + 5 + 45 / 10`	**`"-354"`**	**Must have quotes**
J.	`(double) (16 / 5)`	**`3.0`**	
K.	`(int) (.98765 * 10)`	**`9`**	
L.	`(double) 10 / 4 + 15 / 2`	**`9.5`**	
M.	`Math.pow(Math.floor(2.95), Math.ceil(4.0))`	**`16.0`**	
N.	`Math.floor(-39.75 / 10.0)`	**`-4.0`**	
O.	`15 / 4 / 3.0 - 18 / 5 + (15 / 10.0)`	**`-0.5`**	

2. Code tracing: 2 points each, 20 points total. Place you answer in the box to the right of the code. If the code results in a syntax error, answer **syntax error.** If the code results in a runtime error, answer **runtime error.**

AS SHOWN or - 2. First two instances of "answer" counted wrong.

A. `5 8 2.5`
B. `-20`
C. `6x32.01`
D. `18`
E. `265`
F. `70`
G. `Runtime error (due to divide by zero)`
H. `3 16 8 -5`
I. `24 10`
J. `4517`
K.

3. Syntax errors, 1 point each.

A. " in middle of string or missing escape sequence

```
System.out.println("Line 1"\nLine2");
System.out.println("Line 3");
System.out.println();
System.out.println("Line 5");
```

B. variable xs already declared

```
int x2 = 72;
int y2;
y2 = x2 / 10 + 3;
int x2 = y2 * 5;
y2++;
```

C. No error

D. Subtraction not defined for String operands

```
String s4 = "Java";
int x4 = 10;
int y4 = 3;
s4 = s4 + x4 - y4 * 5;
String s5 = s4 + s4 + s4;
```

E. No error

F. STRING no a valid data type or Java is case sensitive OR bad escape sequence in println String. (Either okay)

```
STRING s6 = "5 + 4";
System.out.println(4 - 5 + s6
            + "   10\3 = 3 in Java");
System.out.println(s6);
```

G. Variables resulttt not defined or variable result is misspelled

```
int result = 0;
for (int i = 4;i <= 25;i++) {
      int temp = i * 3;
      resulttt += temp;
}
System.out.println(result);
```

H. variable i no longer in scope after loop or i is undefined

```
double total = 0.0;
for (int i = 0; i < 100; i++) {
      double temp = Math.sqrt(i * 1.0);
      total = temp + total;
}
System.out.println(total + " " + i);
```

I. Cannot assign a double expression to a String

```
double a7 = 3.7;
String s7 = a7 + (5 - 10);
```

J. Missing return statement. (can circle method header or last line or point out missing return.)

```
public static int methodJ(int x, int y) {
      System.out.println(x);
      x++;
      y--;
      System.out.println(x + y);
}
```

K. Parameters (or arguments) not in correct order or of the wrong type.

```
String s9 = "**";
int y = s9.length() + 10;
methodK(s9, y);
System.out.println(s9 + " " + y);

public static void methodK(int x, String s)
{
      System.out.print(x + " " + s);
}
```

L. variable ck not initialized. Can circle variable declaration of ck++

```
int ck;
int garg = 15;
ck++;
garg -= Math.abs(100);
```

4. Programming: 8 points:

```
    public static int getQuadrant(double x, double y) {
        int result = 0;
        if (x > 0 && y > 0) {
            result = 1;
        } else if (x < 0 && y > 0) {
            result = 2;
        } else if (x < 0 && y < 0)
            result = 3;
        else if (x > 0 && y < 0) {
            result = 4;
        } // else point is on an axis or at the origin
        return result;
    }
```

1 point for each quadrant
2 points for axes and origin handled correctly
2 points for correctly returning
missing return compile error - 1, || not && -3, print out -3

5. Programming: 12 points

```
    public static void printFigure(int n) {
        for (int i = 1; i <= n; i++) {
            final int NUM_CHARS = (i * i) / 2; // lots of ways to do this

            for(int j = 0; j < NUM_CHARS; j++) {
                System.out.print(i);
            }
            System.out.println("(" + NUM_CHARS + ")");
        }
    }
```

Correct outer loop: 2 points (possible to lose 1, 2, or 3)
Correctly calculate number of chars per line : 4 points
    or use alternative for adding 2, 2, 4, 4, 6, 6, 8, 8 (see alt solution):
Inner loop using calculated value: 2 points
print statement in inner loop with correct output: 2 points
println after inner loop with (N) output: 2 points
Using conditionals this question, - 2
Using Math methods - 3

alternate solution using variables:

```
        int added = 2;
        int numChars = 0;
        for (int i = 1; i <= n; i++) {
            for (int j = 0; j < numChars; j++) {
                System.out.print(i);
            }
            System.out.println("(" + numChars + ")");
            numChars += added;
            // if this is an even line we add 2 more chars next time
            added += 2 * ((i - 1) % 2);
        }
```

6. Programming: 18 points

```
    public static void rollDie(Scanner s, int tgt) {
        System.out.print("Times to roll: ");
        int numRolls = s.nextInt();
        if (numRolls < 1) {
            numRolls = 10;
        }
        System.out.print("Sides on die: ");
        int sides = s.nextInt();
        if (sides < 3) {
            sides = 6;
        }
        int sum = 0;
        for (int i = 0; i < numRolls; i++) {
            int roll = (int) (Math.random() * sides) + 1;
            sum += roll;
            System.out.println("rolled " + roll);
        }
        System.out.println("Sum of rolls: " + sum);
        if (sum > tgt) {
            System.out.println(tgt + " exceeded");
        } else {
            System.out.println(tgt + " not exceeded");
        }
    }
```

method header correct: 1 point
read in time to roll: 1 point
adjust times to roll if < 1: 2 points
read in number of sides: 1 point
adjust number of sides if < 3: 2 points
cumulative sum variable: 1 point
correct loop for rolls: 3 points
correctly generate rolls of 1 to N where N is sides on die: 4 points (-2 first error)
prints out sum of rolls: 1 point
checks and prints correct message if target exceeded: 2 points

Other:
declare Scanner, -1
extra blank lines, -1 (had to use print to get desired output)
lose count -2

7. Graphics Programming: 15 points

```
    public static void drawFigure(Graphics g, int x, int y,
            int size, int numCircles) {

        g.setColor(Color.ORANGE);
        g.fillRect(x, y, size, size);
        int coordinateChange = size / numCircles;
        for (int i = 0; i < numCircles; i++) {
            if (i % 2 == 0) {
                g.setColor(Color.BLACK);
            } else {
                g.setColor(Color.WHITE);
            }
            g.fillOval(x, y, size, size);
            x += coordinateChange;
            y += coordinateChange;
            size -= coordinateChange;
        }
    }
```

change color for rect: 1 point
fill rect correctly: 1 point
determine change in coordinate and circle size: 2 points
loop for number of circles: 2 points
Set color correctly for current circle: 2 points
fill circle correctly: 2 point
alter x correctly: 2 points
alter y correctly: 1 point
alter size correctly: 2 point

Other:
draw all black then all white circles -> doesn't give desired pattern, -2