CS312 Fall 2017 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, 20 points total

A.	`-1 * 5 - 8 * 5 - 15`	`-60`
B.	`25 / 8 + 12 / 7 + 21 / 6`	`7`
C.	`2 + 15 * 10 - 6 / 10`	`152`
D.	`22 / 10 * 7 % 3 * 10 / 2`	`10`
E.	`13672 % 1000 / 100 + 16 % 4 * 3`	`6`
F.	`3.5 % 3.0 + 2.5 * 3`	`8.0`
G.	`7.0 / 2.0 + 13.0 / 10.0 - 1.0`	`3.8`
H.	`5 * (3 + 2) / (6 - 8) + 4`	`-8`
I.	`7 / 2 + 1.5 + 10 / 4`	`6.5`
J.	`15 / 20 + 20 / 15 + 8 / 10`	`1`
K.	`6 / 10.0 + 6 / 10 + 25 / 100.0`	`0.85`
L.	`3 * 2 + "*" + 2 * 4 + "!"`	`"6*8!"`
M.	`"CS" + 31.2 * 10 + "??"`	`"CS312.0??"`
N.	`3 - 7 + 1 + "PY" + 6 + -3`	`"-3PY6-3"`
O.	`1.5 + 20 / 7 + 13 % 65 * 2 - .2`	`29.3`
P.	`((int) (5.0 / 10)) + (double) 5 / 10`	`0.5` (.5 okay)
Q.	`(int) (.075 * 10 + 3 / 2)`	`1`
R.	`"val" + (2 * 1.5) + (6 % 2)`	`"val3.00"`
S.	`Math.ceil(-.99 * 10)`	`-9.0`
T.	`Math.pow(2.0, 3.0)`	`8.0`

2. Code tracing: 2 points each, 18 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. `25 12 57`
B. `runtime error (caused by a divide by 0)`
C. `4 3.0`
D. `syntax error (trying to assign double to int variable)`
E. `-6 -4`
F. `12"Q" JJ`
G. `60`
H. `121`
I. `470`

3. Method Tracing and Parameters Simulation: 2 points each, 14 points total.
For each part write what the output to the screen will be when the code is run.
First two instances of "answer" counted wrong. _ for space okay.

A. `0 6`
B. `-3 50 -5 10`
C. `3 2 12.0`
D. `5`
E. `eeeeeeeee  (9 e's)`
F. `3 1 -3 -3 1 -3`
G. `2 9 2 3 6 10 0`

4. Programming: 12 points:

```
public static void printUTSquare(int size) {
    for (int line = 0; line < size; line++) {
        int numUs = size - line;
        for (int i = 0; i < numUs; i++) {
            System.out.print("U");
        }
        for (int i = 0; i < line; i++) {
            System.out.print("T");
        }
        System.out.println();
    }
}
```

outer loop with correct bounds: 3 points
loop for Us with correct bounds: 3 points
loop for Ts with correct bounds: 3 points
println at the end of each line: 3 points

5. Programming: 12 points

```
public static int getProducts(Scanner key) {
    System.out.print("Enter the number of pairs: ");
    int numPairs = key.nextInt();
    int sumOfProducts = 0;
    for (int pair = 1; pair <= numPairs; pair++) {
        System.out.print("Enter pair " + pair + ", 1st number: ");
        int num1 = key.nextInt();
        System.out.print("Enter pair " + pair + ", 2nd number: ");
        int num2 = key.nextInt();
        int product = num1 * num2;
        sumOfProducts += product;
        System.out.println(num1 + " * " + num2 + " = " + product);
    }
    System.out.println("Sum of products = " + sumOfProducts);
    return sumOfProducts
}
```

Method header correct (with return of int and Scanner parameter): 1 point
asks and gets number of pairs: 1 point
variable for sum of products: 1 point
loop with correct bounds: 2 points
ask and get first number: 1 point
ask and get second number: 1 point
print product of numbers: 1 point
add product to running total: 2 points
print out sum of products: 1 point
return sum of products: 1 point

6. Programming: 13 points

```
public static void diceGame(int numRolls) {
    int total = 0;
    for (int i = 1; i <= numRolls; i++) {
        int roll = (int) (Math.random() * 10);
        if (roll == 0) {
            total = 0;
        } else if (roll == 1) {
            total *= 2;
        } else {
            total += roll;
        }
        System.out.print("Roll " + i + " is " + roll + ". ");
        System.out.println("points = " + total);
    }
}
```

variable for cumulative sum of points: 1 point
loop for number of rolls with correct bounds: 2 points
correctly pick random number between 0 and 9 using Math.random() and casting: 2 points
option to reset points to 0 if roll a 0: 2 points
option to double points if roll is a 1: 2 points
all other options add points to running total: 1 point
print out correct information for round: 2 points

7. Graphics Programming: 12 points

```
public static void starburst(Graphics g, int size,
                                                int numLines) {
    int mid = size / 2;
    int yChange = size / (numLines - 1);

    for (int i = 0; i < numLines; i++) {
        int otherY = yChange * i;
        g.setColor(Color.BLACK);
        g.drawLine(0, mid, size, otherY);
        g.setColor(Color.ORANGE);
        g.drawLine(0, otherY, size, mid);
    }
}
```

calculate number of lines: 2 points
loop for lines with correct bounds: 2 points (note, can be done with 2 loops)
set color to BLACK for left to right lines: 1 point
set color to ORANGE for right to left lines: 1 point
draw left to right lines (BLACK) in correct locations: 3 points
draw right to left lines (ORANGE) in correct locations: 3 points