CSE142 Sample Midterm
Autumn 2018

1. Expressions, 10 points. For each expression in the left-hand column, indicate its value in the right-hand column. Be sure to list a constant of appropriate type (e.g., 7.0 rather than 7 for a double, Strings in quotes).

Expression	Value
`4 * (2 + 4) - 3 * 5.5`	________________
`6 % 4 + 4 % 6 + 6 % 6`	________________
`15 / 4 / 3.0 - 18 / 5 + (15 / 10.0)`	________________
`3 + 3 + "3 * 3" + (3 + 3) * 3 % 3`	________________
`(11 != 7 + 4 || 3 * 2 <= 39 % 10) == false`	________________

2. Parameter Mystery, 12 points. Consider the following program.

```
public class ParameterMystery {
     public static void main(String[] args) {
          String george = "john";
          String paul = george;
          String john = "yoko";
          String ringo = "drums";
          String pete = ringo;

          sgtPepper("ringo", george, "paul");
          sgtPepper("pete", ringo, "george");
          sgtPepper(john, paul, george);
          sgtPepper(george, "bass", pete);
     }

     public static void sgtPepper(String drums, String guitar, String bass) {
          System.out.println(guitar + " and " + bass + " with " + drums);
     }
}
```

Write the output produced by this program in the box below.

3. If/Else Simulation, 12 points. Consider the following method.

```
public static void ifElseMystery(int a, int b) {
      if (a < b) {
            a++;
      }
      a = a + 3;
      if (a < b) {
            a++;
      } else if (a > b) {
            b++;
      }
      if (a >= b) {
            b = b - 5;
      }
      System.out.println(a + " " + b);
}
```

For each call below, indicate what output is produced.

```
     Method Call                      Output Produced
   ---------------------------------------------------
     ifElseMystery(2, 8);             _______________

     ifElseMystery(6, 10);            _______________

     ifElseMystery(2, 7);             _______________

     ifElseMystery(12, 9);            _______________

     ifElseMystery(7, 7);             _______________

     ifElseMystery(4, 5);             _______________
```

4. While Loop Simulation, 12 points. Consider the following method:

```
public static int mystery(int z) {
      int x = 1;
      int y = 1;
      while (z > 2) {
            y = y + x;
            x = y - x;
            z--;
      }
      return y;
}
```

For each call below, indicate what value is returned:

```
     Method Call                Return Value
   ---------------------------------------------
     mystery(3);                _______________

     mystery(4);                _______________

     mystery(-1);               _______________

     mystery(6);                _______________

     mystery(0);                _______________

     mystery(2);                _______________
```

5. Assertions, 15 points. You will identify various assertions as being either always true, never true or sometimes true/sometimes false at various points in program execution. The comments in the method below indicate the points of interest.

```
public static int mystery(int x) {
      int y = 1;
      int z = 0;
      // Point A
      while (x > y) {
            // Point B
            z = z + x - y;
            x = x / 2;
            // Point C
            y = y * 2;
            // Point D
      }
      // Point E
      return z;
}
```

Fill in the table below with the words ALWAYS, NEVER or SOMETIMES.

	x > y	z > 0	y % 2 == 0
Point A			
Point B			
Point C			
Point D			
Point E			

6. Programming, 15 points. Write a static method called longWords that takes two parameters: a Scanner, input, and an integer, numWords. The method should prompt the user for numWords words using the given Scanner. After each word, if there are more words to be entered, the number of words remaining should be shown. The method should then print the longest word entered and return the total number of characters in all words entered. If two more words are tied for the longest word, the method should report the one entered earliest.

For example, assume the following declaration is made:

```
Scanner console = new Scanner(System.in);
```

Below are some sample calls to longWords and resulting output (user input bold and underlined):

Call	longWords(console, 5)	longWords(console, 9)	longWords(console, 1)
Output	Next word? **apple** 4 more words... Next word? **blueberry** 3 more words... Next word? **watermelon** 2 more words... Next word? **pear** 1 more words... Next word? **grape** Longest word: watermelon	Next word? **alpha** 8 more words... Next word? **bravo** 7 more words... Next word? **charlie** 6 more words... Next word? **delta** 5 more words... Next word? **echo** 4 more words... Next word? **foxtrot** 3 more words... Next word? **golf** 2 more words... Next word? **hotel** 1 more words... Next word? **india** Longest word: charlie	Next word? **elementary** Longest word: elementary
Return value	33	47	10

You may assume that numWords is always greater than 0, and that the user enters a single word with at least one character each time they are prompted. You must exactly reproduce the format of these logs.

Write your solution to problem 6 on the next page.

Write your solution to problem 6 here:

7. Programming, 15 points. Write a static method called diceWar that takes two integer parameters, max and target. The method should simulate a game between two players in which each player rolls a die with max sides and scores a point if they roll at least target. The game is played in a series of rounds, with each player rolling once per round. A minimum of 3 rounds should be played, but if the score is tied after 3 rounds, the game should continue until there is a winner. In each round, the method should print the round number, followed by each player's roll and if they earned a point. The method should also print the number of sides and the target value at the beginning of the game, and the final score and winner at the end.

   Below are some sample calls to diceWar and possible resulting output:

Call diceWar(6, 2)

Output

```
Rolling a 6-sided die, need 2 to score...
Round 1:
    Player 1 rolled: 5 - POINT!
    Player 2 rolled: 4 - POINT!
Round 2:
    Player 1 rolled: 6 - POINT!
    Player 2 rolled: 2 - POINT!
Round 3:
    Player 1 rolled: 4 - POINT!
    Player 2 rolled: 1
Final score - Player 1: 3, Player 2: 2
Player 1 wins!
```

Call diceWar(6, 2)

Output

```
Rolling a 6-sided die, need 2 to score...
Round 1:
    Player 1 rolled: 3 - POINT!
    Player 2 rolled: 5 - POINT!
Round 2:
    Player 1 rolled: 4 - POINT!
    Player 2 rolled: 5 - POINT!
Round 3:
    Player 1 rolled: 4 - POINT!
    Player 2 rolled: 6 - POINT!
Round 4:
    Player 1 rolled: 1
    Player 2 rolled: 5 - POINT!
Final score - Player 1: 3, Player 2: 4
Player 2 wins!
```

Call diceWar(10, 7)

Output

```
Rolling a 10-sided die, need 7 to score...
Round 1:
    Player 1 rolled: 2
    Player 2 rolled: 6
Round 2:
    Player 1 rolled: 2
    Player 2 rolled: 10 - POINT!
Round 3:
    Player 1 rolled: 10 - POINT!
    Player 2 rolled: 9 - POINT!
Final score - Player 1: 1, Player 2: 2
Player 2 wins!
```

Call diceWar(4, 2)

Output

```
Rolling a 4-sided die, need 2 to score...
Round 1:
    Player 1 rolled: 1
    Player 2 rolled: 2 - POINT!
Round 2:
    Player 1 rolled: 3 - POINT!
    Player 2 rolled: 2 - POINT!
Round 3:
    Player 1 rolled: 4 - POINT!
    Player 2 rolled: 1
Round 4:
    Player 1 rolled: 2 - POINT!
    Player 2 rolled: 3 - POINT!
Round 5:
    Player 1 rolled: 3 - POINT!
    Player 2 rolled: 1
Final score - Player 1: 4, Player 2: 3
Player 1 wins!
```

   Notice that in the two examples on the right, more than 3 rounds are played because the score was tied. You may assume that max is always greater than 0, and that target is always greater than 0 and less than or equal to sides. You must exactly reproduce the format of these logs, though the actual output may differ due to randomness.

   Write your solution to problem 7 on the next page.

Write your solution to problem 7 here:

8. Programming, 9 points. Write a static method named countWords that accepts a String, str, as a parameter and that returns the number of words in that String. In this context, a word is defined as any sequence of non-space characters surrounded by spaces. For example, the call countWords("this is CSE 142"); should return 4 because the String that was passed in contains four sequences of characters surrounded by spaces.

   Below are some sample calls to countWords and resulting return values:

Call	**Return value**
countWords("this is CSE 142");	4
countWords("Partyin' partyin' yeah fun fun fun fun");	7
countWords(" T'was brillig, and the *$&*#$ toves ");	6
countWords(" @#$%^$#@ this-and-that ");	2
countWords("a");	1
countWords(" ");	0
countWords("");	0

   Notice that words may be entirely composed of numbers or special characters. Also, the parameter may start or end with spaces.

   You may not construct any objects (Scanners, Strings, etc.) to solve this problem. You may assume that the String contains no whitespace characters other than spaces (i.e. no tabs, newline characters, etc.).

   Write your solution to problem 8 here: