Create method absoluteAdd

> *By: Jimmy Pan*

- Create a method that takes 2 integers, and returns the sum of the absolute value of both integers.

```
System.out.println(absoluteAdd(-5, -7)); // 12
System.out.println(absoluteAdd(6, -2)); // 8
System.out.println(absoluteAdd(-3, 1)); // 4
System.out.println(absoluteAdd(9, 4)); // 13
```

Create method countingByOne

> *By: Jimmy Pan*

- Create a method that takes 2 integers, print out the numbers from the first number to the second number changing by a count of 1.
  - Don't worry about how the output looks. As long as the numbers are printed in the right order.

```
countingByOne(10, 5); // 10, 9, 8, 7, 6, 5
countingByOne(12, 19); // 12, 13, 14, 15, 16, 17, 18, 19
countingByOne(1, 1); // 1
```

Debugging Practice

> *By: Turk Erdin*

- The code below is broken. I have placed the expected output in the comments. Using the debugger tool and everything you've learned about loops, go through and debug this faulty code so it prints as expected.

```
import java.util.*;

public class DebuggingPractice {
    public static void main(String[] arg) {
        Scanner sc = new Scanner(System.in);

        /**
           Expected output:

           How many layers in your pyramid do you want?
           4

           *
           * *
           * * *
           * * * *
```

```
         */
        System.out.println("How many layers in your pyramid do you want?");
        int levels = sc.nextInt();

        System.out.println();

        for (int i=0; i<=levels; i++) {
            for (int j=i-1; j>0; j++) {
                System.out.print("* ");
            }

            System.out.println();
        }

        /**
           Expected output:

           What is your name?
           Turk

           K R U T
         */
        System.out.println("What is your name?");
        String str = sc.nextLine().toUpperCase();
        String str2 = "";

        System.out.println();
        for (int i=str.length(); i<0; i--) {
            str2 += str.charAt(i);
        }

        System.out.println(str2);

        sc.close();
    }
}
```

if/else if/else <-> switch/case

> *By: Jimmy Pan*

- Create a method that repeatedly asks the user for an integer, then output the day of the week that number is. 0 is Sunday, 1 is Monday, 2 is Tuesday, etc. End when the user inputs an invalid day of the week.
    - Write 2 methods, one using if/else if/else statements, and another using switch/case statements.

```
Enter a day of the week (0 - 6): 3
3 is Wednesday
Enter a day of the week (0 - 6): 6
6 is Saturday
```

```
Enter a day of the week (0 - 6): 15
15 is an invalid day of the week
```

Trace nested if/else if/else statements

> *By: Jimmy Pan*

- What will get printed onto the console? No IDE!

```java
public class NestedMysteryCode {
    public static void main(String[] args) {
        nestedMysteryCode(5, 10, 3);
        nestedMysteryCode(7, 2, 7);
        nestedMysteryCode(4, 4, 4);
    }

    public static void nestedMysteryCode(int a, int b, int c) {
        if (a > b) {
            if (b < c) {
                System.out.println("Case 1");
            } else {
                System.out.println("Case 2");
            }
        } else if (a < b) {
            if (b > c) {
                if (c != 0) {
                    System.out.println("Case 3");
                } else {
                    System.out.println("Case 4");
                }
            } else if (b == c) {
                if (a >= 0) {
                    System.out.println("Case 5");
                } else {
                    System.out.println("Case 6");
                }
            } else {
                System.out.println("Case 7");
            }
        } else {
            if (b == c) {
                System.out.println("Case 8");
            } else if (b > c) {
                if (a % 2 == 0) {
                    System.out.println("Case 9");
                } else {
                    System.out.println("Case 10");
                }
            } else {
                if (c > 0) {
                    System.out.println("Case 11");
```

```
                } else {
                    System.out.println("Case 12");
                }
            }
        }
    }
}
```

while <-> do-while <-> for

> *By: Jimmy Pan*

- Create a method that repeatedly asks the user for an integer until either 10 numbers are entered or a negative number is entered. Sum up all the numbers entered by the user, excluding the negative number.
  - Write 3 versions, using a different type of loop each time.
    - Loops: While | Do-While | For

```
Enter a number: 6
Enter a number: 9
Enter a number: 1
Enter a number: 5
Enter a number: 3
Enter a number: 7
Enter a number: 6
Enter a number: 8
Enter a number: 4
Enter a number: 3
Sum of all the numbers you entered is 52
```

```
Enter a number: 3
Enter a number: 4
Enter a number: 9
Enter a number: 1
Enter a number: -2
Sum of all the numbers before the negative number is 17
```

Array Declaration 2D

> *By: Jimmy Pan*

- Problem 1: Create a method that takes a 2D array of characters, and prints them out in a grid pattern.
  - Output must match the sample output (spaces between each symbol and a new line for each row)
- Problem 2: What type and what method did I use to create the `symbols2D` array?
  - Ragged/Jagged Array with Shorthand Notation
  - Ragged/Jagged Array with Multi-Step Approach
  - Standard Array with Shorthand Notation

- Standard Array with Multi-Step Approach

```
char[][] symbols2D = {
    {'!', '@', '#'},
    {'$', '%', '^'},
    {'&', '*', '(', ')'}
}
```

```
! @ #
$ % ^
& * ( )
```

Array Declaration 3D

> *By: Jimmy Pan*

- Problem 1: Create a method that takes a 3D array of characters and print them out as followed.
  - Output must match the sample output (headers for each row, a tab (\t), then the symbols)
  - Hint:
    - Row is the first [2]
    - Each row in the row is the [3]
      - Hence 3 rows
    - The number of symbols per row of each row is the last [2]
- Problem 2: What type and what method did I use to create the `symbols3D` array?
  - Ragged/Jagged Array with Shorthand Notation
  - Ragged/Jagged Array with Multi-Step Approach
  - Standard Array with Shorthand Notation
  - Standard Array with Multi-Step Approach

```
char[][][] symbols3D = new char[2][3][2];
symbols3D[0][0][0] = '!';
symbols3D[0][0][1] = '@';
symbols3D[0][1][0] = '#';
symbols3D[0][1][1] = '$';
symbols3D[0][2][0] = '%';
symbols3D[0][2][1] = '^';
symbols3D[1][0][0] = '&';
symbols3D[1][0][1] = '*';
symbols3D[1][1][0] = '(';
symbols3D[1][1][1] = ')';
symbols3D[1][2][0] = '-';
symbols3D[1][2][1] = '=';
```

```
Row 1:
    ! @
    # $
    % ^
```

```
Row 2:
    & *
    ( )
    - =
```

Random number generation + Search for value

> *By: Jimmy Pan*

- Problem 1: Create a method that takes in an integer and generates an integer array of that size, filled with random numbers from 1 to 100 (inclusive)

```
import java.util.Arrays
```

```
System.out.println(Arrays.toString(generateArray1to100(8)));
System.out.println(Arrays.toString(generateArray1to100(5)));
System.out.println(Arrays.toString(generateArray1to100(12)));
```

- Problem 2: Using the array of random numbers generator from Problem 1, print out the smallest even and odd number, and the largest even and odd number.
  - This problem may be a bit tricky. It is different from a standard find the smallest/largest problem. But a hint that may help is that the array from problem 1 is always numbers from 1 to 100.
  - Think about edge cases! How should your code behave when there are no odd/even numbers?

```
int[] array = generateArray1to100(20);
System.out.println(Arrays.toString(array));
minMaxOddEven(array);
```

Sample Outputs

```
minEven: 8
maxEven: 96
minOdd: 13
maxOdd: 95
```

It is possible to not have even/odd numbers and for the min/max to be the same number!

```
minEven: None
maxEven: None
minOdd: 52
maxOdd: 52
```

Array Search 2D

> *By: Jimmy Pan*

- Problem 1: Create a method that takes in a string array. Then print out all words that are of length 6 or more.

```
String[] words = {
    "apple", "banana", "cherry", "dog", "elephant", "fish",
    "grape", "hat", "igloo", "jacket", "kangaroo",
    "lemon", "monkey", "noodle", "orange", "penguin",
    "quilt", "rabbit", "snake", "tiger", "umbrella",
    "vanilla", "watermelon", "xylophone", "yogurt", "zebra",
    "sun", "moon", "star", "flower", "ocean", "mountain",
    "river", "cloud", "rain", "snow", "fire", "earth",
    "wind", "volcano", "island", "desert", "forest",
    "castle", "dragon", "knight", "princess", "wizard",
    "unicorn"
};
```

- Problem 2: Create a method that takes in a 2D string array. Then print out all words that are of length 6 or more.

```
String[][] words2D = {
    {"apple", "banana", "cherry"},
    {"dog"},
    {"elephant", "fish", "grape", "hat"},
    {"igloo", "jacket", "kangaroo"},
    {"lemon", "monkey"},
    {"noodle"},
    {"orange", "penguin", "quilt", "rabbit", "snake"},
    {"tiger", "umbrella", "vanilla", "watermelon", "xylophone"},
    {"yogurt", "zebra"},
    {"sun", "moon", "star", "flower"},
    {"ocean"},
    {"mountain", "river", "cloud", "rain"},
    {"snow"},
    {"fire", "earth", "wind", "volcano", "island"},
    {"desert", "forest", "castle", "dragon", "knight"},
    {"princess", "wizard"},
    {"unicorn"}
};
```

Expected Out For Both Problems

```
banana
cherry
elephant
igloo
jacket
kangaroo
monkey
noodle
orange
penguin
rabbit
```

```
umbrella
vanilla
watermelon
xylophone
yogurtflower
mountain
volcano
island
desert
forestcastle
dragon
knight
princesswizard
unicorn
```

Managing and manipulating array contents

> *By: Jimmy Pan*

- Problem 1: Create a method that takes in an array and squares each element in the array.
- Problem 2: Create a method that takes the array from Problem 1, and calculate the average.

```
import java.util.Arrays;
```

```
// Problem 1 Test Case
int[] array = {7, 1, 4, 9, 18, 72, 36, 65};
squareUp(array);
System.out.println(Arrays.toString(array)); // [49, 1, 16, 81, 324, 5184, 1296, 4225]

// Problem 2 Test Case
System.out.println(average(array)); // 1397
```

Generate Random Numbers

> *By: Jimmy Pan*

- Create a method that generates a random String of length 6, using only 0-9 and a-f, then add `0x` in front of it. Also generate a random number from 1 to 16777215.
  - Please use the given main method and do not make any modifications to it.
  - Fun fact: `0xfa12ce` is what we call a hexadecimal number.

```
public static void main(String[] args) {
    Object[] numbers = generateRandomNumbers();
    String hex = (String) numbers[0];
    int dec = (int) numbers[1];
    System.out.printf("Random Hexadecimal Number: %s%n", hex);
```

```
    System.out.printf("Random Decimal Number: %d%n", dec);
}
```

Tracing Method Calls

> *By: Jimmy Pan*

- What is the output of this code?

```
import java.util.Arrays;
public class Main {
    public static int firstMethod(int[] arr, int x) {
        int z = 0;
        for (int i = 0; i < arr.length; i++) {
            z = z + arr[i] * x;
        }
        return z;
    }

    public static String secondMethod(int x, String str) {
        String s = "";
        for (int i = 0; i < x; i++) {
            s = s + str;
        }
        return s;
    }

    public static double[] thirdMethod(int x) {
        double[] arr = new double[x];
        for (int i = 0; i < x; i++) {
            arr[i] = x;
        }
        return arr;
    }

    public static void finalMethod(int x, String str, double[] arr) {
        System.out.printf("%d     %s     %s", x, str, Arrays.toString(arr));
    }

    public static void main(String[] args) {
        finalMethod(firstMethod(new int[]{5, 7, 35, 52}, 3), secondMethod(3, "xyz"),
thirdMethod(5));
    }
}
```

Complete The Coin Machine

> *By: Jimmy Pan*

- Complete the coin machine class.

- Output does not need to match.
- The amount the coins add up to much be correct.
- The output format must be kept.
  - Please use the given method `changeGiven`

```
public class Main {
    public static void main(String[] args) {
        CoinMachine cm = new CoinMachine(50, 50, 50, 50);

        cm.giveChange(50.00, 49.99);
        cm.giveChange(35.00, 27.99);
        cm.giveChange(32.00, 25.00);
        cm.giveChange(1.23, 0.95);
        cm.giveChange(0.75, 0.55);
        cm.giveChange(33.33, 27.75);
        cm.giveChange(0.92, 0.50);
        cm.inStorage();
    }
}

public class CoinMachine {
    private int quarter;
    private int dime;
    private int nickel;
    private int penny;

    public CoinMachine() {
        this.quarter = 0;
        this.dime = 0;
        this.nickel = 0;
        this.penny = 0;
    }

    public CoinMachine(int quarter, int dime, int nickel, int penny) {
        this.quarter = quarter;
        this.dime = dime;
        this.nickel = nickel;
        this.penny = penny;
    }

    public void giveChange(double payment, double price) {
        if (payment < price) {
            System.out.println("Customer did not pay enough.");
            return;
        }

        // keep count of amount of each coin that is going to be given to the user
        int q = 0;
        int d = 0;
        int n = 0;
        int p = 0;
```

```
        // payment - how much the user pays
        // price   - their total for the purchase
        /*****************************************/
        // TODO: calculate the amount of coins needed

        /*****************************************/
    }

    private void changeGiven(int quarter, int dime, int nickel, int penny) {
        System.out.println("Coins used:");
        System.out.printf("%d Quarter(s)%n", quarter);
        System.out.printf("%d Dime(s)%n", dime);
        System.out.printf("%d Nickel(s)%n", nickel);
        System.out.printf("%d Penny(s)%n", penny);
    }

    public void inStorage() {
        System.out.printf("%-10s%d%n", "25¢", this.quarter);
        System.out.printf("%-10s%d%n", "10¢", this.dime);
        System.out.printf("%-10s%d%n", "5¢", this.nickel);
        System.out.printf("%-10s%d%n", "1¢", this.penny);
    }

    public void addQuarter(int count) {
        if (count < 0) {
            System.out.println("Negative not allowed.");
        } else {
            this.quarter = this.quarter + count;
        }
    }

    public void addDime(int count) {
        if (count < 0) {
            System.out.println("Negative not allowed.");
        } else {
            this.dime = this.dime + count;
        }
    }

    public void addNickel(int count) {
        if (count < 0) {
            System.out.println("Negative not allowed.");
        } else {
            this.nickel = this.nickel + count;
        }
    }

    public void addPenny(int count) {
        if (count < 0) {
            System.out.println("Negative not allowed.");
        } else {
```

```
            this.penny = this.penny + count;
        }
    }
}
```

Sample Expected Output

```
Coins used:
0 Quarter(s)
0 Dime(s)
0 Nickel(s)
1 Penny(s)
Coins used:
28 Quarter(s)
0 Dime(s)
0 Nickel(s)
1 Penny(s)
Coins used:
22 Quarter(s)
15 Dime(s)
0 Nickel(s)
0 Penny(s)
Coins used:
0 Quarter(s)
2 Dime(s)
1 Nickel(s)
3 Penny(s)
Coins used:
0 Quarter(s)
2 Dime(s)
0 Nickel(s)
0 Penny(s)
Coins used:
0 Quarter(s)
31 Dime(s)
49 Nickel(s)
3 Penny(s)
Coins used:
0 Quarter(s)
0 Dime(s)
0 Nickel(s)
42 Penny(s)
25¢       0
10¢       0
5¢        0
1¢        0
```

HAS-A

> *By: Jimmy Pan*

- Figure out the output of this code without the use of an IDE.

```
public class Main {
    public static void main(String[] args) {
        A a = new A(17);
        B b = a.riddle();
        b.answer();
    }
}

class A {
    private B b;
    private int c;

    public A() {
    }

    public A(int c) {
        this.b = new B(c);
        this.c = c * 2;
    }

    public B riddle() {
        System.out.println("My number is twice that of the other number.");
        System.out.println("Added together, our sum is " + (this.c + this.b.getC()) +
".");
        System.out.println("What are our numbers?");
        return this.b;
    }
}

class B {
    private int c;

    public B() {
    }

    public B(int c) {
        this.c = c;
    }

    public int getC() {
        return this.c;
    }

    public void answer() {
        System.out.println("Our numbers are " + this.c + " and " + this.c * 2 + ".");
    }
}
```

IS-A

> *By: Jimmy Pan*

- Figure out the output of this code without the use of an IDE.

```
public class Main {
    public static void main(String[] args) {
        A a = new A();
        B b = new B();
        A ab = new B();
        B ba = (B) ab;

        a.one();
        a.two();
        b.one();
        b.two();
        ab.one();
        ab.two();
        ba.one();
        ba.two();
    }
}

class A {
    public A() {
    }

    public void one() {
        System.out.println("I am a statement.");
    }

    public void two() {
        System.out.println("I am another statement.");
    }
}

class B extends A {
    public B() {
    }

    @Override
    public void one() {
        System.out.println("I am one more statement.");
    }
}
```

(Trick or Treat) and Trace!

> *By: Jimmy Pan*

- Figure out the output of this code without the use of an IDE.

```
public class Main {
    public static void main(String[] args) {
        Halloween you = new Halloween();
        you.buyCandy(50);
        for (int i = 1; i <= 5; i++) {
            you.giveCandy(i * ((i % 2 == 0) ? 3 : 4));
        }
        System.out.printf("I end the night with %d piece(s) of candy.%n",
you.getCandy());
    }
}

public class Halloween {
    private int candy;

    public Halloween() {
        this.candy = 0;
    }

    public Halloween(int candy) {
        this.candy = candy;
    }

    public int getCandy() {
        return this.candy;
    }

    public int buyCandy(int amount) {
        if (amount < 0) {
            System.out.println("Silly you, you can't buy negative pieces of candy
:)");
        } else {
            System.out.printf("You went out and bought %d piece(s) of candy.%n",
amount);
            this.candy = this.candy + amount;
        }

        return this.candy;
    }

    public int giveCandy(int amount) {
        if (amount < 0) {
            System.out.println("Silly you, you can't give negative pieces of candy
:)");
        } else {
            System.out.printf("You gave out %d piece(s) of candy.%n", amount);
            this.candy = this.candy - amount;
        }

        return this.candy;
    }
```

```
    public void trickOrTreat(boolean treat) {
        if (treat) {
            System.out.printf("Here are %d pieces of candy!%n", 5);

        } else {
            System.out.printf("You have been tricked and had %d pieces of candy stolen
from you :(%n", 2);
        }
    }
}
```