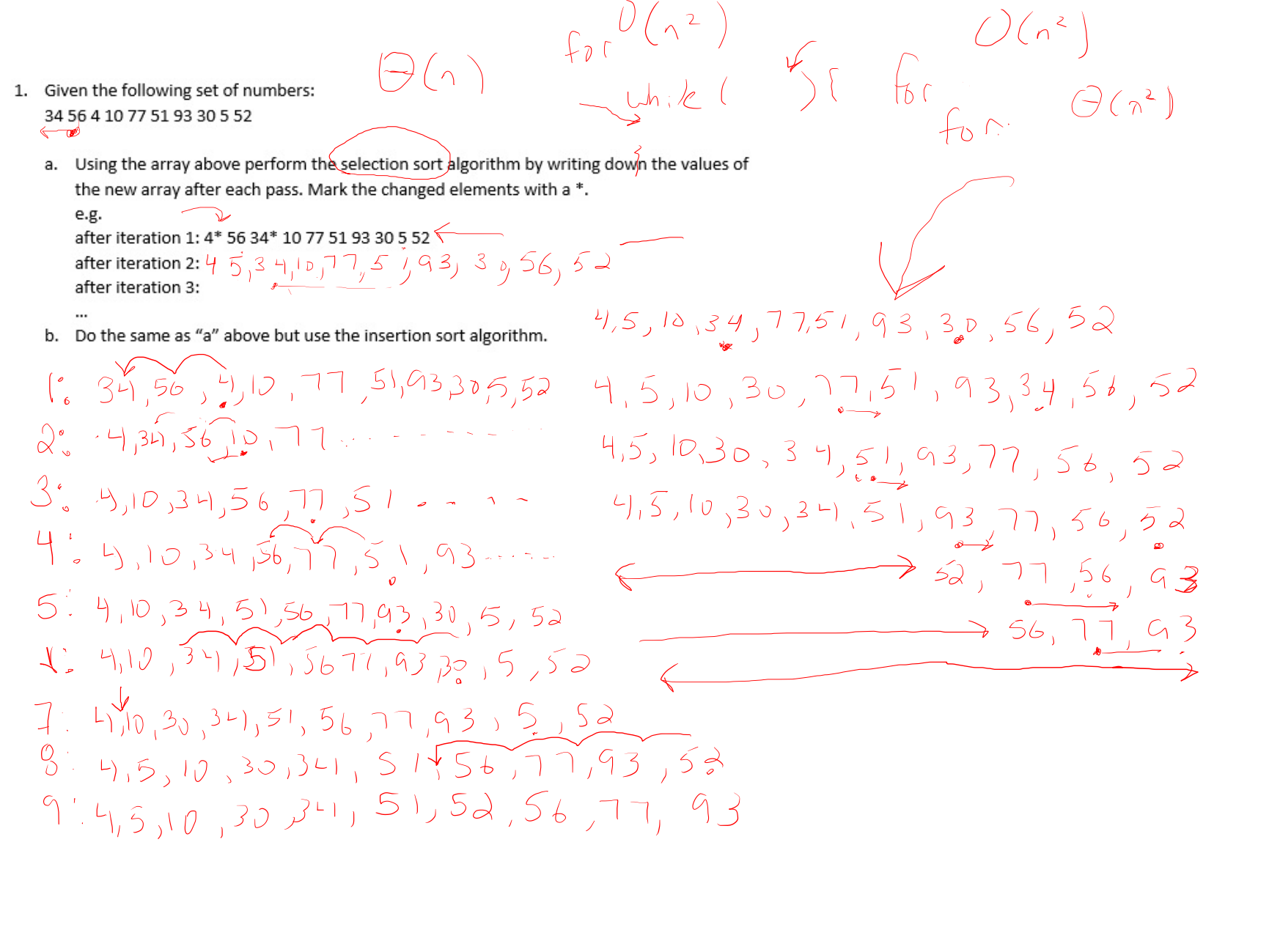
1. Given the following set of numbers:
   34 56 4 10 77 51 93 30 5 52

   a. Using the array above perform the selection sort algorithm by writing down the values of the new array after each pass. Mark the changed elements with a *.
      e.g.
      after iteration 1: 4* 56 34* 10 77 51 93 30 5 52
      after iteration 2:
      after iteration 3:
      ...

   b. Do the same as "a" above but use the insertion sort algorithm.



Write code or Write a code segment. - Don't create class. Don't create method.

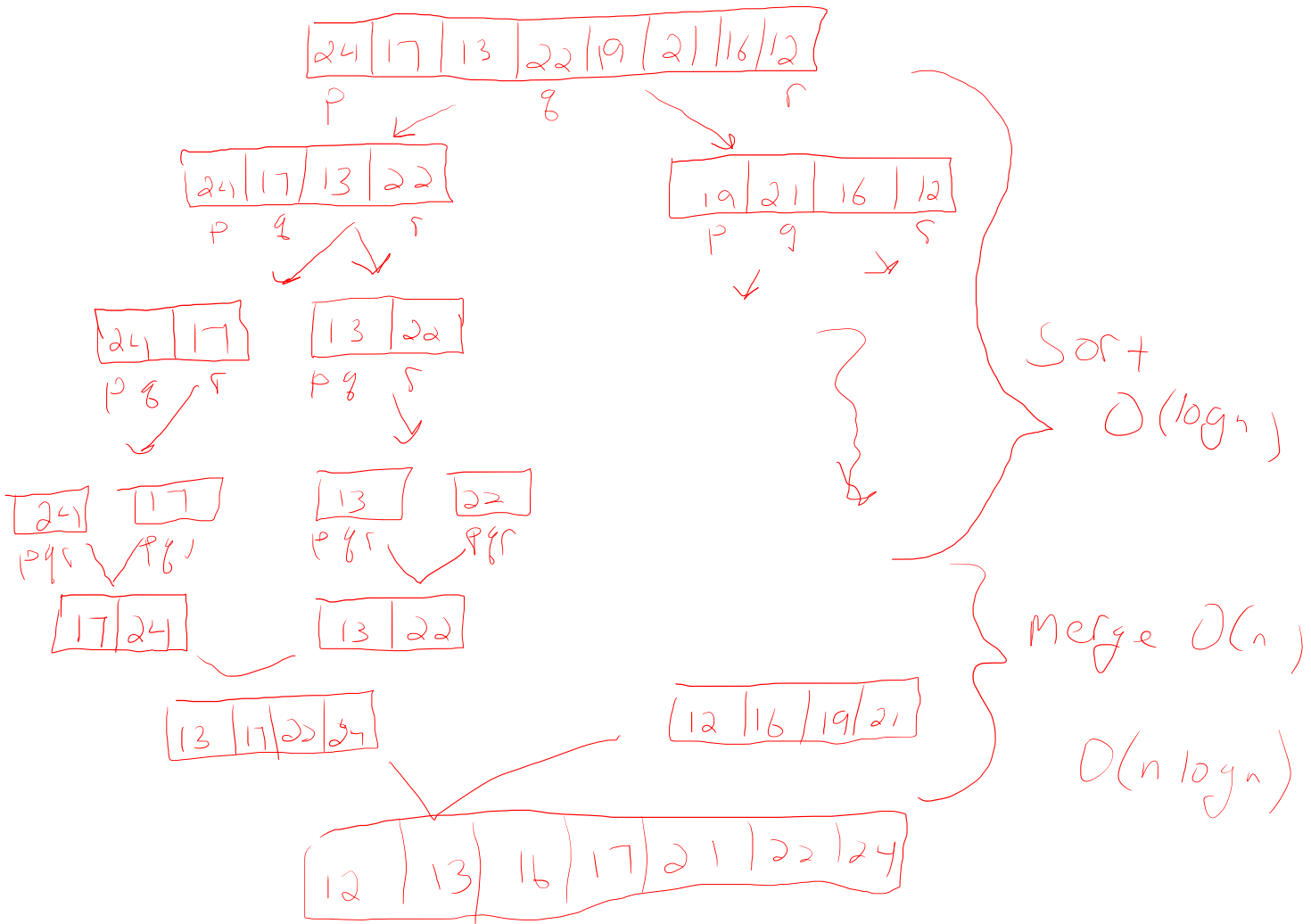
2. Given the code below:

```
int[][] array1 = new int[30][30];
int[][] array2 = new int[30][30];
```

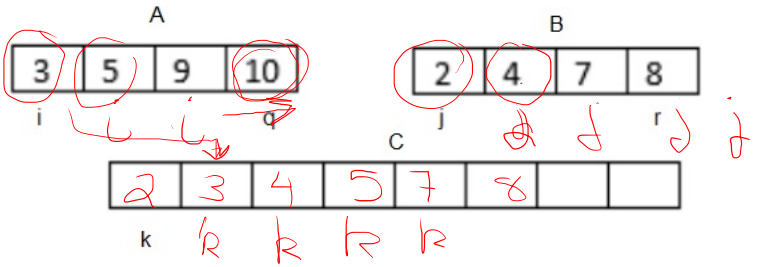
Write code that will create a third array and populate the values of each element in the third array with the elements of array1 multiplied with array2. e.g. array3[0][0] = array1[0][0]*array2[0][0];

```
array3[0][0] = array1[0][0]*array2[0][0];
array3[0][1] = array1[0][1]*array2[0][1];
array3[0][2] = array1[0][2]*array2[0][2];

//N x M
//int[][] array3 = new int[30][30];
//int[][] array3 = new int[array1.length][array1[0].length];
// number of rows array1.length
// number of cols array1[0].length

int[][] array3 = new int[N][M];

for (int row=0; row < N; row++) {
      for (int col= 0; col < M; col++) {
            array3[row][col] = array1[row][col]*array2[row][col];
      }
}
```

3. Given an array A has the values 24, 17, 13, 22, 19, 21, 16, 12
Describe using a diagram how the Merge Sort algorithm would sort the algorithm by recursively dividing "A" into 2 parts until there is only one element for each sub array and then how it merges the array back together. For reference look at the khan academy website on its overview of merge sort. Located here https://www.khanacademy.org/computing/computer-science/algorithms/merge-sort/a/overview-of-merge-sort
Look at the diagram. Don't forget to label the p,q and r indexes for each sub array as it is divided and merged.



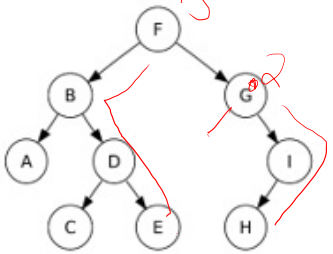
4. In PSUEDO code write the merge part of the merge sort algorithm. Use the arrays below as an example. Note, these two arrays A and B are sorted. At the end of your algorithm the two arrays should be sorted into the C array. I've labeled the indices of each array with variables to help you get started. Don't write the while loop first. First, just think about what value would go into the kth position.




```
// assum A and B are Created
int[] C = new int[A.length + B.length];
int i = 0;
int q = A.length-1;
int j = 0;
int r = B.length - 1;
int k = 0;

while(j <= r && i <= q) {
        if (A[i] < B[j]) {
                C[k] = A[i];
                 i++:
        } else {
                C[k] = B[j];
                j++;
        }
        k++;
}
```

```
if (i <= q) {
        //values remaining in A that need to be copied
        for (int x=i; x <= q; x++)
             C[k++] = A[x];
}
if (j <= r) {
        // copy values from B  to C
        for (int x=j; x <=r; x++)
            C[k++] = A[x];
}
/// alternate solution
//values remaining in A that need to be copied
while(i <= q)
     C[k++] = A[i++];

// copy values from B  to C
while (j <= r)
     C[k++] = A[j++];
```

5. Given the tree:



In-order

A B C D E F G H I

pre-order

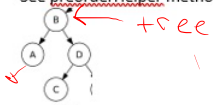
F B A D C E G I H

post order

A C E D B H I G F

a. Show the output of in-order, pre-order and post-order traversal.

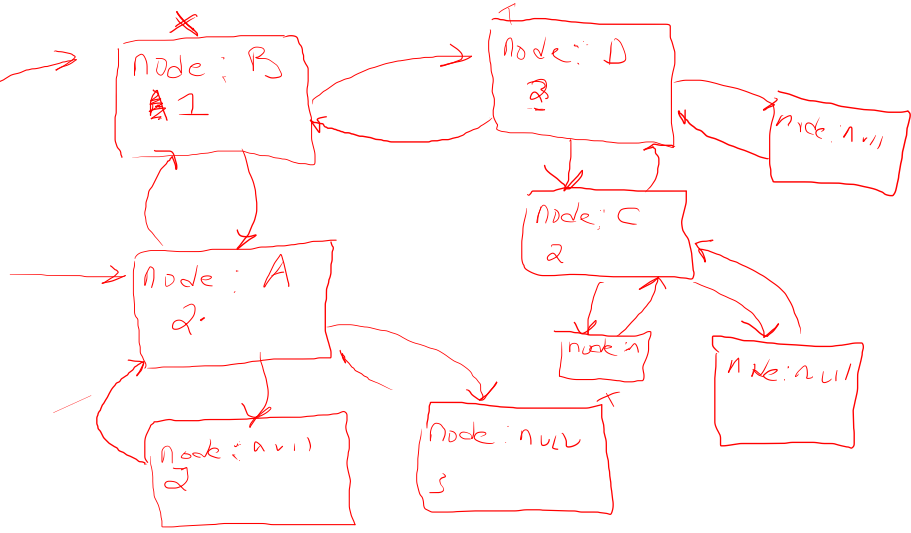
6. Draw a recursive method call stack, with stack frames, e.g. a recursive algorithm that has two recursive calls in the non base case code. A good example is a pre-order traversal binary tree. See preorderHelper method below. Use this tree to trace your call stack:



tree

```
Main() {
    //tree variable is pointing to the "B" node above.
    preorderHelper(tree);

}
```

```
// recursive method to perform preorder traversal
private void preorderHelper(TreeNode<E> node) {
    if (node == null) {
        return;
    }

    System.out.printf("%s ", node.data); // output node data
    preorderHelper(node.leftNode); // traverse left subtree
    preorderHelper(node.rightNode); // traverse right subtree
}
```



Console

B A D C

Code Problem

7. Write a code segment that prompts the user for a String and prints the value to the output. Wrap your code in a try/catch block that catch's Exception, and exits if an exception occurs.

8. Write a java method, called getMonthString, that accepts as a parameter the month of the year as an integer (1-12). The method should return a String that represents the month. For example, if the method is called like: String m = getMonthString(2);

The variable m should have the string "February".

Implement this using a switch statement.

Implement this using an if/else structure.

```
public static String getMonthString(int month) {
        String str = "";
        switch(month) {
                case 1:
                        str = "January";
                        break;
                case 2:
                        str = "February";
                .....   break;
                case 12:
                        str "December";
                        break;

        }
        return str;
}
```

```
try {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter String: ");
        String value = input.next();//block waiting for user to input
        System.out.println("value = "+value);
} catch(Exception exc) {
        System.out.println("error");
        System.exit(1);
}
```

```
public static String getMonthString(int month) {
        if (month == 1)
                return "January";
        else if (month == 2)
                return "Feb..";

        else if (month == 12)
                return "December:


        return "";

}
```

9. Write a java method called getDayOfWeek, that accepts as a parameter the day of the month (1-31). The method should return an integer that represents the day of the week 1-7. To make this easy we will assume the first day of the week always falls on Sunday. For example,

Think of this table that shows the day for each month and the day of week it falls on

Sunday(1)	Monday(2)	Tues(3)	Wed(4)	Thurs(5)	Friday(6)	Saturday(7)
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	34	25	26	27	28
29	30	31				

If the method is called like: int day = getDayOfWeek(25); The variable day would be populated with 4, because the 25th day falls on a Wednesday. DON'T FORGET THE MODULUS OPERATOR!

```
public static int getDayOfWeek(int dayOfMonth) {
        if (dayOfMonth%7 == 0) {
                return 7;
        }
        return dayOfMonth%7;
}
public static int getDayOfWeek(int dayOfMonth) {

        return (dayOfMonth-1)%7 + 1;
}
```