Algorithm Analysis 2

The following method finds a value in an array. If the value is present, the method returns in the index of the first occurrence of the value. If the value is not present, the method returns -1.

```
int find(int[] array, int value) {
    int i = 0;
    boolean found = false;
    int index = -1;

    while (!found && i < array.length) {
        if (array[i] == value) {
            found = true;
            index = i;
        } //end if

        i = i + 1;
    } //end while

    return index;
}
```

Model 1

Question 1. Suppose the array `a` contains the values {`17, 11, 2, 48, 13, 2, 19`}. What does the method invocation `find(a, 2)` return?

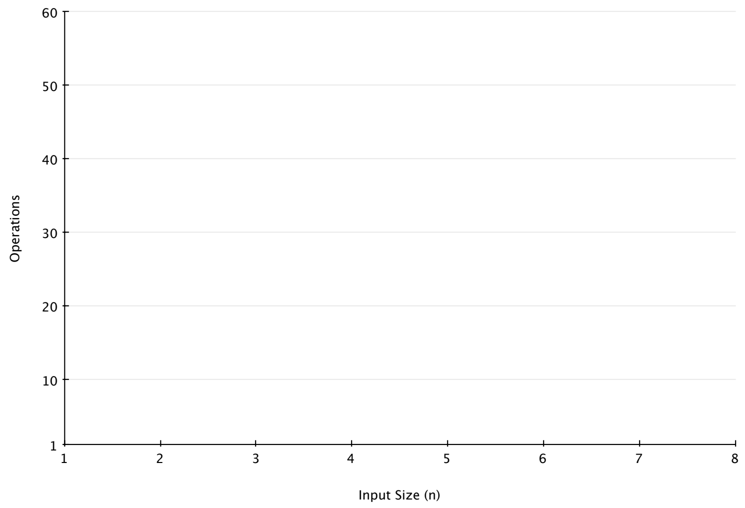
Question 2. What does the method invocation `find(a, 7)` return?

Question 3. In the table below each row corresponds to a single method invocation of the `find` method. Complete the table. As in the Algorithm Analysis 1 worksheet, an operation is either an assignment, comparison, or arithmetic operation.

Invocation	`a`	`value`	# Operations Executed
`find(a, value)`	{2, 41, 6}	2	
`find(a, value)`	{17, 11, 4, 20, 19}	17	
`find(a, value)`	{9, 27, 58, 1, 7, 16, 4}	9	

Question 4. For *any* array `a`, what value of `x` results in the *least* number of operations being executed during the invocation `find(a, x)`? Why?

Question 5. Plot the number of operations from the table above. Based on your data, is the least number of operations dependent on the size of the input? Why or why not?



Question 6. As with the previous table, each row of the table below corresponds to a single method invocation of the `find` method. Complete the table.

Invocation	`a`	`value`	# Operations Executed
`find(a, value)`	{2, 41, 6}	6	
`find(a, value)`	{17, 11, 4, 20, 19}	19	
`find(a, value)`	{9, 27, 58, 1, 7, 16, 4}	4	

Question 7. For any array `a`, what value of `x` results in the *most* number of operations being executed during the invocation `find(a, x)`? Why?

Question 8. As with the first table, plot the number of operations in the table above on the graph on the previous page of this worksheet. Based on your data, is the most number of operations dependent on the size of the input? Why or why not?

Question 9. `find` is $\Omega(1)$. Complete the following sentence:

A method is $\Omega(1)$ if...

Question 10. `find` is $O(n)$. Complete the following sentence:

A method is $O(n)$ if...

Question 11. In complete sentences, explain what it means for a method to be $\Omega(n)$ and $O(n^2)$.

Question 12. Suppose a method is $\Omega(n)$ and $O(n)$. Is the method $\Theta(n)$? Why or why not?

Analyze the methods below and answer the associated questions.

```
boolean inRange(int[] array, int low, int high) {
    for (int i = 0; i < array.length; i++) {
        if (array[i] < low || array[i] > high) {
            return false;
        } //end if
    } //end for

    return true;
}
```

Question 13. `inRange` is $\Omega($ $)$ and $O($ $)$. Justify your answer.

```
void bubbleSort(int[] array) {
    boolean isSorted = true;
    int temp;

    // check to see if array is already sorted
    for (i = 0; i< array.length; i++) {
        if (array[i] > a[i + 1] {
            isSorted = false;
        } //end if
    } //end for

    if (!isSorted) {
        // if it is not sorted, sort the array
        for (i = 0; i < array.length; i++) {
            for (j = 0; j < array.length; j++) {
                if (array[j] > array[j + 1] {
                    temp = array[j];
                    array[j] = array[j + 1];
                    array[j + 1] = temp;
                } //end if
            } //end for
        } //end for
    } //end if
}
```

Question 14. `bubbleSort` is $\Omega(\quad)$ and $O(\quad)$. Justify your answer.