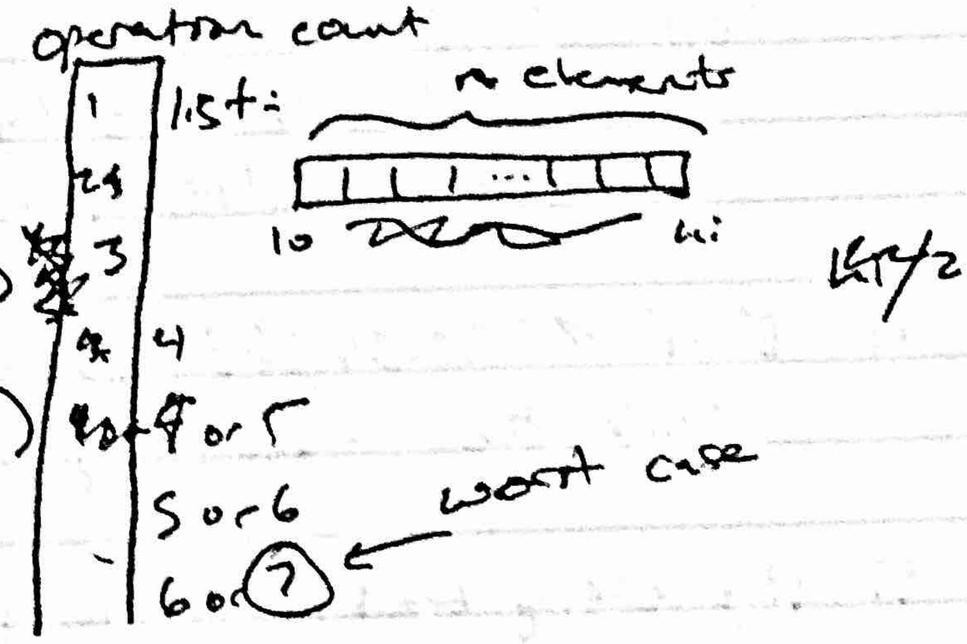
Assignment 1:

1. a) Procedural abstraction is the process of hiding the implementation of certain functions and/or variables from a programs users by breaking up a program's code into parts based on individual tasks and subprocesses.

b) One approach would be to use an "assert" function to assert that input data should meet certain conditions.

2
```
while (low ≤ high) {
    mid = (low + high) / 2;
    if (target < list [mid])
        high = mid - 1;
    else if (target > list [mid])
        low = mid + 1;
    else break;
}
```

operation count



Each time the loop iterates the search area for list is cut in half. Thus, for n elements in the worst case, we would keep n divide by 2 until it's one. Therefore, we'd have $\log_2(n)$ as a approximate worst case. Thus, the time complexity can be expressed as $O(\log n)$.

3. int fun (int n) {
    int count=0;
    for (int i=n; i>0; i/=2) ← runs for O(log n)
        for (int j=0; j<i; j++;)] runs for a complexity of O(n)
            count+=1;
    return count;
}

In cases of nested loops, time complexities are multiplied. So, the for loop that iterates from 0 to i has a TC of O(n). Because the loop outside iterates from n to 0 by dividing by two, we have O(log n). Thus, the two multiply and we get O(n log n).

4. While counting each digit, divide the number by 10 until that number is less than 0.

5. By setting breakpoints in your code, you can stop/pause running code to monitor variable values. In addition, you can step through your code to ensure you have proper flow of logic. If you step into a function, you can follow the debugger into the code of that function to verify that you're passing the correct parameters (for instance). If you step out of the function you will return to the line after where you stepped into the function.

6. Properties of a constructor:
   - they should be declared in the public section
   - they have no return type
   - they are invoked when a new object is created
   - they can have input parameters
   - has the same name as the class

7. A class is a declared map of functions and variables that make up a specific abstract data type. An object is an instance of a class. Before an object is created, no memory is allocated for the class. When an object is instantiated, memory becomes allocated.

8. 1. Test the extreme values for possible test data.
   2. Use the debugger to track variables and verify proper processing of information.

9. 1. Use the scope resolution operator for a class with a declared namespace.

header

```
namespace example {
    class myClass {
        ...
    }
}
```

main.cpp

```
example:: function();
```

common example:

```
std:: cout << "Hello";
```

2. Declare a class within a namespace and use the namespace in implementation.

header

main.cpp

```
using namespace example;
```

3. Declare a class in a namespace and use the namespace containment implementation:

header	main.cpp.

see previous	using namespace example;
	namespace example {
	// code goes here...
	}

10. a) ~~Compiler error~~ There is no output; but it does wait for the user to input something in the console before returning 0.

b) The console will return an error upon compiling because "x" by default is a private member of Test.

b) ~~There is no output~~ No errors. The program is returned with exit code: 0.