(1) Student and Computing Information

- Nicholas Phillips and Student ID: A031344011

- Advance Programming Principles and Assignment 2.

- Completed on a MacBook Pro running macOS Monterey version 12.1. The compiler used was visual studios.

(2) Purpose Statement: The purpose of this assignment was the use of dynamic array structure, function, pointers, binary, and bubble sort. This assignment takes in at least 15 input numbers from the user. If the user enters in less than 15 numbers the program will not run. Once 15 numbers are entered, the function will read in the numbers into a dynamic array structure in the read function. Next, the display function will take in the numbers from the read function and display them. Then, the bubble sorting function will then sort the input numbers in ascending order and then display them into the display function. Lastly, now that the function is sorted we will use the binary function to search the sorted numbers and will search for a user input number. If the number is not in the array, it will display , not found.

(3) C++ Code

```
/*

Completed by Nicholas Phillips on Feburary 24th 2022.

Student ID: A01344011

Assignment 2 for CPSC 246 taught by Dr. Lee.

Completed on a MacBook Pro running macOS Monterey version 12.1. The compiler used was visual studios.

Purpose: The purpose of this assignment was the use of dynamic array structure, function, pointers,

binary, and bubble sort. This assignment takes in at least 15 input numbers from the user. If the user

enters in less than 15 numbers the program will not run. Once 15 numbers are entered, the function will read

in the numbers into a dynamic array structure in the read function. Next, the display function will take
```

```
in the numbers from the read function and display them. Then, the bubble sorting function will then sort the

input numbers in ascending order and then display them into the display function. Lastly, now that the function

is sorted we will use the binary function to search the sorted numbers and will search for a user input number. If the number is not

in the array, it will display , not found.


*/


#include <iostream>

#include <iomanip>

using namespace std;


int Binary_searching(double count[], int maxnum, int key);//Function prototype to search for a key-value.

void Bubble_sorting(double count[], int maxnum);//Function prototype to bubble sort the user input numbers.

void display();//Function prototype to display the unsorted and sorted numbers from the user.

void read();//Function prototype to read the input of the users's numbers.


int main() {




display();//Calls to the function display to display the unsorted and sorted numbers from the user.


}
```

```
//Binary search function, searches the sorted array from the bubble sort.

int Binary_searching(double count[], int maxnum, int key) {

   int start=1, end=maxnum;

   int mid=(start+end)/2; //midpoint of the array

   while(start<=end&&count[mid]!=key){

       if(count[mid]<key){

         start=mid+1;

       }

       else{

         end=mid-1;

       }

      mid=(start+end)/2;

    }

   if(count[mid]==key)//sees  if  the  midpoint  is  equal  to  the  key  value  the  user
searches

   return mid;

   else

   return -1; //returns false

}


//Sorting algorithm that is comparison-based algorithm in which each pair of adjacent
elements is compared

//and the elements are swapped if they are not in order.

void Bubble_sorting(double count[], int maxnum){

bool exchanges;

do{

 exchanges = false; //assume no exchanges

 for(int i = 0; i < maxnum - 1; i++){

  if(count[i] > count[i + 1]){
```

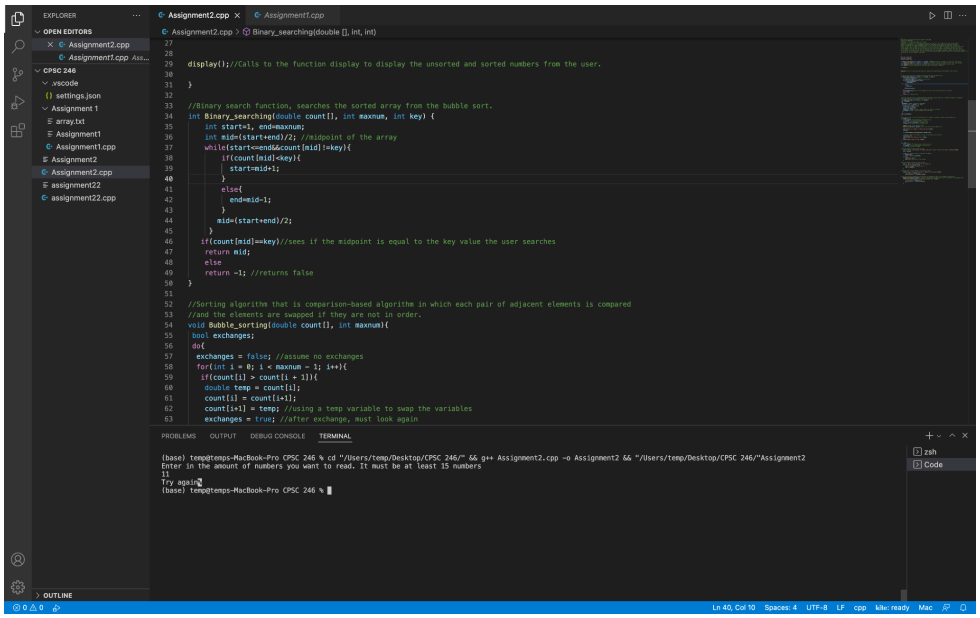
```
   double temp = count[i];

   count[i] = count[i+1];

   count[i+1] = temp; //using a temp variable to swap the variables

   exchanges = true; //after exchange, must look again

  }

 }

}while(exchanges);

}


//Display function is used to display the user's numbers, unsorted, sorted, and
searched value.

void display () {

   double *count; //To dynamically allocate an array

   int maxnum; //To hold the number of inputs

   count = new double[maxnum]; //Dynamic array

   int key;//User search value for binary sort


   read();//Calls to the function read.

   Bubble_sorting(count, maxnum);//Calls from the function bubble sort.


   cout<<"Enter in a number to search the array"<<endl;

   cin>>key;


   int results=Binary_searching(count, maxnum, key);


   //Checks to see if the number is found in the binary sort function.

   if(results==-1)

   cout<<"Number not found in array "<<endl;
```

```
   else
   cout<<"Number found in array "<<endl;
}


void read () {
   double *count; //To dynamically allocate an array
   int maxnum; //To hold the number from the user.
   int x;//Loop control variable


//Get the number of inputs from the user
    cout << "Enter in the amount of numbers you want to read. It must be at least 15
numbers"<<endl;
   cin >> maxnum;


   //Checks if the user enters in atleast 15 numbers
   if (maxnum>=15) {
       count = new double[maxnum];
   } else {
       cout<<"Try again";
       exit (0);//Exits out of the program
   }


//Get the numbers entered in from the user.
   cout << "Enter in the numbers for the array.\n";
   for (x = 0; x < maxnum; x++){
       cout << "Number " << (x + 1) << ": ";
       cin >> count[x];
   }
```

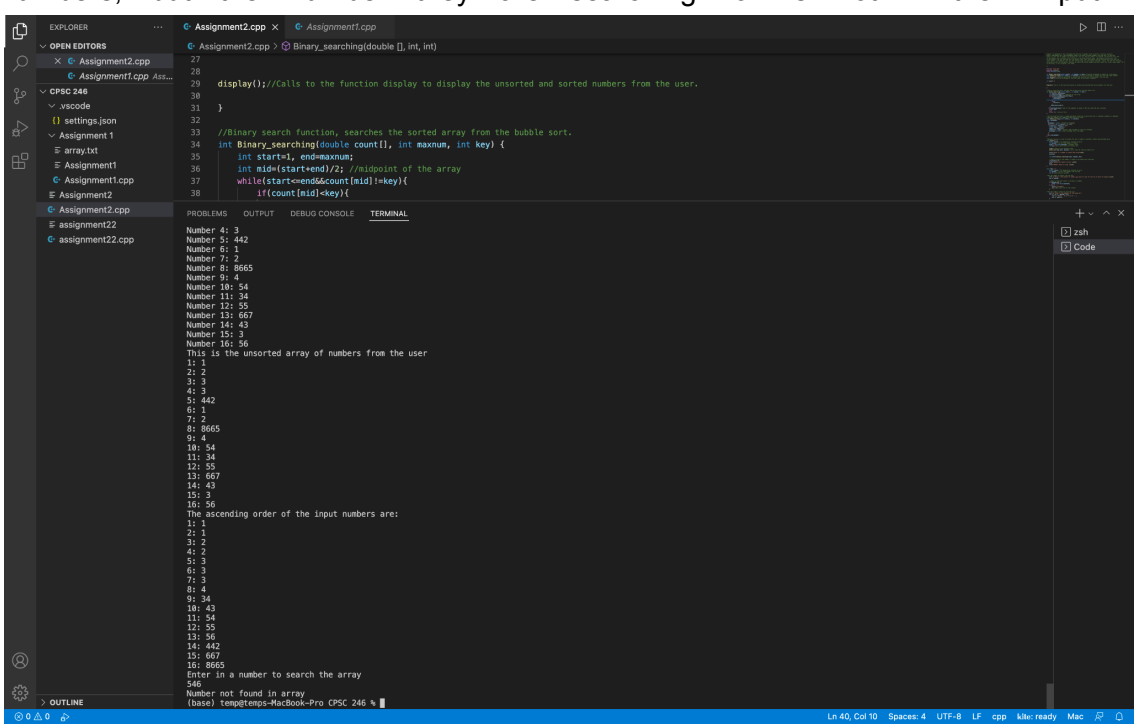
```
//Displays the unsorted numbers from the user input.

   cout<<"This is the unsorted array of numbers from the user"<<endl;

       for (x=0; x < maxnum; x++)

       cout <<(x+1)<<": "<<count[x]<<endl;


//Calls the function bubbleSort and displays the ascending order of the numbers from
the user.

     Bubble_sorting(count,maxnum); //Calls to the bubble sort function to sort the
numbers in ascending order.

   cout<<"The ascending order of the input numbers are: "<<endl;

       for (x = 0; x < maxnum; x++){

       cout<<(x+1)<<": "<<count[x]<<endl;

       }

   }
```

(4) Output from the code.

The first screen shot shows what happens when the user doesn't enter at least 15 or more numbers into the program.



The second screen shot shows what happens when the program is run with at least 15 numbers, but the number they are searching for is not in their input numbers.



The third screen shows what happens when at least 15 numbers are being given by the user and if the number they are searching is in their input numbers.