Assignment

Submission deadline: 6th week during sessional class

You will also face an online based on the offline assigned to you during the assignment submission. For example, if you are roll 30 from A1, your offline is Searching and you will face an online based on Searching during the sessional class.

Subsection	Roll Mod 3	Problem
A1	0	Searching
A1	1	Bubble Sort
A1	2	Heap Sort
A2	0	Selection Sort
A2	1	Counting Sort
A2	2	Calculator
B1	0	Searching
B1	1	Bubble Sort
B1	2	Heap Sort
B2	0	Selection Sort
B2	1	Counting Sort
B2	2	Calculator

0. Searching:
In this problem you will implement a binary searching algorithm. The program will input an array i.e., consecutive 16 bit integers separated by space (maximum 16 integers will be given as input. The array may/may not be sorted) from the user until the user inputs “x”. Next the user will input another 16 bit integer, **n**. The program will output whether the number **n** is present in the array. The program will keep taking **n,** until the character ‘**x’** is passed as an input.

```
Enter Input Array: 3 5 1 6 -10 3 78 36 86 35 30 x
Enter n: 3
3 was found in the array
Enter n: 5
5 was not found in the array
Enter n: 78
78 was found in the array
Enter n: x
Thanks !!!
```

1. Bubble sort:

In this problem you will implement a Bubble Sort algorithm. The program will input an array i.e., consecutive 16 bit integers separated by space (maximum 16 integers will be given as input.) from the user until the user inputs the character “x”. The program will sort the array in ascending order and print in the console.

```
Enter Input Array: 3 5 1 6 -10 3 78 36 86 35 30 x
Sorted Array: -10 1 3 3 5 6 30 35 36 78 86
```

2. Heap sort:

In this problem you will implement a Heap Sort algorithm. The program will input an array i.e., consecutive 16 bit integers separated by space (maximum 16 integers will be given as input.) from the user until the user inputs the character “x”. The program will sort the array in descending order and print in the console.

```
Enter Input Array: 3 5 1 6 -10 3 78 36 86 35 30 x
Sorted Array: -10 1 3 3 5 6 30 35 36 78 86
```

3. Selection sort:

In this problem you will implement a Selection Sort algorithm. The program will input an array i.e., consecutive 16 bit integers separated by space (maximum 16 integers will be given as input.) from the user until the user inputs the character “x”. The program will sort the array in descending order and print in the console.

```
Enter Input Array: 3 5 1 6 -10 3 78 36 86 35 30 x
Sorted Array: -10 1 3 3 5 6 30 35 36 78 86
```

4. Counting sort:

In this problem you will implement a Counting Sort algorithm. The program will input an array i.e., consecutive integers separated by space (the integers will have range 0 to 256, maximum 16 integers will be given as input.) from the user until the user inputs the character “x”. The program will sort the array in ascending order and print in the console.

```
Enter Input Array: 3 5 1 6 3 78 36 86 35 30 x
Sorted Array: 1 3 3 5 6 30 35 36 78 86
```

5. Calculator:

In this problem you will implement a calculator with three simple operations: add (**code: a**), subtract (**code: s**), and multiplication (**code: m**). The inputs can be floating point numbers. You can not use any built in Floating point arithmetic operation, i.e., use built in integer arithmetic operators to implement the operations. The results and inputs can be both positive and negative. You can assume that the inputs and the end result can be contained in a total of 9 digits (digits before the decimal point + digits after the decimal point. E.g., 3.457 has 4 digits, 3123.56789 has 9)

```
Welcome to Calculator v 1.0.
Press a to add, s to subtract and m to multiply.
Operation: a
Enter first number: 22.45
Enter second number: 3.5
The summation is: 25.95
Operation: s
Enter first number: 22.45
Enter second number: 3.3
The difference is: 19.15
Operation: m
Enter first number: 3.6
Enter second number: 2
Operation: 7.2
```