# Implementation and Run-time Analysis of Sorting and Searching Algorithms

**Lab Objective:**

- To implement different sorting algorithms such as Bubble sort, Insertion sort and Selection Sort and searching algorithms such as Sequential Search and Binary Search.
- To compare and analyze their performance by varying size of datasets.

**Lab Outcome:**

- Students are expected to be able to implement the aforesaid sorting and searching algorithms and are also expected to solve real-life problems related to these.
- Furthermore, students are expected to learn how to generate random large dataset in C/ C++. and to calculate execution time of a function/program.

**Discussion:**

| ***BUBBLESORT (A,n)*** | ***INSERTIONSORT(A,n)*** | ***SELECTIONSORT(A,n)*** |
|---|---|---|
| 1. for i = 1 to n-1<br>2. for j = n downto i+1<br>3. if A[j] < A[j-1]<br>4. swap A[j] with A[j-1] | 1. for i = 2 to n<br>2. key = A[i]<br>3. j = i - 1<br>4. while j > 0 and A[j] > key<br>5. A[j+1] = A[j]<br>6. j = j - 1<br>7. A[j+1] = key | 1. for i = 1 to n-1<br>2. k = i<br>3. for j = i+1 to n<br>4. if A[j] < A[k]<br>5. k = j<br>6. swap A[i] with A[k] |

| ***SEQUENTIALSEARCH (A,n,value)*** | ***BINARYSEARCH(A,n,value)*** |
|---|---|
| 1. for i = 1 to n<br>2. if A[i] = = value<br>3. return i<br>4. return -1 // -1 = not found | 1. low = 1, high = n<br>2. while low<=high<br>3. mid = (low+high)/2<br>4. if A[mid] > value<br>5. high = mid - 1<br>6. else if A[mid] < value<br>7. low = mid + 1<br>8. else<br>9. return mid<br>10. return -1 // -1 = not found |

**Generating Random Dataset**

- Random dataset can be generated using srand() and rand() function in C/C++.
- Example code in C/C++

```
// seeding the random sequence so that each time the generated
sequence is different
srand (time(NULL));
// generating 10 numbers in random
for(int i=0; i<10; i++)
     myArray[i] = rand()%100;
     // numbers will be in the range from 0 to 99
```

**Calculating Execution/Run time of a function/program**

- ❖ Current system timestamp is returned by clock() function.
- ❖ Example code to calculate execution time of a function FUNC()

```
long start = clock();
FUNC();
long end = clock();
long difference = end-start;
// difference contains number of clock ticks or usually duration
in microseconds
double seconds = (difference*1.0)/CLOCKS_PER_SEC;
// seconds holds the duration in seconds
```

**Lab Assignment (LAB 01):**

- ✓ **Implement all three sorting and searching algorithms in a single program using appropriate functions and the same dataset (a small one) and Submit your program electronically.**

  Your program must be named as 245_LAB01_<your-student-id>.cpp and upload the file into https://dropitto.me/CSE245 by 21 September 11:59:59 PM.

- ✓ **Prepare a report (hardcopy) comparing their performance based on execution time for datasets of different sizes and Submit it on the next lab.**

I am expecting that you will write a report based on the following sample table. (This is the result of the experiment in my machine. Your result must be different than me and your closest friends !!!)

| Size of Dataset | Bubble Sort | Insertion Sort | Selection Sort |
|---|---|---|---|
| 1000 | 0.009177 | 0.001618 | 0.002509 |
| 5000 | 0.226590 | 0.039365 | 0.606692 |
| 10000 | 0.907910 | 0.157673 | 0.242460 |

**The larger the dataset you will use and the more the visual information you will provide The better marks you can expect -:)**