LAB - 2

Fundamentals of Algorithmic Problem Solving-I:

(Analysis of time complexity of small algorithms through step/frequency count method.)

PROGRAM EXERCISE

Lab. Exercise (LE)

2.1 Write a program to **test whether a number n, entered through keyboard is prime or not** by using different algorithms you know for atleast 10 inputs and note down the time complexity by step/frequency count method for each algorithm & for each input. Finally make a comparision of time complexities found for different inputs, plot an appropriate graph & decide which algothm is faster.

Sl. No.	Input (n)	Prime Number Testing		
		Algorithm-1 (Time by frequency)	Algorithm-2 (Time by frequency)	Algorithm-3 (Time by frequency)
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				

N.B: If you can solve more than three different ways, then add more columns right of Algorithm-3 column.

2.2 Write a program **to find out GCD (greatest common divisor)** using the following three algorithms.

a) Euclid’s algorithm
b) Consecutive integer checking algorithm.
c) Middle school procedure which makes use of common prime factors. For finding list of primes implement sieve of Eratosthenes algorithm.

Write a program to find out which algorithm is faster for the following data. Estimate how many times it will be faster than the other two by step/frequency count method in each case.

i. Find **GCD of two numbers when both are very larg**e i.e.GCD(31415, 14142) by applying each of the above algorithms.

ii. Find **GCD of two numbers when one of them can be very large** i.e.GCD(56, 32566) or GCD(34218, 56) by applying each of the above algorithms.

iii. Find **GCD of two numbers when both are very smal**l i.e.GCD(12,15) by applying each of the above algorithms.

iv. Find **GCD of two numbers when both are same** i.e.GCD(31415, 31415) or GCD(12, 12) by applying each of the above algorithms.

Write the above data in the following format and decide which algorithm is faster for the particular data.

Sl. No.	**Input GCD(x, y)**	**GCD Algorithm**			
		Euclid's algorithm (Frequency Count)	**Consecutive integer checking algorithm. (Frequency Count)**	**Middle school procedure algorithm (Frequency Count)**	**Remarks (Which one Faster than other two)**
1	GCD (31415, 14142)				
2	GCD (56, 32566)				
3	GCD (34218, 56)				
4	GCD(12,15)				
5	GCD (31415, 31415)				
6	GCD (12, 12)				

Home Exercise (LE)

2.3 Write a menu driven program as given below, to sort an array of n integers in ascending order by **insertion sort algorithm** and determine the **time required (in terms of step/frequency count)** to sort the elements. Repeat the experiment for different values of n and different nature of data (i.e.apply insertion sort algorithm on the data of array that are already sorted, reversly sorted and random data). Finally plot a graph of the time taken versus n for each type of data. The elements can be read from a file or can be generated using the random number generator.

INSERTION SORT MENU

- **0.** Quit
- **1.** n Random numbers=>Array
- **2.** Display the Array
- **3.** Sort the Array in Ascending Order by using Insertion Sort Algorithm
- **4.** Sort the Array in Descending Order by using any sorting algorithm
- **5.** Time Complexity to sort ascending of random data
- **6.** Time Complexity to sort ascending of data already sorted in ascending order
- **7.** Time Complexity to sort ascending of data already sorted in descending order
- **8.** Time Complexity to sort ascending of data for all Cases (Data Ascending, Data in Descending & Random Data) in Tabular form for values n=5000 to 50000, step=5000

Enter your choice:

If the choice is option 8, the it will display the tabular form as follows:

Analysis of Max-Heap Sort Algorithm

Sl. No.	Value of n	Time Complexity (Data in Ascending)	Time Complexity (Data in Descending)	Time Complexity (Random Data)
1	5000			
2	10000			
3	15000			
4	20000			
5	25000			
6	30000			
7	35000			
8	40000			
9	45000			
10	50000			