CET 214 – Data Structures & Algorithms

Experiment # 2

Experiment Title

Algorithm Development and Implementation

Assessment of CLO(s): III

Performed on 27-09-2024

Student Name			
Roll No.		**Group**	
Semester		**Session**	

Total (Max)	**Criteria 1 (2.5)**	**Criteria 2 (2.5)**	**Criteria 3 (2.5)**	**Criteria 4 (2.5)**	**Total (10)**
Marks Obtained					
Remarks (if any)					

Experiment evaluated by

Instructor's Name	**Engr. Muhammad Asad Husain**		
Date		**Signature**	

Department of Engineering Technology
(UIT University)

Course Code: CET214 Course Title: Data Structures & Algorithms Course Credits: 2+1 Session: Fall 2024

Rubric for assessment criteria to perform experiment number 2.

Level Criteria	UNSATISFACTORY 1	COMPETENT 2	PROFICIENT 3	DISTINGUISHED 4
Capability of writing algorithm/ Procedure	None of the steps are implemented of an algorithm.	Few steps are implemented correctly of an algorithm.	Most of the steps are implemented correctly of an algorithm.	All the steps are implemented correctly of an algorithm.
Capability of writing Program	Programs not completed.	Completeness of code, consistent variable naming and unformatted.	Completeness of code, inconsistent variable naming and well formatted.	Completeness of code, consistent variable naming and well formatted.
Completion of target in Lab	25% target has been completed	50% target has been completed	75% target has been completed	100% target has been completed
Output	None of the outputs are correct.	Few outputs have been found correctly.	Some of the outputs are correct and well formatted.	Most of the outputs are correct and well formatted.

Practical Objective(s):

1. Practicing the process of writing an algorithm to solve a problem
2. Practicing the process of converting an algorithm into program
3. Being able to write an algorithm and the corresponding program on your own

Theory

In this experiment, we will practice how to write an algorithm for a particular problem and convert it into a program.

Example(s):

Example 1:

Algorithm 1(a): A non-empty array DATA with N numerical values is given. These algorithms find the location LOC and the value MAX of the largest element of DATA. The variable K is used as counter.

```
Step 1:  [Initialize] set K: =1, LOC: =1 and MAX:

=DATA [1]

Step 2:  [Increment counter] Set K: =K+1

Step 3:  [Test Counter] if K >N, then: Write: LOC, MAX, and Exit.

Step 4:  [Compare and update] if MAX < DATA [K], then: Set LOC: = K
         and MAX: =DATA [K]

Step 5:  [Repeat loop] Go to Step 2
```

Code:

```
#include <stdio.h>
#include <iostream>
using namespace std;

int main()
{
   int data[]={10,3,16,1,27,98,5,112,65,8,13,123,190};
   int location;
   int k=1;
   int max=data[0];

   label: k=k+1;
```

```
        if  (k<13)
         {
                if (max<data[k])

                     { location=k; max=data[k]; }
                goto label;
            }
    cout<<"location="<<location<<"\n";
    cout<<"Largest element="<<max<<"\n";
    system ("pause");
    return 0;
 }
```

Algorithm 1(b):

```
Step 1:  [Initialize] Set K: =1, LOC: =1 and MAX: =DATA [1]

Step 2:  Repeat Step 3 and 4 while K less than and equal to N

Step 3:  If MAX<DATA [K], then
             Set LOC: =K and MAX: =DATA [K]

Step 4:  Set K: =K+1

Step 5:  Write LOC, MAX

Step 6:  Exit
```

Example 2:

Algorithm 2: This algorithm inputs the coefficients A, B, C of a quadratic equation and outputs the real solutions, if any.

```
Step 1:     Read: A, B, C

Step 2:     Set $D = b^2 - 4ac$

Step 3:     If D > 0, then:
                (a) Set $X_1 = \frac{(-b+\sqrt{D})}{2a}$ and $X_2 = \frac{(-b-\sqrt{D})}{2a}$
                (b) Write: $X_1, X_2$
            Else if D=0, then
                (a) Set $X = \frac{-b}{2a}$
                (b) Write: ‘UNIQUE SOLUTION’, X
            Else:
                Write: ‘NO REAL SOLUTIONS’
            [End of if structure]

Step 4:     Exit
```

Code:

```
#include <stdio.h>
#include <iostream>
#include <math.h>
using namespace std;


int main()
{
         int A,B,C,D;
         float X,X1,X2;

         cout<<"Enter the value of A: ";
         cin>>a;
         cout<<"Enter the value of B: ";
         cin>>b;
         cout<<"Enter the value of C: ";
         cin>>c;
         statement

         if (condition)
          {
               X1=statement; X2=statement;
               cout<<"x1="<<X1<<"\n";
               cout<<"x2="<<X2<<"\n";
           }

         else if (condition)
         {
               statement;
               cout<<"UNIQUE SOLUTION"<<X;
         }
         else
               statement;

          system ("pause");
          return 0;
 }
```

Do It Yourself:

1. Implement algorithm 1(b) in C++
2. Complete the code for algorithm 2
3. Write an algorithm to divide a given array of integers into two sub- arrays. Sub-array1 should consist of the even numbers existing in the array and sub-array2 should consist of the odd numbers existing in the array.
4. Implement the above algorithm in C++.

Question(s):

1. Which of the two algorithms present a better programming practice?
   Algorithm 1(a) or Algorithm 1(b)?
2. Write the reason for your answer in question 1.