Assignment I [Phase I]

Course Code: CSE214

Course Name: Algorithm Lab

Submitted To

Subroto Nag Pinku
Department of CSE
Daffodil International University

Submitted By

Name: S.M. Khasrul Alam Shakil
Id: 191-15-12180
Sec: 0-14

1.Write short notes on Optimization.

Answer: In real life we can do a work in many ways but Optimization refers that process which help to find an alternative way to which is cost effective or highest achievable performance under the given constraints, by maximizing desires factors and minimizing undesired ones, In comparison, maximization means trying to attain the highest or maximum result or outcome without regard to cost or expense. Practice of optimization is restricted by the lack of full information, and the lack of time to evaluate what information is available.
In computer science different function take different time and space for a specific set of output, by doing optimization to the function we can minimize running time or space or both of this function, in a word by optimizing we can make it more efficient.

2.What are the different algorithms you know?

Answer: I know some type of algorithm they are recursive algorithm , backtracking algorithm, devide and conquer algorithm, some are greedy , brute force algorithm and randomized algorithm.
Some of these algorithm name is:

- Searching: Linear Search, Binary Search.
- Sorting: Bubble Sort, Insertion Sort, Selection Sort, Merge sort, Quick Sort.
- Greedy: Bin packing, Knapsack, coin-change, Huffman Coding.

3. Why are you learning so many algorithms?

Answer: We learn algorithm to do specific work in an efficient and optimized way for our need. Some time we need to run a program in less time or sometime we want the algorithm to consume less space. All of the method of doing these are descried in different algorithm.

An *algorithm* is any well-defined computational procedure that takes some value, or set of values, as *input* and produces some value, or set of values, as *output*. An algorithm is thus a sequence of computational steps that transform the input into the output. Different algorithms devised to solve the same problem often differ dramatically in their efficiency. These differences can be much more significant than differences due to hardware and software. Computers were infinitely fast and computer memory was free

Would we have any reason to study algorithms? The answer is yes, if for no other reason than that we would still like to demonstrate that your solution method terminates and does so with the correct answer. If computers were infinitely fast, any correct method for solving a problem would do. We would probably want our implementation to be within the bounds of good software engineering practice (for example, your implementation should be well designed and documented), but we would most often use whichever method was the easiest to implement. Of course, computers may be fast, but they are not infinitely fast. And memory may be inexpensive, but it is not free. Computing time is therefore a bounded resource, and so is space in memory. We should use these resources wisely, and algorithms that are efficient in terms of time or space will help you do so.

4) Show analysis of a recursive algorithm.

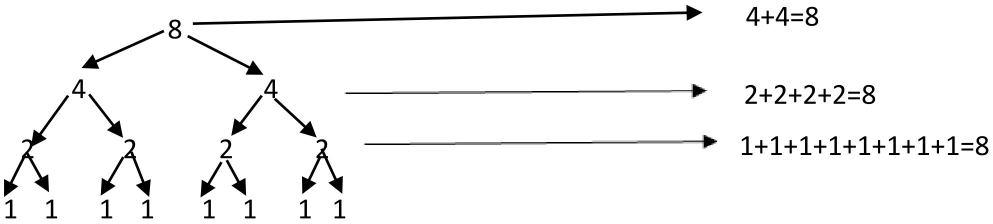
Recursive Algorithm Analysis (Merge sort):

For analyzing the merge sort algorithm we assume that, the function has two part, first we divide the array into 2 sub array and in second part we merge two sub-array into 1 array.

```
void mergeSort(int low,int high)
{
   if(high==low)
      return;
   int mid=(high+low)/2;                          First Part
   mergeSort(low,mid);
   mergeSort(mid+1,high);

   for(int i=low,j=mid+1,k=low; k<=high; k++)
   {
      if(i==mid+1)
         tmp[k]=ara[j++];
      else if(j==high+1)
         tmp[k]=ara[i++];
      else if(ara[i]<=ara[j])                             Second Part
         tmp[k]=ara[i++];
      else
         tmp[k]=ara[j++];
   }
   for(int i=low; i<=high; i++)
      ara[i]=tmp[i];
}
```

In first part we divide the array in two sub part and pass them recursively, in the second part we merge the two part. We can easily show it by the help of a tree. Suppose array size is 8



From the try we see that for merging two sub-array we are doing n iteration where n is the size of the array. And the total number of iteration is happening is L*n , Here L is the number of level in the tree.

When,

n=2 level=1

n=4 level=2

n=8 level=3

n=16 level=4

Where the number of level is showing a logarithmic relation with the size of the array and it is

L=log(n) (base 2)

So, from the above observation total number of execution

L*n

=nlog(n)

Normally best case happens when the array is in sorted order and in worst case the array will be in reversely sorted order, in both case we have the same number of Level in the analyzing tree.

So time Complexity,

11	7	8	3	1

Worst Case: O (nlog(n))

1	3	7	8	11

Best Case: O (nlog(n))

5. Design an iterative and recursive algorithm and prove that your algorithm works

Answer:

Suppose I need to find summation of natural number 1 to N. So I am making an iterative and recursive algorithm for it.
Suppose N=10 ,so summation of first 10 number 1-10 is 1+2+3+4+5+6+7+8+9+10 =55

Designing an algorithm for it (Iterative form):

```
Summation(N)
    sum=0
    for <-1 to N
        sum+=N
    return sum;
```

(recursive form):

```
Summation(N)
    if(n==0)
        return
    sum+=N
    Summation(n-1)
```

If we put n=10 thn for both of the algorithm we will get result 55 that we calculated above. So both of the algorithm work successfully