18B17CI472-ALGORITHMS LAB (ALGO Lab)
Lab 2

NOTE:

- This lab is to be completed individually. Do not share your work or code with anyone else.
- You can use any programming language that you like; we suggest Python, C++, or C.
- For all of our labs, please avoid using Google to find suggestions or solutions. The goal is to use your own brain to work these problems out, which will help you develop the skills to do well in the exams and, more importantly, become a substantially better computer programmer.
- You have to make lab report in hard copy.

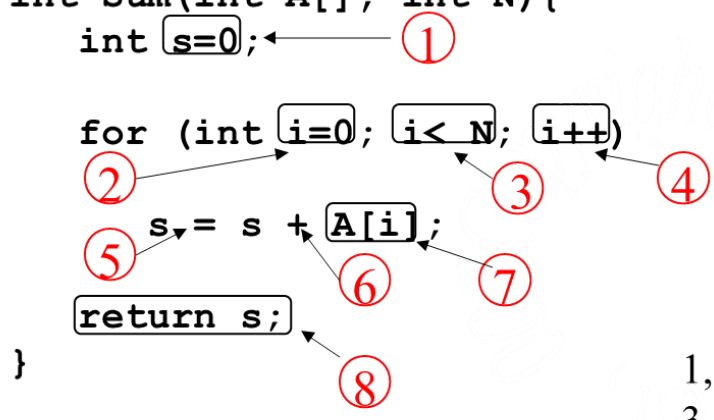
---

1. Growth of Function:

As discussed in class, write a program to calculate the total number of steps required for *N* =10 and N=100 using the algorithm provided below.

```
// Input: int A[N], array of N integers
// Output: Sum of all numbers in array A

int Sum(int A[], int N){
    int s=0;
    for (int i=0; i< N; i++)
        s= s + A[i];
    return s;
}
```



1,2,8: Once
3,4,5,6,7: Once per each iteration of for loop, N iteration
Total: 5N + 3
The *complexity function* of the algorithm is : $f(N) = 5N + 3$

2. Algorithm Bubble sort

Write a program for the algorithm that sorts the element of an array 'A' (having *n* elements) in the ascending (increasing) order. The pass counter is denoted by variable 'P' and the variable '*E*' is used to count the number of exchanges performed on any pass. The last unsorted element is referred by variable '*l*'.

Step 1 **Initialization**
set $l = n$, $P = 1$

Step 2 **loop,**
Repeat step 3, 4 while ($P = n - 1$)
set $E = 0$, **R** Initializing exchange variable.

Step 3 **comparison, loop.**
Repeat for $i = 1, 1, \ldots\ldots l - 1$.
if ($A[i] > A[i + 1]$) then
set $A[i] = A[i + 1]$, **R Exchanging values.**
set $E = E + 1$

Step 4 **Finish, or reduce the size.**
if ($E = 0$) then
exit
else
set $l = l - 1$.

(A) The following Procedure Sub algorithm swaps the values of the parameters, all values return implicitly.

Solution:

Function Swapping (*x*, *y*)
The above procedure sub algorithm swaps the values of variable '*x*' and '*y*'. All variables are assumed to be real.

Step 1 **R** swapping, values of parameters
Set $x == y$

Step 2 **R** returning, at point of call
return.

(B) In this Function Subalgorithm 'Add' can be translated into an equivalent Procedure Subalgorithm.

Solution:

Procedure Add (*x*, *y*, *z*, *w*)
The above Procedure Subalgorithm adds the value to the variable '*x*', '*y*' and '*z*'. Variable '*w*' holds the computed result of the addition. All variables are assumed to be of type real.

Step 1 **R** **computing, sum**
set $w = x + y + z$

Step 2 **R** **returning, at point of call**
Return