1 .(a) How can we measure the performance of an algorithm

illustrate the concept of time space tradeoff with the help of an example.

Ans .**There are many criteria upon which can judje an algorithm**

1 ) Does it do what we want it to do?

2) Does it work correctly according to the original specification of the task?

3) Is there documentation that describes how to use it and how it works?

4) Are producers created in such a way that they perform logical subfunctions?

5) Is the code readable?

There are other **criteria for judging** algorithms that have a more direct relationship to performance. These have to do with their computing time and storage requirements.

**Space/Time complexity**:--The space complexity of an algorithm is the amount of memory it needs to run to completion .The time complexity of an algorithm is the amount of computer time it needs run to completion

Example of the space complexity

Algorithm sum(x,y)

{

Return x+y;

}

Two computers words will be required to store variable x and y so,

Space complexity (s)=2

Example of the time complexity

The time complixity of an algorithm is given the number of steps taken by the algorithm

Algorithm sum(a[],5)

{

Sum=0;

For(i=0;i<=5;i++)

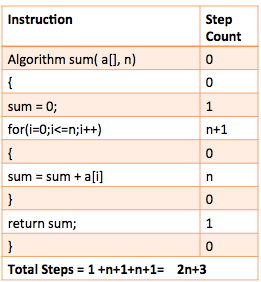
{

Sum=sum+a[i];

}

Return sum;

}



(b) Calculate space and time complexity for the following . Represent them using asymptotic notations .

Ans: **ALGORITHM**  **STEPS COUNT**

1. for i:1 to length of A n+1

2. If A[i] is equal to x n

3. Return TRUE 0

4. return FALSE 1

The running time of an algorithm on a particular input is thw number of **steps** executed.

Tatal steps= (n+1)+n+0+1

In asymptotic notation: O(n)

(c) Introduce statements to increment count at all appropriate points in the following algorithm

Algorithm Transpose(a,n)

{

for i=1 to n-1 do

count= count+1

for j=i+1 to n do

{

Count=count+1

{

t=a[i,j]; count=count+1

a[i,j]=a[j,i]; count=count+1

a[j,i]=t; count=count+1

}

}

Count=count+1

}

ALGORITHM transpose(a,n)

{

For i=1 to n-1 do

{

Count=count+2;

For j=i+1 to n do

Count = count+2;

}

Count = count+3;

}