Examples

Assume that in the code pieces below, each function call has the time complexity shown next to it.

Give the Worst, Best, and possibly Average time complexity. Also give the general time complexity (what you would report).

* Example 1

if (valid(val)){ // O(1)

 foo(arr, N, val) ; // O(N)

}

else {

 printf("Error"); // O(1)

}

Worst: O(\_\_\_\_) , Best: O(\_\_\_\_), Average: O(\_\_\_\_), General TC: O(\_\_\_\_)

* Example 2 (no false branch)

if (valid(val)){ // O(1)

 foo(arr, N, val) ; // O(N)

}

How many paths are possible when this code executes?

Worst: O(\_\_\_\_) , Best: O(\_\_\_\_), Average: O(\_\_\_\_), General TC: O(\_\_\_\_)

* Example 3

if (validBig(val,N)){ // O(N)

 foo(arr, N, val) ; // O(N)

}

else {

 printf("Error"); // O(1)

}

Worst: O(\_\_\_\_) , Best: O(\_\_\_\_), Average: O(\_\_\_\_), General TC: O(\_\_\_\_)

* Example 4

if (validBig(N,val)){ // O(N)

 fooSmall(N, val) ; // O(1)

}

else {

 bar(N); // O(N2)

}

Worst: O(\_\_\_\_) , Best: O(\_\_\_\_), Average: O(\_\_\_\_), General TC: O(\_\_\_\_)

* Example 5

Assume array A is sorted in increasing order and has N elements.

k=N-1;

while ((k>=0)&&(A[k]>val)){

 k--;

}

Worst: O(\_\_\_\_) , Best: O(\_\_\_\_), Average: O(\_\_\_\_), General TC: O(\_\_\_\_)

General guidelines

1. When giving the general TC, we give the WORST case.
2. If possible, analyze and give best and average cases as well.
3. When analyzing average case, assume uniform distribution (each event/case is equally likely to happen).