Finding Time Complexity for Recurrences:

Identify the number of times a recursive call\_\_\_\_\_\_\_\_\_\_\_\_\_and what the new \_\_\_\_\_\_\_\_\_\_\_\_\_ is

Local time complexity is\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Note:

* c is generally used as a \_\_\_\_\_\_\_\_\_\_\_\_\_
* c \_\_\_\_\_\_ Θ(1)
* n  \_\_\_\_\_\_   cn  \_\_\_\_\_\_  Θ(n)

int foo(int N){

 int a,b,c;

 if(N<=3) return 1500; // Note N<=3

 a = 2\*foo(N-1);

 // a = foo(N-1)+foo(N-1);

 printf("A");

 b = foo(N/2);

 c = foo(N-1);

 return a+b+c;

}

Base case: T( \_\_ ) = \_\_\_\_\_\_\_\_\_\_

Recursive case: T( \_\_ ) = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

T(N) gives us the Time Complexity for foo(N). We need to solve it (find the closed form)

void bar(int N){

 int i,k,t;

 if(N<=1) return;

 bar(N/5);

 for(i=1;i<=5;i++){

 bar(N/5);

 }

 for(i=1;i<=N;i++){

 for(k=N;k>=1;k--)

 for(t=2;t<2\*N;t=t+2)

 printf("B");

 }

 bar(N/5);

}

Base case: T( \_\_ ) = \_\_\_\_\_\_\_\_\_\_

Recursive case: T( \_\_ ) = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Solve T(N)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Level | Arg/Problem Size | Nodes per Level | 1 Node TC | Level TC |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Level | Arg/Problem Size | Nodes per Level | 1 Node TC | Level TC |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Level | Arg/Problem Size | Nodes per Level | 1 Node TC | Level TC |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |