mystery

Time analysis of functions

1. add = O(1)
2. dothething = O(n) We make two recursive function calls because nth Fibonacci number is equal to (n-2)th Fibonacci number + (n-1)th Fibonacci number = O(2n) = O(n).

Optimized version – use an array num. Store the nth Fibonacci number in the nth subscript of the array. This way we avoid redundant recursive calls by accessing the data in the array to get pre-calculated Fibonacci numbers. This improves time efficiency. In my program I implemented it using the using the commands:

```
if(num[n-2]!=0 && num[n-1]!=0){
fib = add(num[n-2], num[n-1]);
}
```

This is why the assembly code for the optimized version has .comm num (a global variable, array of ints, called num)