CS240: Recurrences

Follow the instructions for each problem. Assume **all functions are valid for $n \geq 0$.**

1. Write a recurrence relation that describes the number of **underlined operations** performed:

```
public static int fun(int n) {
    if (n == 0) {
        return 20;
    } else {
        int result = 10 - n;
        return result + fun(n - 1);
    }
}
```

Initial condition(s):	**Recurrence relation:**
T(0) =	T(n) =

2. Write a recurrence relation that describes the number of **assignments to sum**:

```
public static int fun(int n) {

    int sum = 0;

    if (n == 0) {
        return 8;
    }

    for (int i = 0; i < n; i++) {
        int result = fun(n - 1) + fun(n - 1);
        sum += result;
    }
    return sum;
}
```

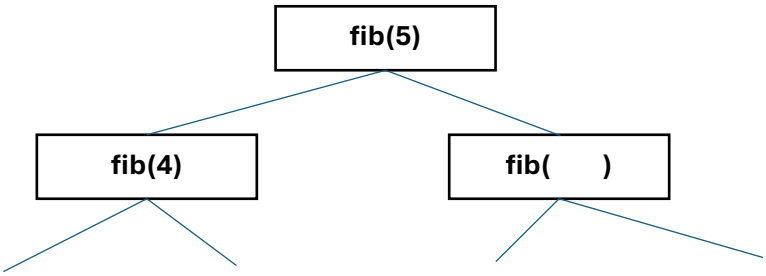
Initial condition(s):	**Recurrence relation:**
T( ) =	

For these problems, consider the **fib** function below:

```
public static int fib(int n)
{
    if (n <= 1) {
        return n;
    }

    return fib(n - 1) + fib(n - 2);
}
```

3. First, *complete* the "tree diagram" to **trace** all `fib` calls, with an initial call of **`fib(5)`:**



How many total calls are there? _____

How "tall" or deep does the tree get? _____ *(include the initial call/root in your count)*

4. Write a recurrence relation that describes the number of **comparisons**:

   (Hint: You will need two initial conditions... why?)

Initial condition(s):	**Recurrence relation:**
T( ) =	
T( ) =	