

## CECS 328: Program #1

For each of the tasks, I have included the algorithm in C# as well as a screenshot of the output.  
The source code for the project is available [here](#).

1. Write a program to calculate  $S(n)$  by calculating the values of the Fibonacci sequence recursively.

### Source:

```
public static long Sn(long number)
{
    long result = 0;

    for (int i = 1; i <= number; i++)
    {
        result += Fibonacci(i);
    }

    return result;
}

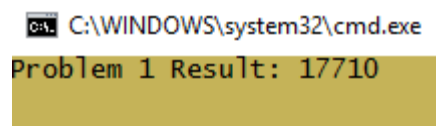
public static long Fibonacci(long number)
{
    if(number <= 2)
    {
        if(number == 0)
        {
            return 0;
        }
        return 1;
    }

    return Fibonacci(number - 1) + Fibonacci(number - 2);
}
```

### Sample run:

```
#region Problem 1: Calculate S(n) by calculating the values of the Fibonacci sequence
recursively
// S(n) definition: S(n) = f(0) + f(1) +...+ f(n)

int sampleN = 20;
Console.WriteLine($"Problem 1 Result: {Sn(sampleN)}\n\n");
#endregion
```



C:\WINDOWS\system32\cmd.exe  
Problem 1 Result: 17710

2. Write a non-recursive program to calculate  $S(n)$ .

**Source:**

```
public static long NonRecursiveSn(long number)
{
    long previous = 1;
    long previous2 = 1;
    long current = 0;
    long sum = 2;

    for (int i = 3; i <= number; i++)
    {
        current = previous + previous2;
        sum += current;
        previous = previous2;
        previous2 = current;
    }

    return sum;
}
```

**Sample run:**

```
#region Problem 2: Write a non-recursive program to calculate S(n)
Console.WriteLine("Problem 2 Result: \n");
Console.WriteLine($"Fibonacci Table for n = {sampleN}:");

for (int i = 0; i <= sampleN; i++)
{
    Console.WriteLine($"f({i}) = {Fibonacci(i)}");
}
Console.WriteLine("-----");
Console.WriteLine($"Sum: {NonRecursiveSn(sampleN)}\n\n");
#endregion
```

C:\WINDOWS\system32\cmd.exe

```
Problem 2 Result:
Fibonacci Table for n = 20:
f(0) = 0
f(1) = 1
f(2) = 1
f(3) = 2
f(4) = 3
f(5) = 5
f(6) = 8
f(7) = 13
f(8) = 21
f(9) = 34
f(10) = 55
f(11) = 89
f(12) = 144
f(13) = 233
f(14) = 377
f(15) = 610
f(16) = 987
f(17) = 1597
f(18) = 2584
f(19) = 4181
f(20) = 6765
-----
Sum: 17710
```

3. Algebraically verify that  $g(n)$  is a solution of Equ 1 by substituting  $g(n)$  in Equ 1.

(This portion is attached to the assignment packet)

4. Write a third iterative program by summing  $g(n)$  over  $n$ .

### Source:

```
public static long GrimaldiSum(long number)
{
    long result = 0;

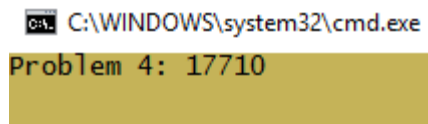
    for (int i = 1; i <= number; i++)
    {
        result += Grimaldi(i);
    }

    return result;
}

public static long Grimaldi(long k)
{
    return (long)((1 / Math.Sqrt(5)) * (Math.Pow(((1 + Math.Sqrt(5)) / 2), k) - Math.Pow(((1 - Math.Sqrt(5)) / 2), k)));
}
```

### Sample run:

```
#region Problem 4: Third possible way to sum Fibonacci
Console.WriteLine($"Problem 4: {GrimaldiSum(20)}\n\n");
#endregion
```

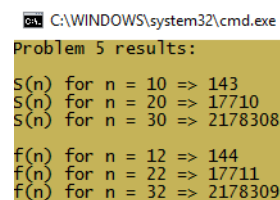


C:\WINDOWS\system32\cmd.exe  
Problem 4: 17710

5. Calculate these values of  $S$  for  $n = 10, 20, 30$ . Calculates values of  $f$  for  $n = 12, 22, 32$ .

### Sample run:

```
#region Problem 5: Calculate S for 10, 20, 30; Calculate f for 12, 22, 32
Console.WriteLine($"Problem 5 results: \n");
Console.WriteLine($"S(n) for n = 10 => {GrimaldiSum(10)}");
Console.WriteLine($"S(n) for n = 20 => {GrimaldiSum(20)}");
Console.WriteLine($"S(n) for n = 30 => {GrimaldiSum(30)}\n");
Console.WriteLine($"f(n) for n = 12 => {Fibonacci(12)}");
Console.WriteLine($"f(n) for n = 22 => {Fibonacci(22)}");
Console.WriteLine($"f(n) for n = 32 => {Fibonacci(32)}\n");
#endregion
```



C:\WINDOWS\system32\cmd.exe  
Problem 5 results:  
S(n) for n = 10 => 143  
S(n) for n = 20 => 17710  
S(n) for n = 30 => 2178308  
  
f(n) for n = 12 => 144  
f(n) for n = 22 => 17711  
f(n) for n = 32 => 2178309

6. Prove the identity  $S(n) = f(n + 2) - 1$  using induction.

(This portion is attached to the assignment packet)

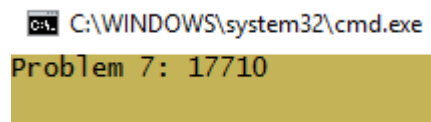
7. Example run using the alternate sum from #6.

### Source:

```
public static long SnAlt2(long number)
{
    return Fibonacci(number + 2) - 1;
}
```

### Sample run:

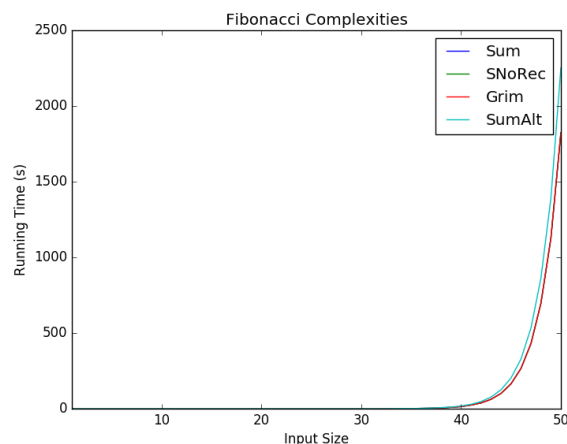
```
#region Problem 7: Fourth way to calculate S(n)
Console.WriteLine($"Problem 7: {SnAlt2(20)}\n\n");
#endregion
```

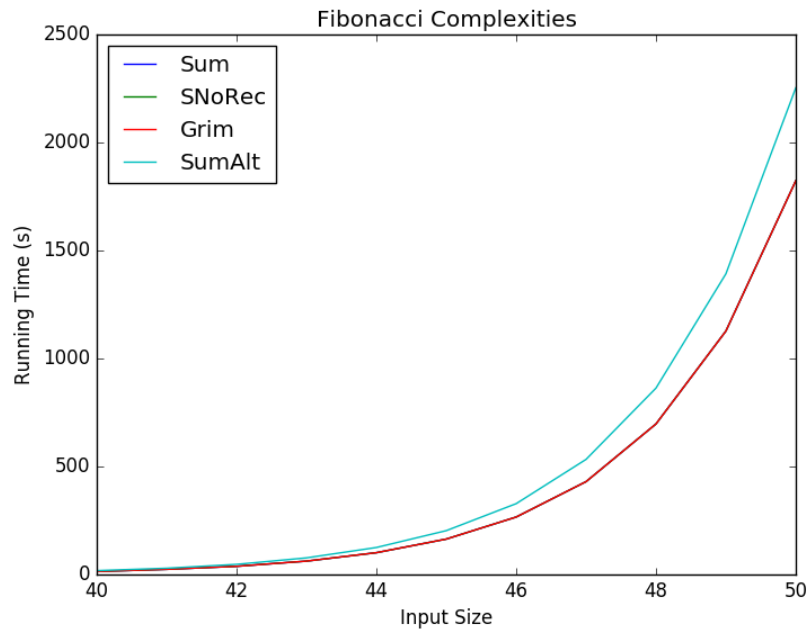


8. Since I am using Int64 (long in C#) to calculate the sums, the largest  $n$  that can be used is such that  $S(n)$  does not exceed 9,223,372,036,854,775,807. Another limitation would be the possibility of running out of stack space. Since the Fibonacci calls are recursive, a sufficiently large  $n$  would cause the program to crash.

9. Compare the running times of the 4 methods.

I computed the sum with each method for values of  $n$  starting at 1 and ending at 50. I recorded the running times for each. Finally, I ran a python script to plot the growth of the running times versus the input size.





#### 10. Summary and conclusions.

We see in the plots above that the 4<sup>th</sup> method grows the fastest by a considerable amount. However, the difference is not much. The drawbacks of the recursive methods are that they have linear growth in space complexity:  $O(n)$ , as opposed to the non-recursive methods that have a constant space complexity.