Part 2 Tasks:

1.

First	2
Second	3
Third	5
Fourth	8
Fifth	13
Sixth	21
Seventh	34
Eighth	55
Ninth	89
Tenth	144
Eleventh	233
Twelfth	377
Thirteenth	610
Fourteenth	987
…	…
Twentyfourth	121.393

2. As we can see, the numbers get quite large as we reach the 24th month, after this, it would become very tedious to manually calculate the subsequent number. It would probably be smart to look for the pattern and derive an equation for all n such that $n+1 = n+(n-1) \Rightarrow n = (n-1)+(n-2)$ *for nth Fibonacci number* The problem is how can we determine what the value of n is in the first place. We solve this using recursion, it would be quite difficult to come up with an equation for any n. *Creating a computer program would be quite simple.*

3. In order to get the function itself, we'd have to analyze the values deeply and the final answer would probably be some kind of exponential since the values get exponentially bigger. Without coming up with this equation there would be no way to calculate the *nth Fibonacci number* without having any givens.

4. After making the program used to calculate Fibonacci numbers, the function works in linear time, given a number n, it will make n recursive calls to calculate the number. In terms of space, I used an array to store the past values so it will use one index per subsequent value. This could

also be implemented extensively using a queue, stack, or tree to possibly eliminate the need to store all previous values like in an array.

Fibonacci binary tree:
https://inst.eecs.berkeley.edu/~cs61bl/r//cur/trees/fibonacci-tree.html?topic=lab15.topic&step=7&course=