CS 2640

Project 2

Juan Vera

Problem

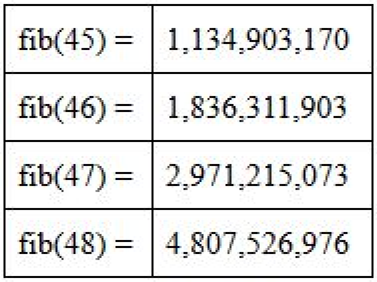
The initial problem is to create a fibonacci number calculator where the user inputs an integer n and gets returned the nth fibonacci number. Afterwards, we must print the first n numbers in the fibonacci sequence. The main problems are input validation and figuring out what the maximum input value can be considering out byte-sized constraints.

First Glance

At first glance, the pseudocode for the fibonacci sequence shows us we need to loop from 0 to n-1, while keeping track of variable a, b, and a temp variable. To do this, we can allocate space for an array, and have 5 registers for: array, array size, a, b, and temp. Once we've done this, we can iterate through a loop, calculating the i^{it} number, storing it at array[i], and incrementing the array pointer by 4 each time until the pointer is greater than the array + array size pointer. However, the array size pointer depends on user input. In order to do this, we must ask the user to input an integer, then store the integer as the array size register. Since fibonacci numbers can't be negative, we know we will always be using unsigned integers. From here, comes a big problem. If a user inputs a value bigger than the max value possible for 32 bits, then we get overflow.

Solution

To solve this, we must figure out the maximum fibonacci value we can reach without passing 2^{32}-1 (4,294,967,295). As we can see in the chart, the highest fibonacci number we can use is fib(47), since fib(48) > 2^{32}-1. Therefore, we can set the limit to be 47. To do this, we must have an input validation loop, where if we get user's n > 47, we



fib(45) =	1,134,903,170
fib(46) =	1,836,311,903
fib(47) =	2,971,215,073
fib(48) =	4,807,526,976

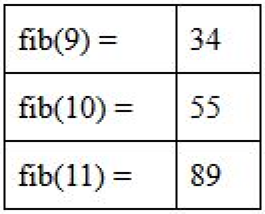
can branch to a function that tells the user that their input is too high, then jump to the start of the program. Similarly, we can create a second loop for if the user inputs an n<0, since we can't have negative numbers in a fibonacci sequence.

Finishing Up

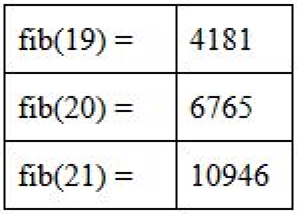
Now that we know we can only have a max of 48 values, we can statically allocate 48 words of space for the array, so that we will have enough space for any n number the user inputs. Finally, we can print the nth fibonacci number by looking at the last index of the array. Similarly, we can print the fibonacci sequence just as we stored the values in the array, but instead of storing, we'll be getting the values and printing them throughout each iteration of the output loop. For registers, we can reuse any except for the register that holds n, since we still need it to end the loop.

Results

When inputting 10 for n, my program returns 55. For 20, 6765. As we can see by the charts, these numbers are the ones we should be getting for their n values. When I try inputting a negative number or n > 47, I get the respective errors as many times as I try inputting them.



fib(9) =	34
fib(10) =	55
fib(11) =	89



fib(19) =	4181
fib(20) =	6765
fib(21) =	10946