CSCI 115 Lab

Week 3- Fibonacci Numbers

- Professor: Dr. Matin Pirouz
  Email: mpirouz@csufresno.edu

- TA: Shreeja Miyyar
  Email:shreejarao12@mail.fresnostate.edu

Table of Contents

Fibonacci Numbers

- Fibonacci sequences are series of numbers where the number is found by adding the previous two numbers.

- The Fibonacci sequence is as follows:

  0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89……

- The equation is given by:

  F(n)=F(n-1)+F(n-2) for n>1

  & ***F(0)=0*** and ***F(1)=1***

- For e.g.

  $F_0 = 0, F_1 = 1$

  $F_2 = F_1 + F_0 = 1$

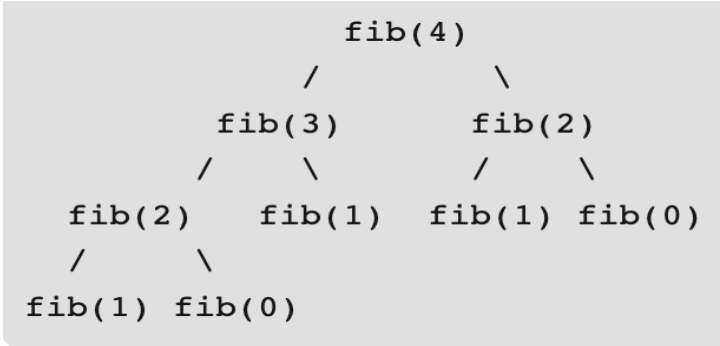
  $F_3 = F_2 + F_1 = 2$

Discuss the first natural solution !

Methods to find the Nth Fibonacci number

- Recursion

  In this method, the nth Fibonacci number is calculated by repeatedly calling the defined function until the nth number if reached.



  - Time to calculate fib(n) is the sum of time taken to calculate fib(n-1) and fib(n-2). It also considers the constant time taken for previous addition O(1).
  - Therefore the time complexity is O(2^n) which means exponential.
  - For Fibonacci recursive implementation or any recursive algorithm, the space required is proportional to the maximum depth of the recursion tree, because, that is the maximum number of elements that can be present in the implicit function call stack. Hence, space complexity is O(N).
  - This method does lot of repeated work and is inefficient.

Efficient Algorithm

Methods to find the N^{th} Fibonacci number (Cont'd)

- Iterative
  - In this method, the n^{th} Fibonacci number is calculated by iterating over the N numbers and calculating the Fibonacci number for each iteration.
  - This can be done by either storing the Fibonacci sequences in an array which would result in O(N) space complexity or storing the previous, current and next Fibonacci values in an integer variable which results in space optimized solution O(1).
  - The time complexity of the iterative code is linear, as the loop runs from 2 to n, i.e. it runs in O(n) time.
  - This is more efficient than the recursive algorithm.

Recursion algorithm

- **Step 1** - Read the input N to find the N^{th} Fibonacci number
- **Step 2** - Define a method fib which takes an integer argument.
- **Step 3** - Check if N is 0 or 1. If Yes, return the input back
- **Step 4-** If No, return fib(n-1)+fib(n-2)
- **Step 5-** Print the returned value as output.

Iterative algorithm - 1 (using arrays)

- **Step 1** - Read the input N to find the N^{th} Fibonacci number
- **Step 2** - Define a method fib which takes an integer argument.
- **Step 3** - Create an array of size N and set the element at position 0 and 1 of the array as 0 & 1 respectively.
- **Step 4** – Loop through the elements from 2 to N and calculate the Fibonacci number for each position using the formula.
- **Step 5** – Return the Fibonacci number in Nth position and display the result.

Iterative algorithm - 2 (using temporary variables)

- **Step 1** - Read the input N to find the N^{th} Fibonacci number
- **Step 2** - Define a method fib which takes an integer argument.
- **Step 3 -** Initialize *previous* and *current* variable to 0 and 1 respectively.
- **Step 4 -** Loop through the elements from 2 to N and calculate the value of *next* variable by adding *previous* and *current*. Set *previous* as *current* and *current* as *next*.
- **Step 5 -** Finally, return the *current/next* variable since it contains the number in Nth position and display the result.

Lab Assignment

1. Write the first natural solution that you find to the problem (an inefficient algorithm) and implement it to find nth number of Fibonacci number F(n)

   Hint: Use Recursive algorithm to find nth Fibonacci number

2. Write an efficient algorithm and implement it to find nth number of Fibonacci number F(n)

   Hint: Use Iterative algorithm(1 or 2) to find nth Fibonacci number

3. Record the time it takes to execute 120th Fibonacci number on both algorithms

   Hint: You can use clock() function to record execution time.

4. Fill out the report sheet, compare and explain your results

   Hint: Compare and explain why do you think one is efficient than the other.

Coding guidelines

- Start coding by creating a function/method which takes an integer argument N. N is the input for which the Fibonacci number should be displayed.
- Write the code to calculate the Nth Fibonacci number (Recursive or iterative) in the function defined above.
- In the main function, take the user input and call the created method.
- Also, write the code for calculating the execution time in the main function.
- Print the output.

Questions?