GROUP 22

Lovely Digra IIT2017090
Abhishek Vishwakarma IIT2017091
Anuj Raj IIT2017092

Problem Statement

From an array of 1000 random numbers , 10^4 < A[i] <10^6, find the location of the Smallest fibonacci number if exists.

Introduction

Fibonacci number?

A number whose belongs to Set containing elements from Fibonacci Series, which goes like :

$$F(n) = F(n-1) + F(n-2) \qquad \text{for } n >= 2,\ F(1) = F(2) = 1$$

Algorithm

Simple(Naive) Approach :

> for checking if N is Fibonacci no. or not one can generate the series till the nth fibonacci is equal or greater than N i.e

Fib(n) >= N.

> If it is equal , check if it is smaller than previous smallest Fib from the array.

> Repeat this for each element in the Array .

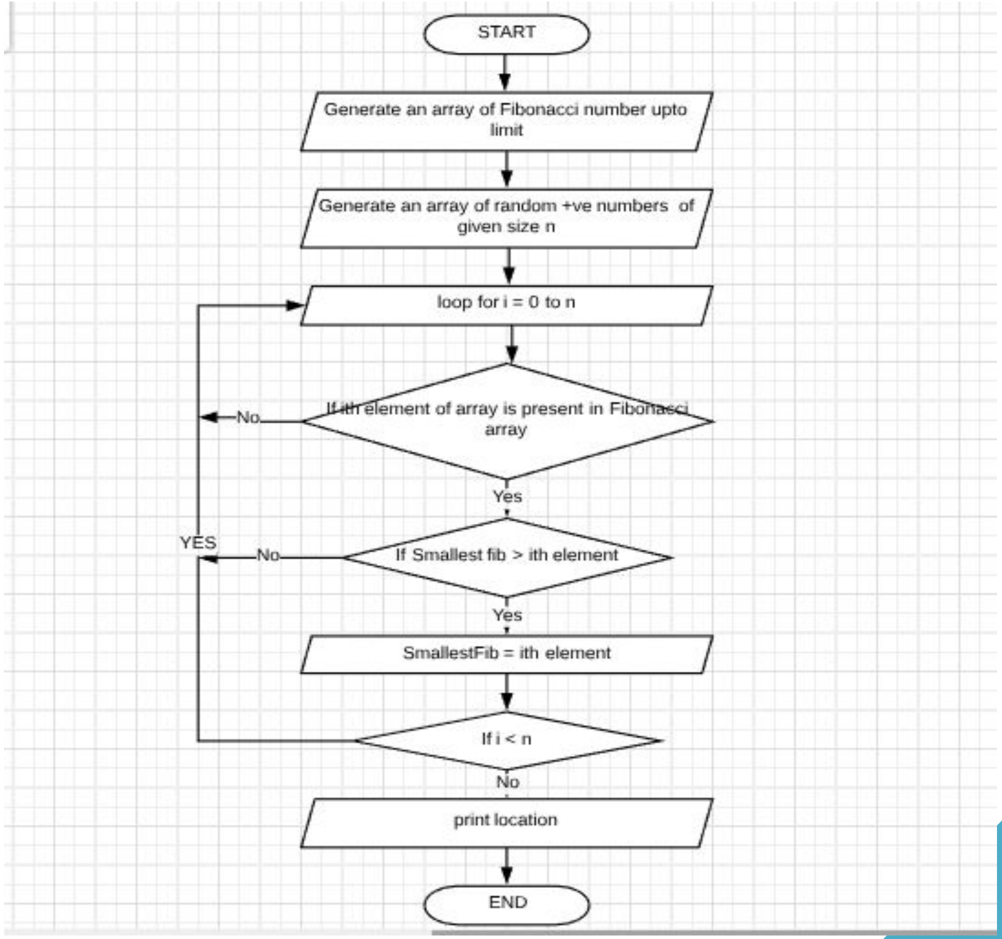
> At the end print the location stored of the smallest fib in Array if exists.

Algorithm

Optimisation :

> A Look Up Table containing Fibonacci numbers < 10^6 can be generated .
> Now only the task is to check for each element to be present in Look up Table which done through binary Search.
> If present check for smallest Fib.

Flowchart


START
Generate an array of Fibonacci number upto limit
Generate an array of random +ve numbers of given size n
loop for i = 0 to n
If ith element of array is present in Fibonacci array
No
Yes
YES
If Smallest fib > ith element
No
Yes
SmallestFib = ith element
If i < n
No
print location
END

Pseudo Code

```
Iterations ← n
Small ← Int_Max
Leftlimit  ← 10^4
Rightlimit ← 10^6
Procedure createFibMap(a0, a1, rightlimit)
Fib[0] ← a0 ,        a0 = 1
Fib[1] ← a1,         a1 = 1
For all i = 1to rightLimit do
        Fib[i] = fib[i-1] + fib[i-2]
Procedure randArray(arr, size, leftLimit, rightLimit)
For all i = 1 to size do
        arr[i] = rand()%((rightLimit-1)-(leftLimit+1)+1) + leftLimit+1
For all i =1 to size do
        //Check if arr[i] is present in fib[]
If binarySearch(arr[i]) == true then
        Small ← i
If small != Int_Max then
        Print small
```

Experimental Analysis

The Best Case

Best Case is when All the elements in the Random generated Array is the fibonacci number in the middle of the lookUp Table.
with growth rate lower bound $\Omega(n)$
where ‘n’ is the size of the random generated Array.

The Worst Case

Worst Case arises when none of the element in the array is a fibonacci.The growth rate will be bounded by

$O(n * \log(m))$

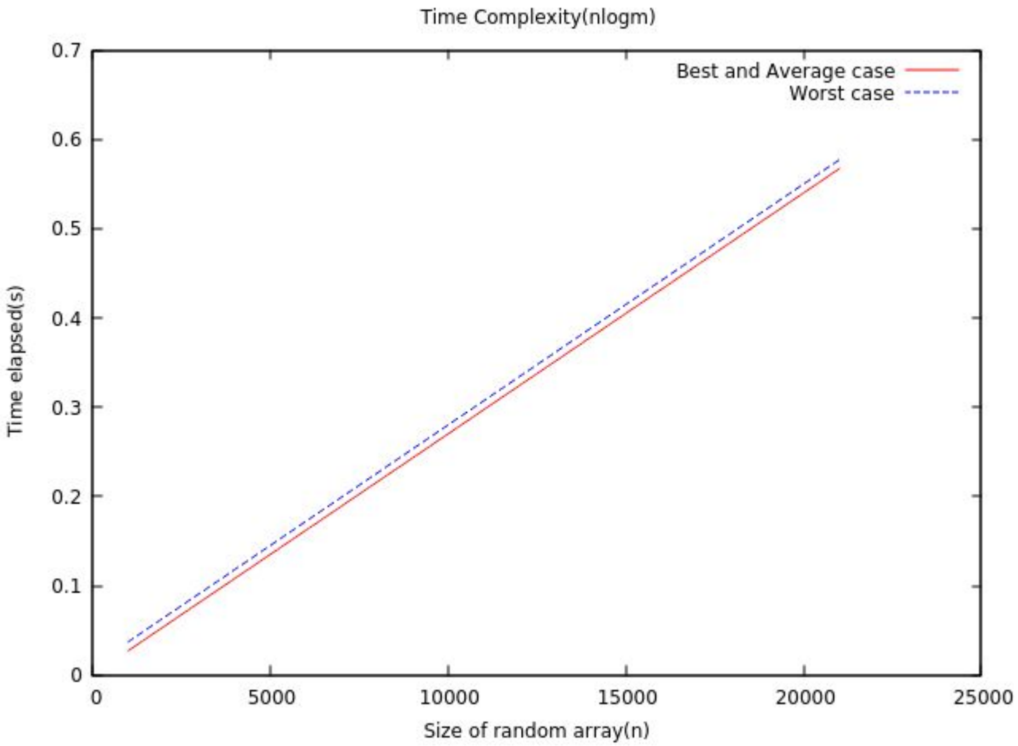
Where ‘m’ is the total fib nos in the Lookup Table.

But since ‘m’ is constrained due to the condition that Fib(n) < 10^6, it becomes a constant.

Thus it is omitted and Finally growth rate becomes

$O(n)$

The Graph


Time Complexity(nlogm)
Best and Average case
Worst case
Time elapsed(s)
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0
5000
10000
15000
20000
25000
Size of random array(n)

Conclusion

The Algorithm is correct and optimal under the constraints defined in the Problem Statement.

Thank You!