Koneru Lakshmaiah Education Foundation

(Deemed to be University)

FRESHMAN ENGINEERING DEPARTMENT

A Project Based Lab Report

On

FIBONACCI STRINGS

SUBMITTED BY:

2400032350 NITHYA SAHITHEE.G

UNDER THE GUIDANCE OF

Mr. E. Sai Krishna

Assistant Professor, CSE.

KL UNIVERSITY
Green fields, Vaddeswaram – 522 502
Guntur Dt., AP, India.

DEPARTMENT OF BASIC ENGINEERING SCIENCES-1

CERTIFICATE

This is to certify that the project based laboratory report entitled "**Fibonacci Strings**" submitted by Ms.**NITHYA SAHITHEE GAJULAVARTHI** bearing **Regd. No.2400032350** to the **Department of Basic Engineering Sciences-1, KL University** in partial fulfillment of the requirements for the completion of a project-based Laboratory in "**Computational Thinking for Structural Design**" course in I B Tech I Semester, is a bonafide record of the work carried out by them under my supervision duringthe academic year 2024 – 2025.

Mr E. Sai Krishna	Dr. D.Haritha
PROJECT SUPERVISOR	HEAD OF THE DEPARTMENT

ACKNOWLEDGEMENTS

It is great pleasure for me to express my gratitude to our honorable President **Sri. Koneru Satyanarayana**, for giving the opportunity and platform with facilities in accomplishing the project based laboratory report.

I express the sincere gratitude to our Director **Dr. A.Jagadeesh** for his administration towards our academic growth.

I express sincere gratitude to HOD-BES-1 **Dr. D.Haritha** for her leadership and constant motivation provided in successful completion of our academic semester. I record it as my privilege to deeply thank for providing us the efficient faculty and facilities to make our ideas into reality.

I express my sincere thanks to our project supervisor Mr. E. Sai Krishna for his novel association of ideas, encouragement, appreciation and intellectual zeal which motivated us to venture this project successfully.

Finally, it is pleased to acknowledge the indebtedness to all those who devoted themselves directly or indirectly to make this project report success.

NITHYA SAHITHEE.G 2400032350

ABSTRACT

FIBONACCI STRINGS

- Fibonacci String Generation: • To construct the Fibonacci string for a given n, use the recursive relation to concatenate the previous two strings (F(n-1) and F(n-2)).

- Substring Search: • For each string si, count how many times it appears as a substring in the given Fibonacci string Fn. This can be done by a standard substring search method like KMP algorithm or using strstr (if you are allowed to use the C standard library).

- Efficiency: • Directly generating Fibonacci strings for large n can result in very large strings, so be mindful of memory usage and performance. Dynamic programming or memoization might be helpful if n is large.

- C Implementation Plan: • First, generate the Fibonacci string for the given n. • For each si, use a substring search algorithm to count the occurrences in the Fibonacci string

- Core Algorithm: The implementation leverages a bottom-up dynamic programming technique to avoid redundant calculations and optimize performance.

- Memory Optimization: Techniques for minimizing memory usage, especially for large Fibonacci strings, are discussed.

INDEX

INTRODUCTION

Fibonacci strings, a fascinating sequence of strings derived from the renowned Fibonacci number sequence, offer a unique blend of mathematical elegance and practical applications. In this paper, we delve into the world of Fibonacci strings and present a C implementation to efficiently generate and analyze these intriguing patterns. Fibonacci strings offer an interesting way to explore recursion, string manipulation, and sequence generation in C. Implementing Fibonacci strings in C provides practice with string handling, recursion, and dynamic memory allocation if the sequence becomes large.

AIM

The aim of a Fibonacci series program in C is to generate and display the Fibonacci sequence, where each number is the sum of the two preceding ones. Typically, the series starts with 0 and 1. The goal is to calculate the sequence up to a specified number of terms or a given position in the series. This is achieved through iteration or recursion. The Fibonacci series has various applications in mathematics, computer science, and nature.

ADVANTAGES & DISADVANTAGES

Advantages:- 1. *Learning Recursive and Iterative Techniques* - Teaches recursive and iterative programming approaches.

2. *Understanding Time Complexity and Optimization* - Demonstrates complexity concepts and the need for optimization.

3. *Memory Management Practice* - Offers experience with manual memory management, pointers, and dynamic memory allocation.

4. *Mathematical and Real-World Applications* - Shows relevance to real-world problems and applications in fields like mathematics and data structures.

5. *Foundational Algorithmic Thinking* - Helps in learning advanced concepts like memoization and dynamic programming.

Disadvantages:- 1. *Inefficient Recursive Approach*: Using recursion for Fibonacci can be inefficient as it involves repeated calculations of the same values, resulting in an exponential time complexity (O(2^n)). This can lead to stack overflow for large values of n.

2. *Limited Precision*: The int or long data types in C have fixed size limits, so they can quickly overflow for larger Fibonacci numbers, causing incorrect results.

3. *Memory Usage*: Recursive functions use stack memory for each function call. Large values of n can lead to high memory usage and stack overflow in recursive implementations.

4. *Complexity in Optimization*: Optimizing Fibonacci calculations in C often requires techniques like memoization or dynamic programming, which can increase code complexity and reduce readability.

5. *Lack of Built-in Big Integer Support*: C does not have native support for big integers, so calculating large Fibonacci numbers requires using libraries, which can be cumbersome and increase code dependency.

Future Enhancements

1. Memoization:

Implement memoization to store previously calculated Fibonacci numbers and reuse them, improving efficiency for large inputs.

2. Matrix Exponentiation:

Use matrix exponentiation for faster computation of Fibonacci numbers, especially for large n, reducing the time complexity to O(log n).

3. Iterative Solution:

While the recursive solution works, it can be optimized with an iterative approach to avoid stack overflow for large numbers.

4. Dynamic Programming:

Implement dynamic programming to compute Fibonacci numbers in a bottom-up approach, saving space and time by storing intermediate results.

5. Handling Large Numbers:

Modify the program to handle large Fibonacci numbers, possibly using libraries that support big integers (like GMP) or implement your own large number handling.

6. Interactive User Input:

Improve user experience by adding interactive prompts and validation for the input number.

SYSTEM REQUIREMENTS

- **SOFTWARE REQUIREMENTS:**

  The major software requirements of the project are as follows:

Language	: C Compiler (e.g., GCC, Turbo C)

Operating System**:** Windows Xp or later. Windows XP or later, or any Linux-based operating system.

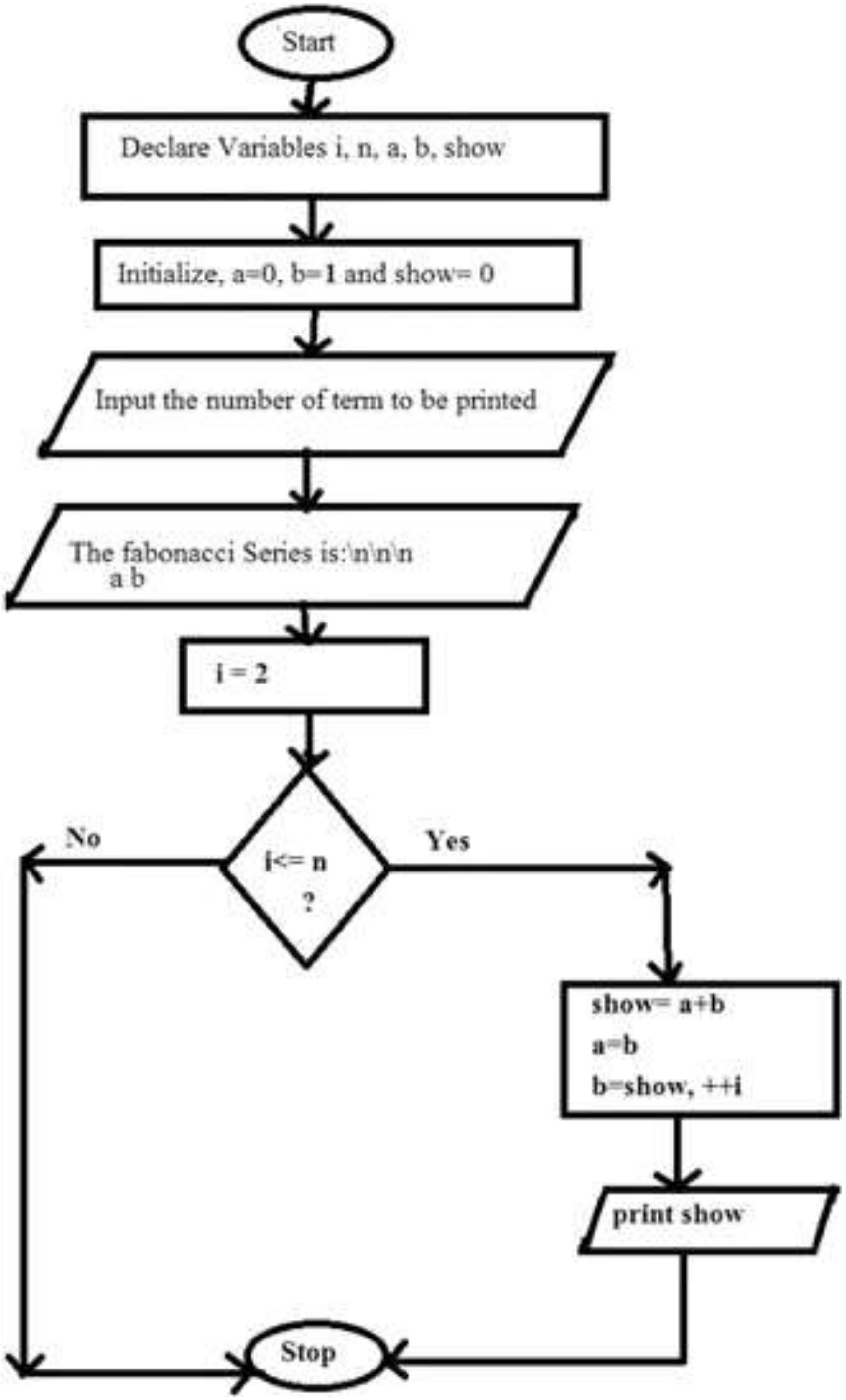
IDE (optional): Code::Blocks, Dev-C++, Visual Studio, or any preferred text editor.

- **HARDWARE REQUIREMENTS:**

The hardware requirements that map towards the software are as follows:

RAM	: 512 MB or more (recommended)
Processor	: 1 GHz or faster

FLOWCHART


Start
Declare Variables i, n, a, b, show
Initialize, a=0, b=1 and show= 0
Input the number of term to be printed
The fabonacci Series is:\n\n\n
a b
i = 2
i<= n
?
No
Yes
show= a+b
a=b
b=show, ++i
print show
Stop

ALGORITHM

The Fibonacci series in C can be explained in the following steps:

1. Initialization

The first two numbers in the Fibonacci series are always 0 and 1. These will be the starting values.

2. Input

Ask the user for the number of terms they want in the Fibonacci sequence.

3. Variable Setup

Initialize three variables:

- a: to store the first number in the series (0).
- b: to store the second number in the series (1).
- c: to store the next number, which will be the sum of a and b.

4. Loop

Use a loop to generate the Fibonacci series up to the number of terms requested.

- For each iteration, calculate the next term (c) as the sum of the previous two terms (a and b).
- Print c, then update a and b for the next iteration:
  - Set a to b (previous second number).
  - Set b to c (new number).

5. End Condition

The loop runs until all terms are printed based on the user's input.

6. Output

Display the Fibonacci sequence.

This is the general approach to generating the Fibonacci series using iteration in C.

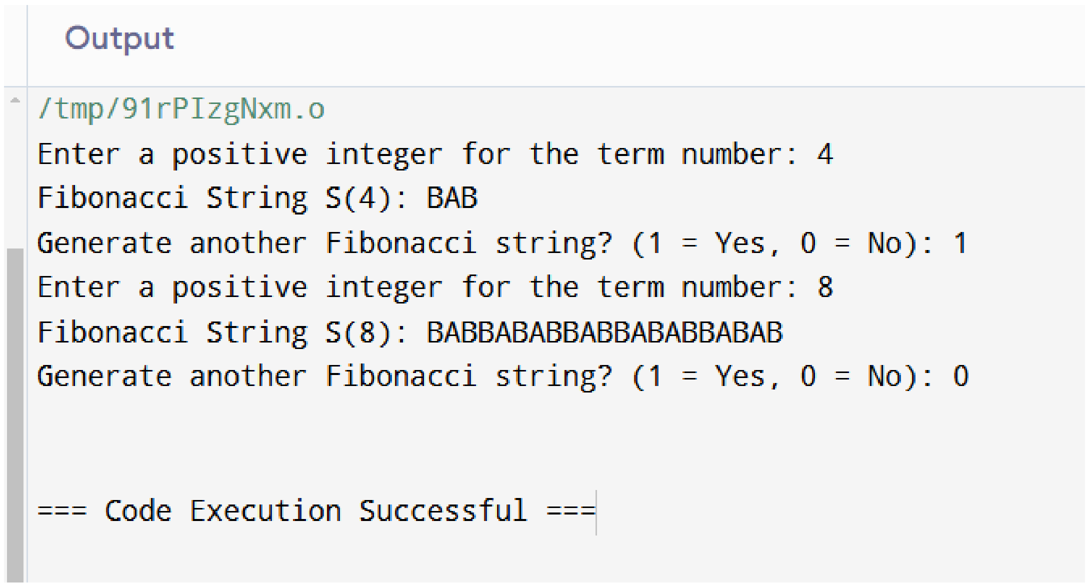
IMPLEMENTATION

```
#include  <stdio.h>
#include <stdlib.h>
#include <string.h>
char* fibonacciString(int n) {
    if (n == 1) return strdup("A");
    if (n == 2) return strdup("B");
    char* term1 = strdup("A");
    char* term2 = strdup("B");
    char* nextTerm;
    for (int i = 3; i <= n; i++) {
        nextTerm = (char*)malloc(strlen(term1) + strlen(term2) + 1);
        strcpy(nextTerm, term2);
        strcat(nextTerm, term1);
        free(term1);
        term1 = term2;
        term2 = nextTerm;
    }
    free(term1);
    return term2;
int main() {
    int term, repeat;
    do {
        printf("Enter a positive integer for the term number: ");
        if (scanf("%d", &term) != 1 || term < 1) break;
        char* result = fibonacciString(term);
        printf("Fibonacci String S(%d): %s\n", term, result);
        free(result);
        printf("Generate another Fibonacci string? (1 = Yes, 0 = No): ");
        if (scanf("%d", &repeat) != 1 || repeat == 0) break;
        while (getchar() != '\n');
    } while (repeat == 1);
    return 0;
}
```

INTEGRATION AND SYSTEM TESTING

OUTPUTS

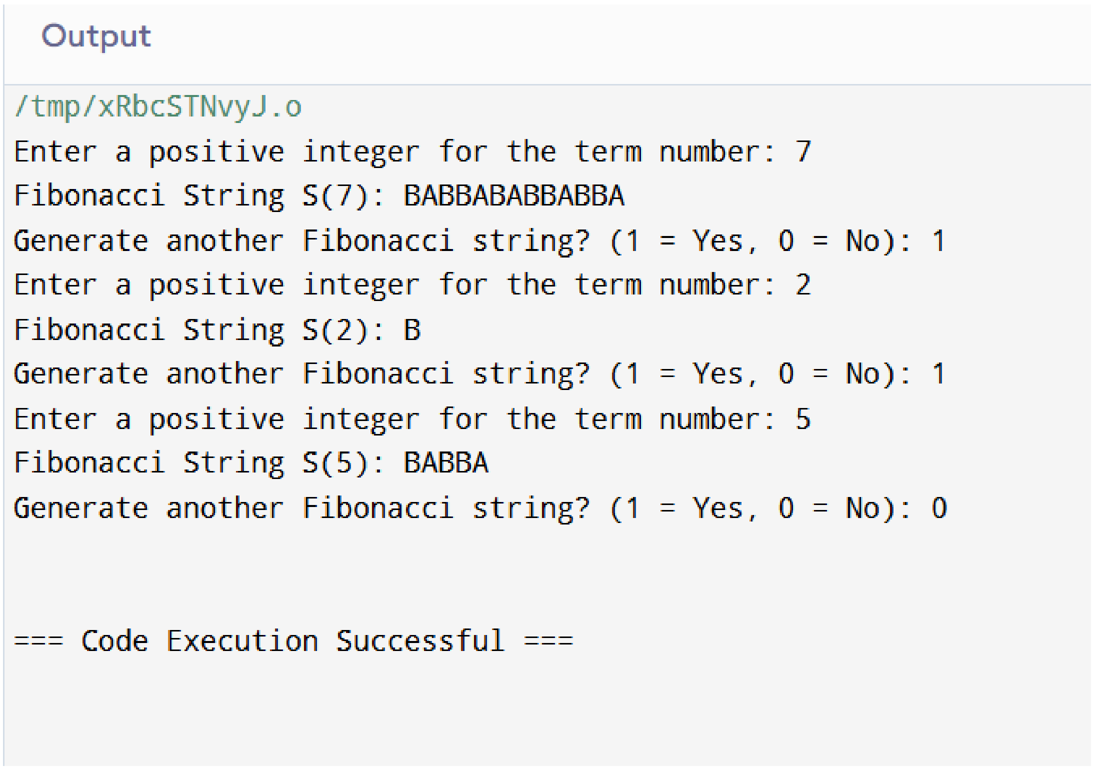
SCREEN SHOTS:



Output

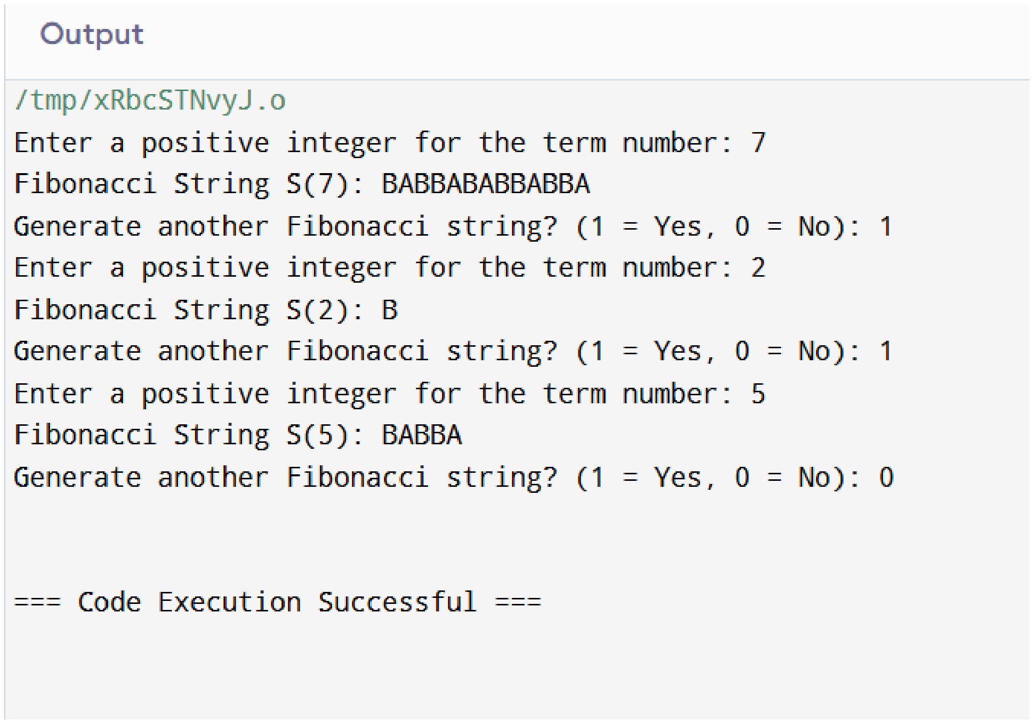
```
/tmp/91rPIzgNxm.o
Enter a positive integer for the term number: 4
Fibonacci String S(4): BAB
Generate another Fibonacci string? (1 = Yes, 0 = No): 1
Enter a positive integer for the term number: 8
Fibonacci String S(8): BABBABABBABBABABBABAB
Generate another Fibonacci string? (1 = Yes, 0 = No): 0


=== Code Execution Successful ===
```


Output

```
/tmp/xRbcSTNvyJ.o
Enter a positive integer for the term number: 7
Fibonacci String S(7): BABBABABBABBA
Generate another Fibonacci string? (1 = Yes, 0 = No): 1
Enter a positive integer for the term number: 2
Fibonacci String S(2): B
Generate another Fibonacci string? (1 = Yes, 0 = No): 1
Enter a positive integer for the term number: 5
Fibonacci String S(5): BABBA
Generate another Fibonacci string? (1 = Yes, 0 = No): 0


=== Code Execution Successful ===
```



Output

```
/tmp/xRbcSTNvyJ.o
Enter a positive integer for the term number: 7
Fibonacci String S(7): BABBABABBABBA
Generate another Fibonacci string? (1 = Yes, 0 = No): 1
Enter a positive integer for the term number: 2
Fibonacci String S(2): B
Generate another Fibonacci string? (1 = Yes, 0 = No): 1
Enter a positive integer for the term number: 5
Fibonacci String S(5): BABBA
Generate another Fibonacci string? (1 = Yes, 0 = No): 0


=== Code Execution Successful ===
```

CONCLUSION

The Fibonacci series is a sequence of numbers where each number is the sum of the two preceding ones, starting from 0 and 1. In C, it can be generated using iteration or recursion. The iterative approach is more efficient for larger values due to lower time complexity. The Fibonacci series is widely used in mathematics, computer science, and various applications such as algorithm optimization and cryptography. By understanding and implementing Fibonacci in C, one can improve problem-solving skills and gain insight into recursive and iterative programming techniques.

Fibonacci strings follow a simple recursive pattern, where each string is formed by concatenating the two previous ones. This sequence grows exponentially in length, similar to the Fibonacci numbers. The study of Fibonacci strings highlights how simple rules can lead to complex structures, with applications in areas like computer science and mathematics.Fibonacci strings follow a simple recursive pattern, where each string is formed by concatenating the two previous ones. This sequence grows exponentially in length, similar to the Fibonacci numbers. The study of Fibonacci strings highlights how simple rules can lead to complex structures, with applications in areas like computer science and mathematics.

REFERENCES

1.Reference Books:

"The Art of Computer Programming, Volume 1: Fundamental Algorithms" by Donald E. Knuth.

The C Programming Language.

2. Research Papers:

Baghery, Abdolhossein, & Behzad, Mahdi. (2014). "A Simple Proof of Fibonacci String Formula."

3.Online Resources:

3. TutorialsPoint - Fibonacci Series in C

URL: https://www.tutorialspoint.com/cprogramming/c_program_fibonacci_series.htm