

# **PROJECT PRESENTATION**

**TITLE : FIBONACCI SERIES AND BINARY SEARCH  
USING RECURSION**

**TEAM NAME: TEAM B**

**TEAM MEMBERS: ABIESH, CHANDRU, BHARATH,  
DANSON, ANJANA, DEEPA.**

# **DEFINITION OF RECURSIVE FUNCTION**

Any function that happens to call itself again and again (directly or indirectly), unless the program satisfies some specific condition/subtask is called a recursive function.

# WORKING FLOW OF RECURSIVE FUNCTION

How does recursion work?

```
void recurse()  
{  
    ... ..  
    recurse();  
    ... ..  
}  
  
int main()  
{  
    ... ..  
    recurse();  
    ... ..  
}
```

The diagram illustrates the working flow of a recursive function. It shows two function definitions: `void recurse()` and `int main()`. Inside `recurse()`, there is a call to `recurse();`. Inside `main()`, there is a call to `recurse();`. Arrows indicate the flow of execution: one arrow points from the `recurse();` line in `main()` to the `void recurse()` line, and another arrow points from the `recurse();` line inside `recurse()` back to the `void recurse()` line. A bracket labeled "recursive call" spans the `recurse();` line in `recurse()` and the arrow pointing back to the `void recurse()` line.

# PROJECT OVERVIEW

**PROJECT DESCRIPTION:** TO FIND THE FIBONACCI SERIES FOR GIVEN NUMBER.

**CONCEPTS USED:** Function and conditional statements.

## LOGIC OF FIBONACCI SERIES



### FIBONACCI SERIES

Default

0 1 1 2 3 5 8 13

$$0 + 1 = 1$$

$$1 + 1 = 2$$

$$1 + 2 = 3$$

$$2 + 3 = 5$$

$$3 + 5 = 8$$

$$5 + 8 = 13$$

## C CODE

```
#include<stdio.h>
int fib(int n,int n1,int n2)
{
    int n3=n1+n2;
    n1=n2;
    n2=n3;
    printf("\n%d",n3);
    if(n>3)
    {
        fib(n-1,n1,n2);
    }
}
void main()
{
    int n,n1=0,n2=1;
    printf("enter the number of terms:");
    scanf("%d",&n);
    printf("\n%d\n %d",n1,n2);
    fib(n,n1,n2);
}
```

# OUTPUT

## INPUT:

Enter the number of terms: 5

## OUTPUT:

0  
1  
1  
2  
3

# **PROJECT OVERVIEW**

**PROJECT DESCRIPTION: TO FIND THE TARGET  
ELEMENT IN AN ARRAY**

**CONCEPTS USED: FUNCTION AND ARRAY OF  
STRINGS..**



# LOGIC OF BINARY SEARCH

Binary Search											
	0	1	2	3	4	5	6	7	8	9	
Search 23	2	5	8	12	16	23	38	56	72	91	
	L=0	1	2	3	M=4	5	6	7	8	9	
23 > 16 take 2 <sup>nd</sup> half	2	5	8	12	16	23	38	56	72	91	
	0	1	2	3	4	L=5	6	M=7	8	H=9	
23 < 56 take 1 <sup>st</sup> half	2	5	8	12	16	23	38	56	72	91	
	0	1	2	3	4	L=5, M=5	H=6	7	8	9	
Found 23, Return 5	2	5	8	12	16	23	38	56	72	91	

# C CODE

```
#include <stdio.h>
int binarySearch(int arr[], int low, int high, int target)
{
    if (low <= high)
    {
        int mid =( low + high) / 2;

        if (arr[mid] == target)
            return mid;

        if (arr[mid] > target)
            return binarySearch(arr, low, mid - 1, target);
        else
        {
            return binarySearch(arr, mid + 1, high, target);
        }
    }

    return -1;
}
```

```
int main()
{
    int n, target;
    printf("Enter the number of elements in the array: ");
    scanf("%d", &n);
    int arr[n];
    printf("Enter %d elements in sorted order:\n", n);
    for (int i = 0; i < n; i++)
    {
        scanf("%d", &arr[i]);
    }
    printf("Enter the element to search: ");
    scanf("%d", &target);
    int result = binarySearch(arr, 0, n - 1, target);
    if (result != -1)
        printf("Element found at index %d\n", result);
    else
        printf("Element not found\n");

    return 0;
}
```

## OUTPUT:

## INPUT:

ENTER THE NUMBER OF ELEMENTS IN THE ARRAY:5  
ENTER 5 ELEMENTS IN SORTED ORDER:2 4 6 8 9  
ENTER THE ELEMENT TO SEARCH: 6

## OUTPUT:

The element is found at index 2

THANK YOU!!