

FIBONACCI AND GCD

Name: E. JAYANTH REDDY

Reg.No.: 19BCE7548

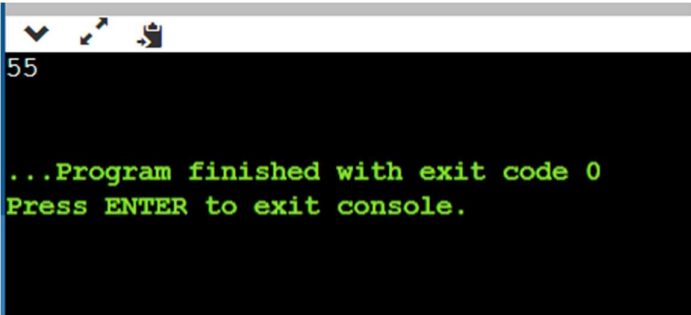
Fibonacci:

Using Naïve algorithm:

```
class Main
{
    static int fib(int n)
    {
        if (n <= 1)
            return n;
        return fib(n-1) + fib(n-2);
    }

    public static void main (String args[])
    {
        int n = 10;
        System.out.println(fib(n));
    }
}
```

OUTPUT:

A screenshot of a Java IDE's console window. The window has a title bar with standard OS icons. The console background is black with green text. The first line of output is '55'. Below it, there are two lines of status text: '...Program finished with exit code 0' and 'Press ENTER to exit console.'

```
55

...Program finished with exit code 0
Press ENTER to exit console.
```

Using DP:

```
class Main
{
```

```
static int fib(int n)
{

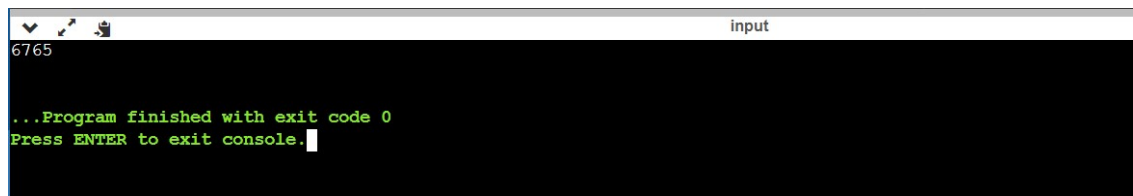
    int f[] = new int[n+2];

    f[0] = 0;
    f[1] = 1;

    for (int i = 2; i <= n; i++)
    {
        f[i] = f[i-1] + f[i-2];
    }

    return f[n];
}

public static void main (String args[])
{
    int n = 20;
    System.out.println(fib(n));
}
}
```

A screenshot of a terminal window with a title bar that includes standard window controls and the text 'input'. The terminal has a black background with green text. It displays the number '6765' in the top left corner. The main text in the terminal reads: '...Program finished with exit code 0' followed by 'Press ENTER to exit console.' with a white cursor at the end of the second line.

```
6765

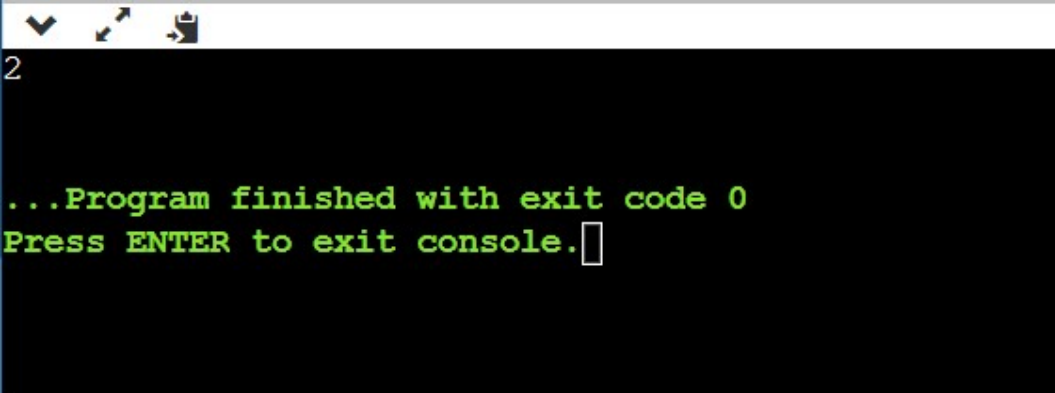
...Program finished with exit code 0
Press ENTER to exit console.
```

GCD:

Using Navie Algorithm:

```
import java.lang.Math;
import java.util.Scanner;
public class Main{
    static int GCD(int a,int b){
        int maximum=Math.max(a,b);
        int currentNumber=maximum-1;
        while(currentNumber>1){
            if((a%currentNumber==0)&&(b%currentNumber==0)){
                return currentNumber;
            }else{
                currentNumber--;
            }
        }
        return 1;
    }
    public static void main(String[] args){
        System.out.println(GCD(20,42));
    }
}
```

```
}
```

A terminal window with a black background and green text. The text reads: "...Program finished with exit code 0" followed by "Press ENTER to exit console." with a white cursor at the end. The window has a title bar with standard OS icons.

```
2

...Program finished with exit code 0
Press ENTER to exit console.
```

Using Euclidean Algorithm:

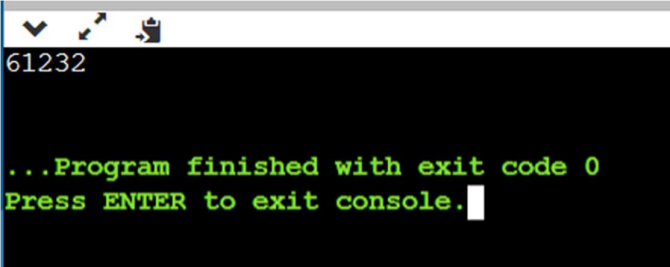
```
import java.util.*;
import java.lang.*;

class Main
{
    public static int gcd(int a, int b)
    {
        if (a == 0)
            return b;

        return gcd(b%a, a);
    }

    public static void main(String[] args)
    {
        System.out.println(gcd(3918848,1653264));
    }
}
```

OUTPUT:

A terminal window with a black background and green text. The text reads: "61232" followed by "...Program finished with exit code 0" and "Press ENTER to exit console." with a white cursor at the end. The window has a title bar with standard OS icons.

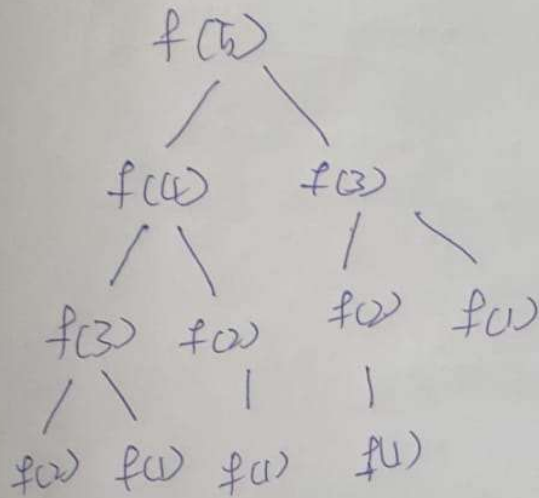
```
61232

...Program finished with exit code 0
Press ENTER to exit console.
```

ANALYSIS:

Fibonacci:-

1) using Naive approach:-



Here, it almost calling the method function at 2^n times

$\therefore O(2^n)$

2) using DP:-

Here we are saving the value of $f(5)$, $f(4)$ etc in an array. So at worst case we call it at $O(n)$ times

GCD:-

1) using naive algorithm:-

Here, we are calling the function for n times then time complexity is $O(n)$

2) using Euclidean algorithm:-

Here, we are calling the function for every $b \% a$ remainder. So, time complexity is $O(\log n)$