N^{th} Fibonacci Number

Problem

Find the n^{th} fibonacci number.

Fibonacci Numbers

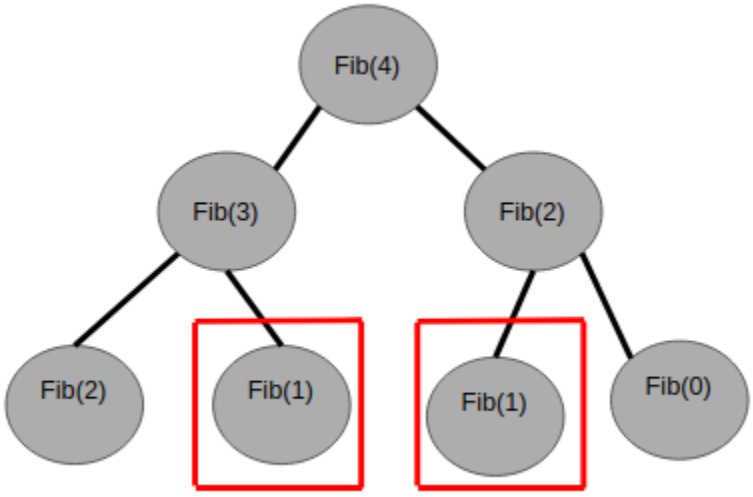
$$\{0, 1, 1, 2, 3, 5, 8, 13, ...\}$$

Let $f(n)$ be the n^{th} fibonacci number, then it follows

$$f(n) = f(n-1) + f(n-2)$$

Since we can write its recurrence relation, hence it follows optimal substructure property.

Its recursion tree looks like



In the above figure, we can see that Fib(1) repeats, hence it follows the overlapping subproblem property also.

Since it follows both optimal substructure and overlapping subproblem property hence we can apply dynamic programming here.

Code

Memoization

```
int fib[200] = {-1};

int computeFib(int n)
{
    if(n == 0 || n == 1)
        return n;

    if(fib[n] != -1)
        return fib[n];

    int res = fib[n-1] + fib[n-2];

    fib[n] = res; // memoization part

    return res;
}
```

Tabulation

```
int n;
cin >> n;

vi fib(n+1);
fib[1] = 0, fib[2] = 1;

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

cout << fib[n] << endl;
```

Time complexity: O(n)