Recursion & Memoization

Dev Karan Singh (devkaran1231)
Expert at codeforces (1817)
5 star at codechef (2040)

Today's plan

- Print increasing number from 1 to n
- Print decreasing number from n to 1
- Print factorial of a number
- Check if given string is a palindrome or not
- Linear search
- Nth fibonacci number (memoization)
- Print subsequence
- Count subsequence with given sum

Next Sunday (23rd)

Recursion:

↳ a function calling itself

```
f() {
    [        ]   Base case
    [ f()    ]   body of the function
}
```

Q print increasing number from 1 to n

$n = 5$

$1 \xrightarrow{+1} 2 \xrightarrow{+1} 3 \xrightarrow{+1} 4 \xrightarrow{+1} 5$

$i = 5$

$n = 5$

```
void print(int i, int n){
    if (i == n){
        cout << 1 << endl;
        return;
    }
    cout << i << "  ",
    print(i+1, n),
}
```

$i=1 \quad n=5$

$i=2, \quad n=5$

$i=3, \quad n=5$

output

1 2 3 4 5

$i=4,\ n=5$

$i=5,\ n=5$

$\boxed{5\ 4\ 3\ 2\ 1}$

1 2 3

done

i=1

```
print(i, 3){
  if(i==n){
    ~~cout, return n;~~
  }
  cout i
  print(2,3)
}
```

i=2

```
print(2,3){
  if(i==n){
    ~~cout return;~~
  }
  cout i
  print(3,3)
}
```

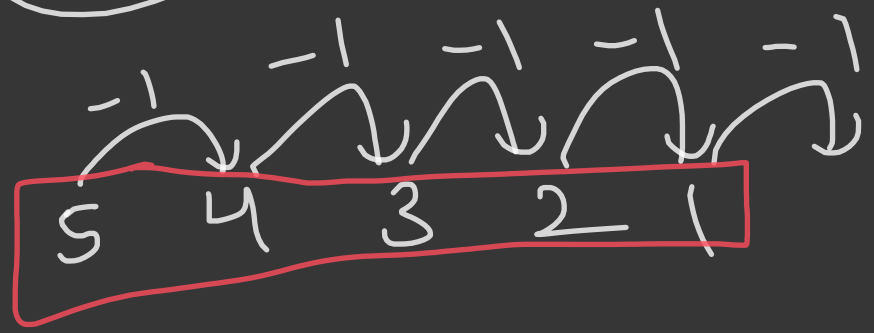
i=3

```
print(3,3){
  if(i==n){
    cout i;
    return,
  }
  ~~cout i;~~
  ~~print(4,3)~~
}
```

Q print decreasing from n to 1

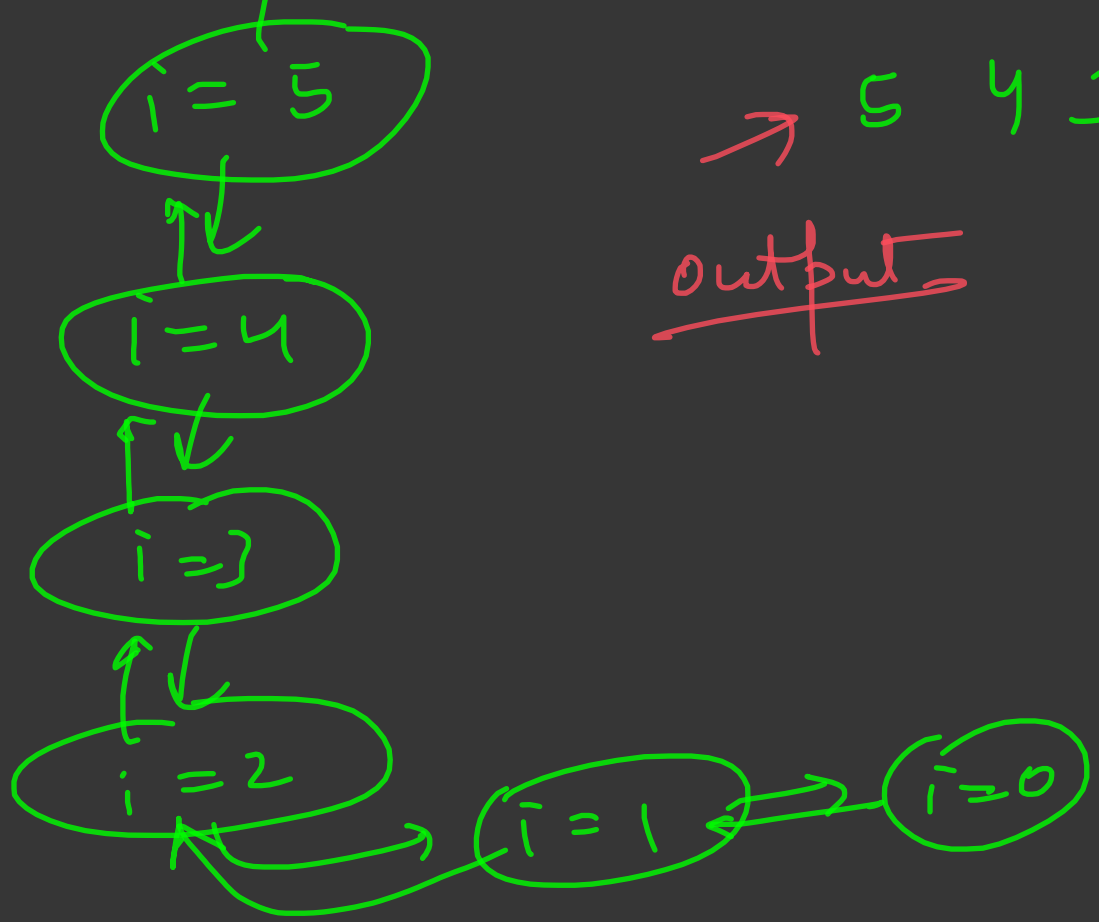
M1

n=5




```
void print (int i){
    if (i==0){
        return;
    }
    cout << i ;
    print (i-1);
}
```

recursion end



→ 5 4 3 2 1

output

M2

```
void print (int i, int n){
    if (i==n){
        cout 1; ★
        return;
    }
    print (i+1, n);
    cout 1; ★
}
```

$n = 3$

$i = 1$

```
print(1, 3){
  if(i==n){
    cout 1
    return,
  }
  print(2,3)
  cout i;
}
```

$i = 2$

```
print(2,3){
  if(i==n)
    cout 1
    return,
  }
  print(3,3)
  cout 1
}
```

$i = 3$

```
print(3, 3){
  if(i==n)
    cout 1,
    return,
  }
  print(4,3)
  cout,
}
```

output : 3 2 1

Q return factorial of a number (n)

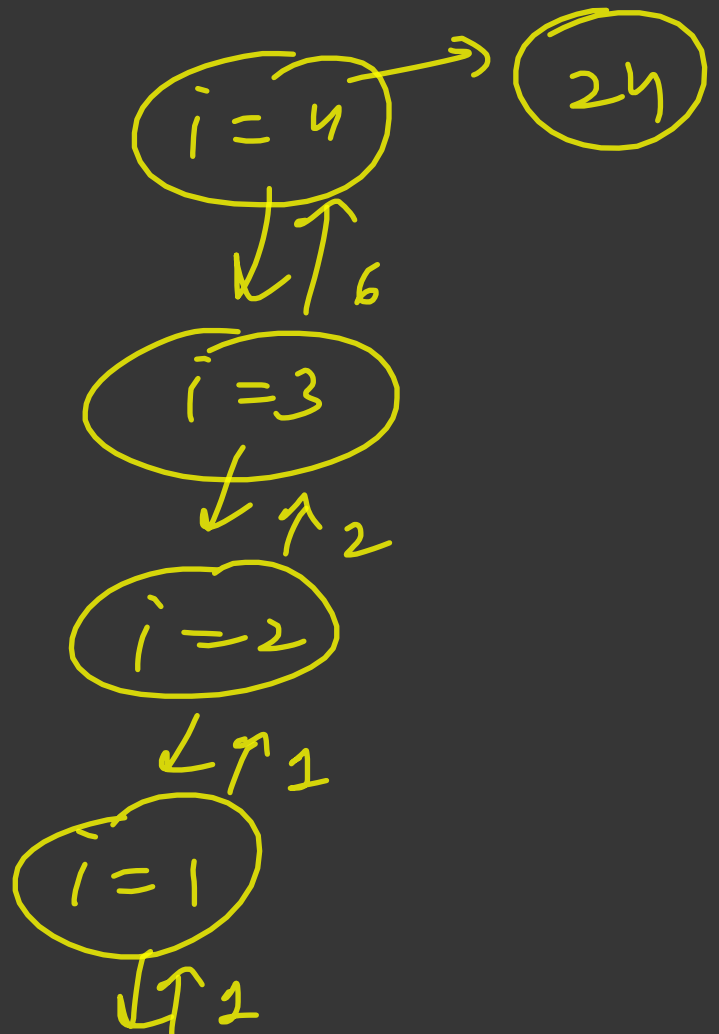
$0! = 1$

$2! = [0!] \times 1$

$2! = [1!] \times 2$

$3! = [2!] \times 3$

```
{ if (i == 0){
    return 1;
  }
  return fact(i-1)xi;
}
```


i = 4
24
6
i = 3
2
i = 2
1
i = 1
1

Q

$n = 7$

$i = 0$

			j		i	
0	1	2	3	4	5	6
a	b	c	d	c	b	a

```
if (j < i){
    return true;
}
if (s[i] != s[j]){
```

```
return false,
}
return
    rec(i+1, j-1);
```

Que Linear search

$x = 11$

	0	1	2	3	4	5
arr →	3,	7,	9,	11,	5,	4

$i = 0$ n an

```
find(i=0, n, an, x){
    if( i == n){
        return false;
    }
    if (an[i] == x){
        return true;
```

}

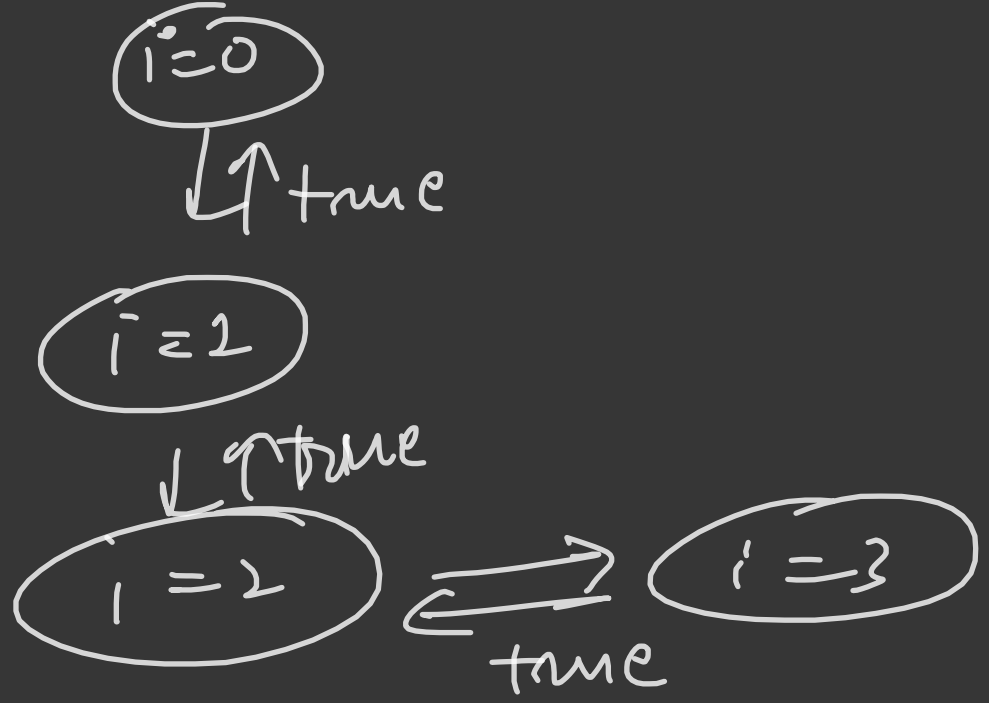
return find(i+1, n, an, x);

}

an → 3, 7, 9, 11, 5, 4

x = 11

true



arr → [1, 2, 3, 4]

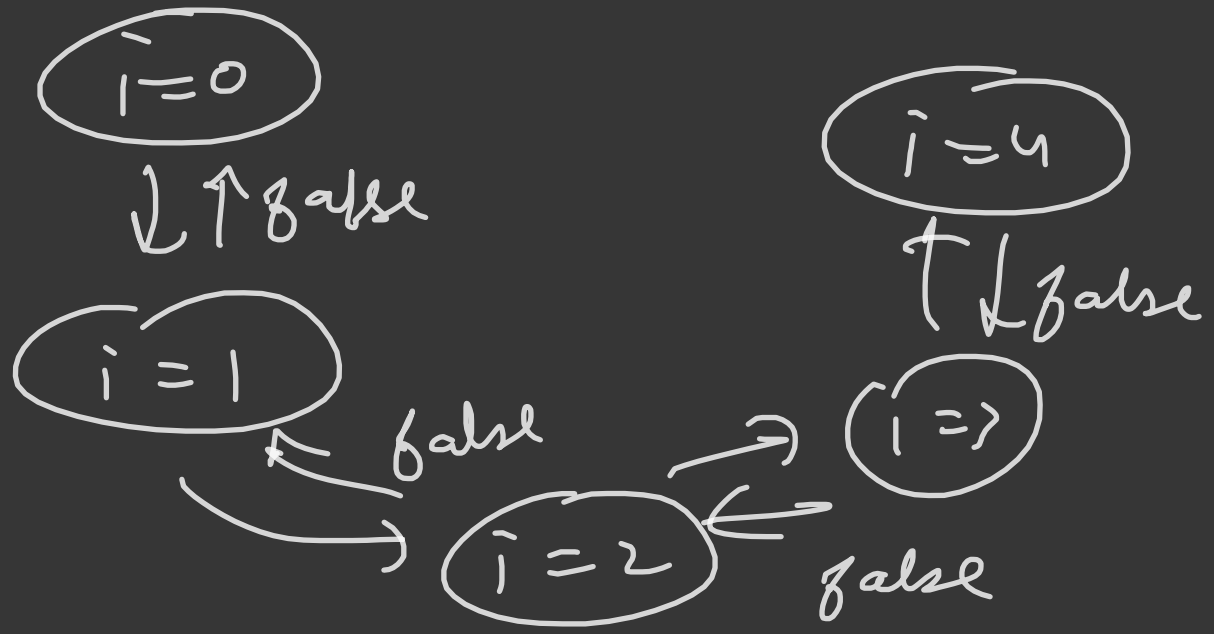
0 1 2 3

x = 5

↑ false

tc → O(N)

space → O(N)



Q return n^{th} fibonnaci number

0 1 1 2 3 5 8 13 --

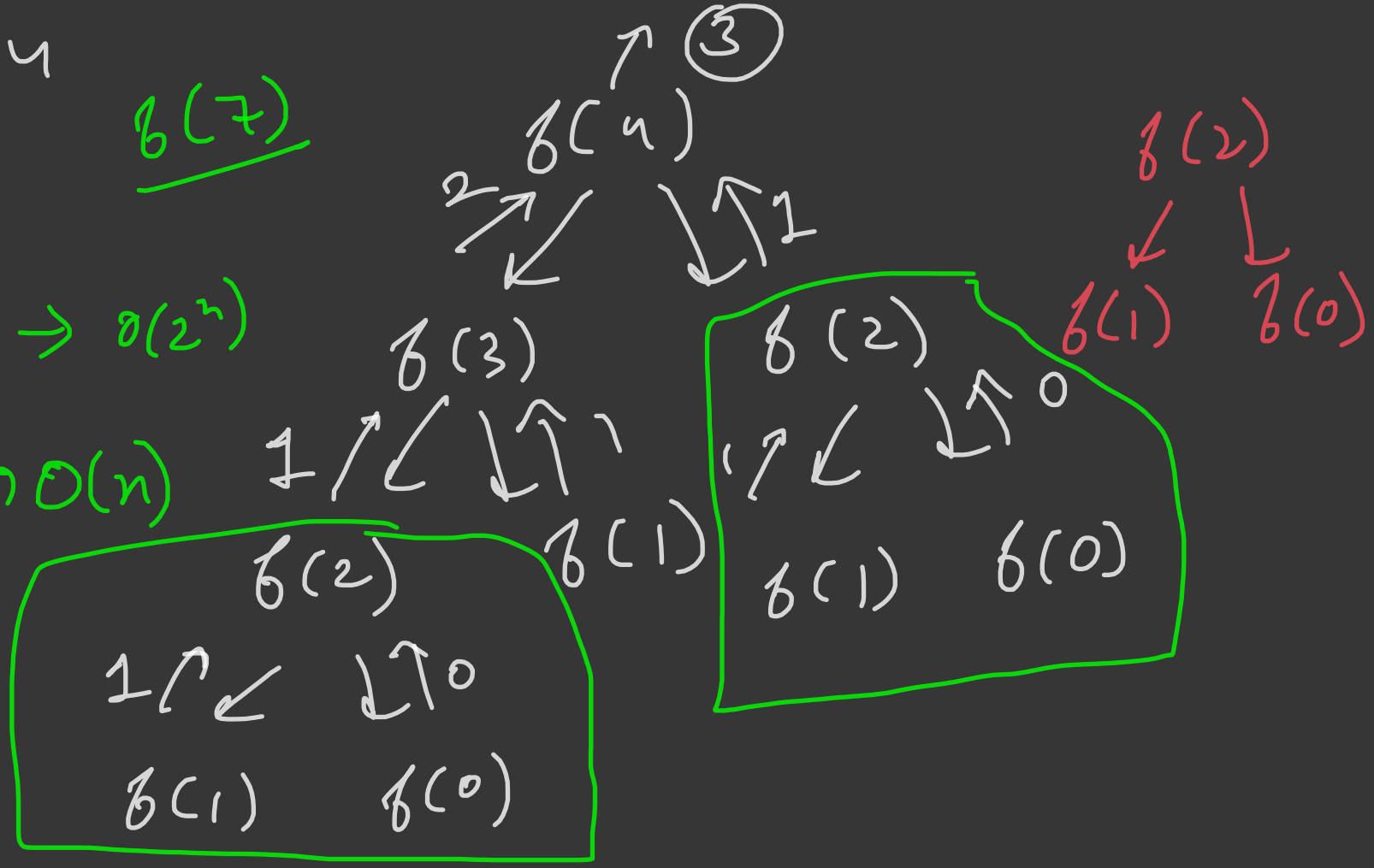
```
int fibo(int n){
    if(n<=1){
        return n;
    }
    return (fibo(n-1) + fibo(n-2));
}
```

$n = 4$

$f(7)$

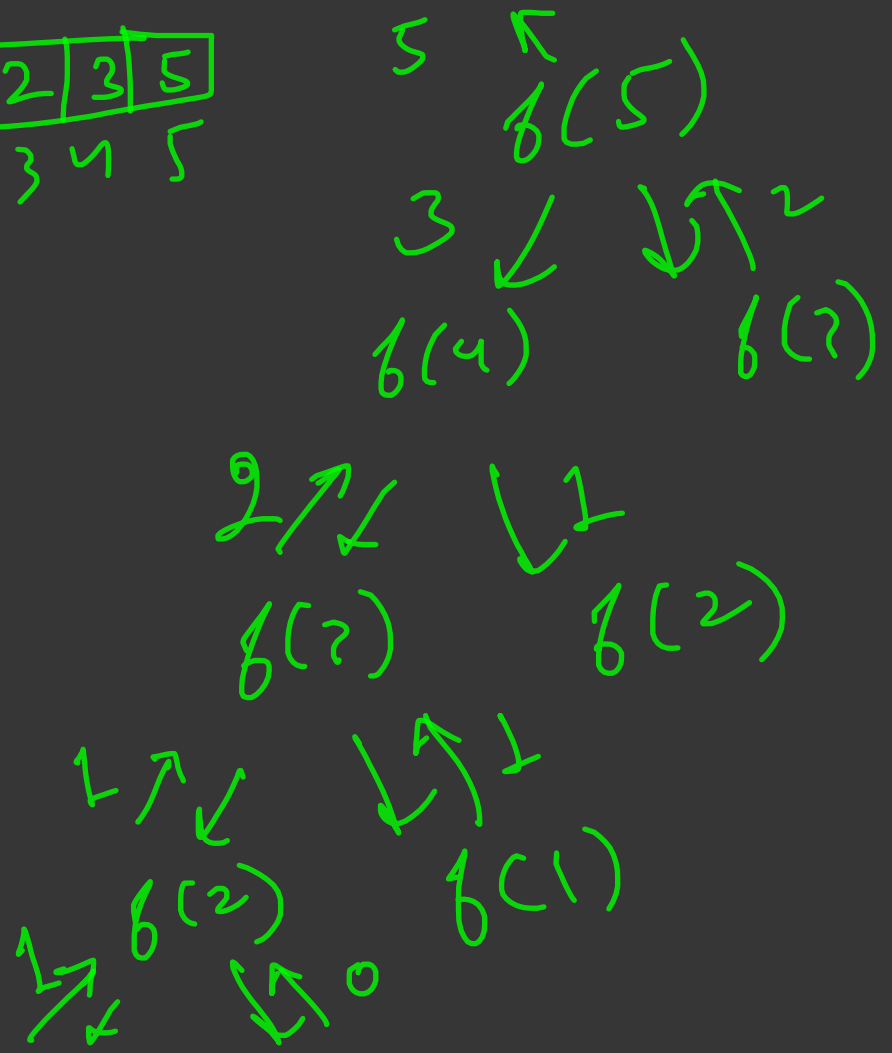
TC $\rightarrow O(2^n)$

SC $\rightarrow O(n)$



0	1	1	2	3	5
0	1	2	3	4	5

dp



$2^n \rightarrow n$

$TC \rightarrow O(n)$

$SC \rightarrow O(n)$

f(1) f(0)

```
int fibo(int n){
    if (n <= 1){
        return n;
    }
    if (dp[n] != -1){
        return dp[n];
```

$2^n \rightarrow n$

3 }

```
dp[n] = fibo(n-1) + fibo(n-2)
return dp[n]
}
```

an → [2, 7, 4, 5, 1]

depends on question

Que subarray v/s subsequence vs subset

subarray → contigious part of the array

subsequence → Non contigious but in order

Relative positions must be same

subset → Anything any order