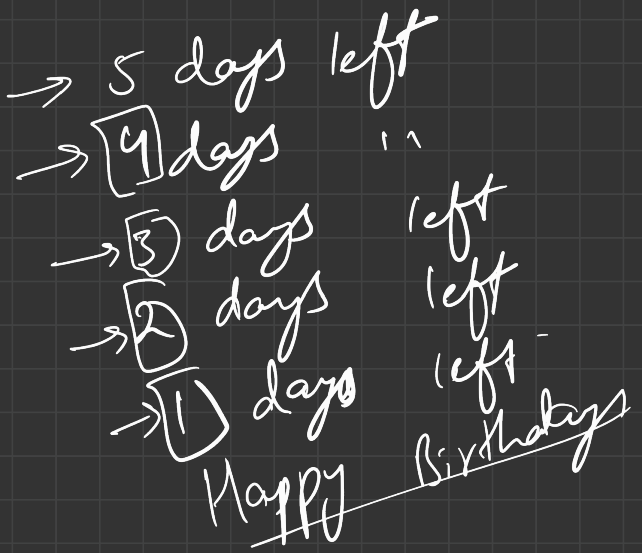
Recursion

* A function which call itself again and again
* Until a specific condition met



iterative approach

```
for(i=5 ; i>0;i--){
    cout <<i <<" day left "<<endl;
}
cout << " Happy Birthday Rishu";
```

stopage

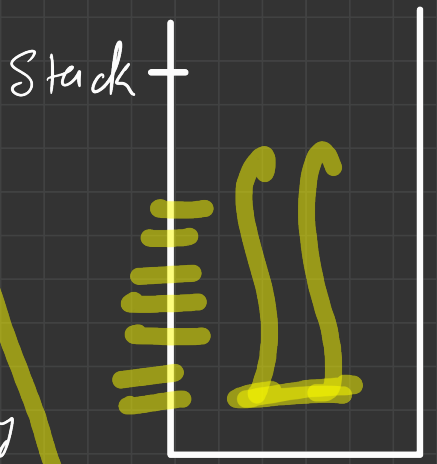
Iterative way == Recursive way

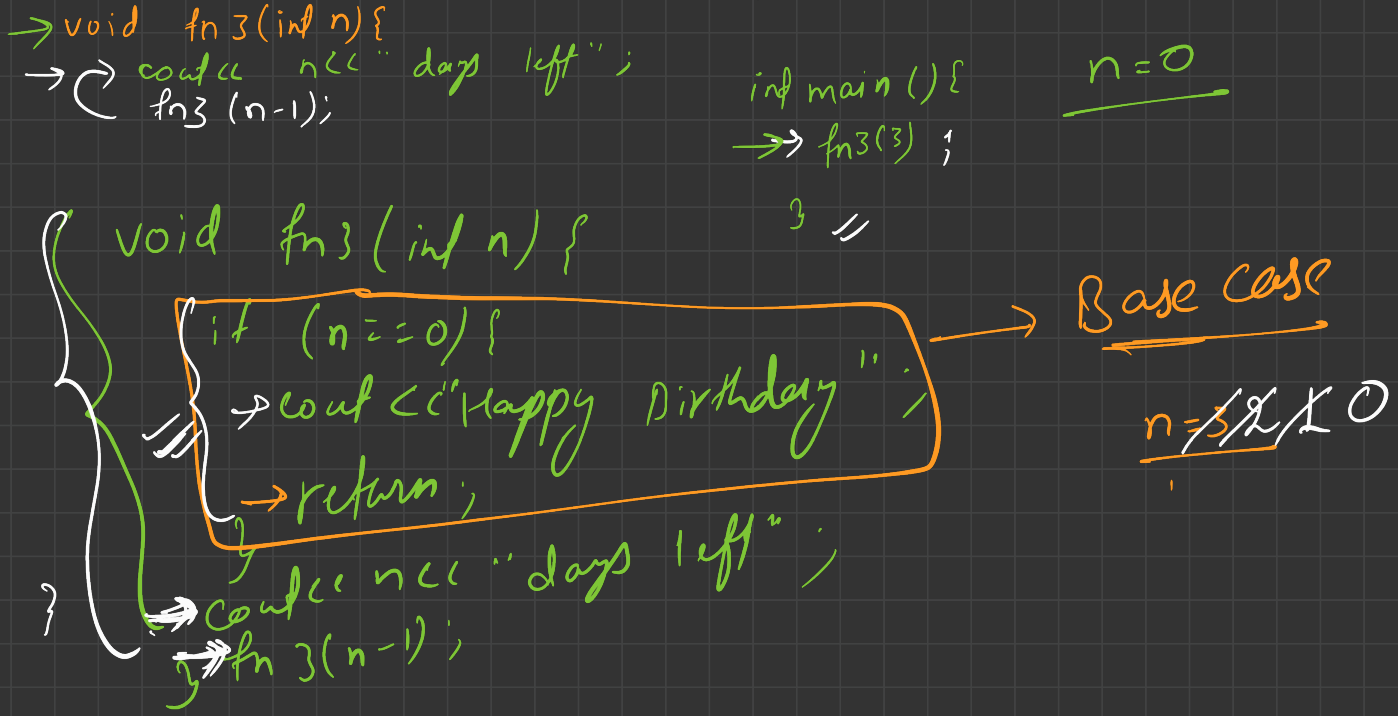
```
void fn3(int n){
  cout << n << " days left";
  fn2(n-1);
}
void fn2(int n){
  cout << n << " days left";
  fn1(n-1);
}
void fn1(int n){
  cout << n << " days left";
  fn0(n-1);
}
void fn0(int n){
  cout << "Happy Birthday";
}
```

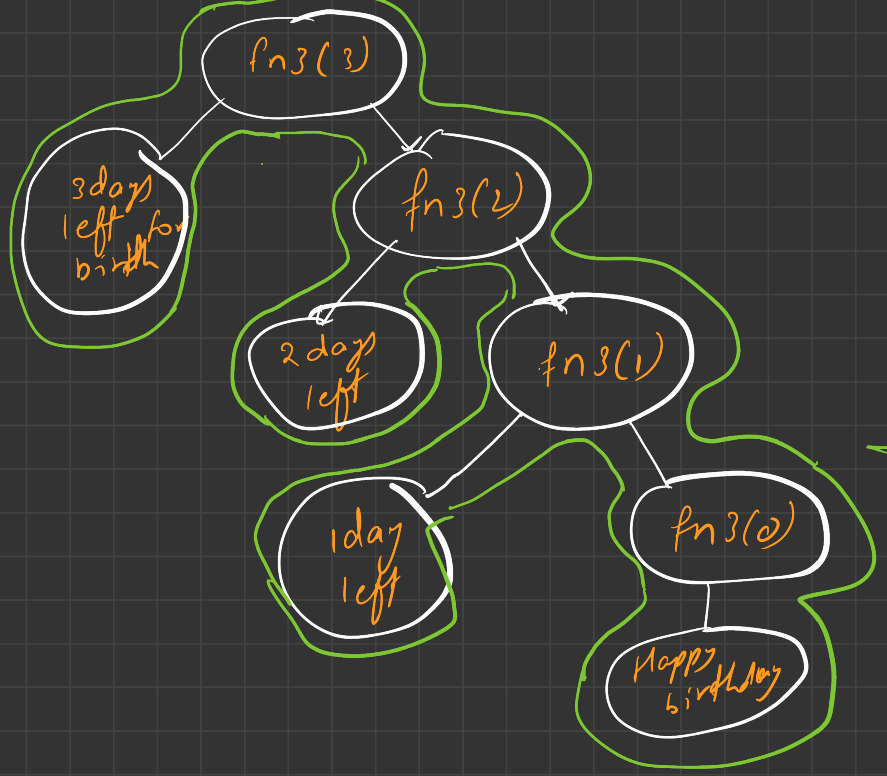
```
int main(){
  fn3(3);
}
```

=

3 days left
2 days left
1 days left
Happy Birthday







3 days left
2 days left
1 day left
Happy Birthday

→ Recursive Tree

Print(0) = Happy Birthday

Print(1) = 1 day left, Happy Birthday

Print(2) = 2 day left, 1 day left, Happy Birthday

Print(3) = 3 day left, Print(2)

1. Print(n) = n day left, Print(n-1)

```
void Print(int n){
    if(n==0){
        cout<<"Happy Birthday"; return;
    }
    cout<<n<<" days left";
    Print(n-1);
}
```

Print n to 1

```
for(i = n; i>0; i--){
    cout<<i;
}
```

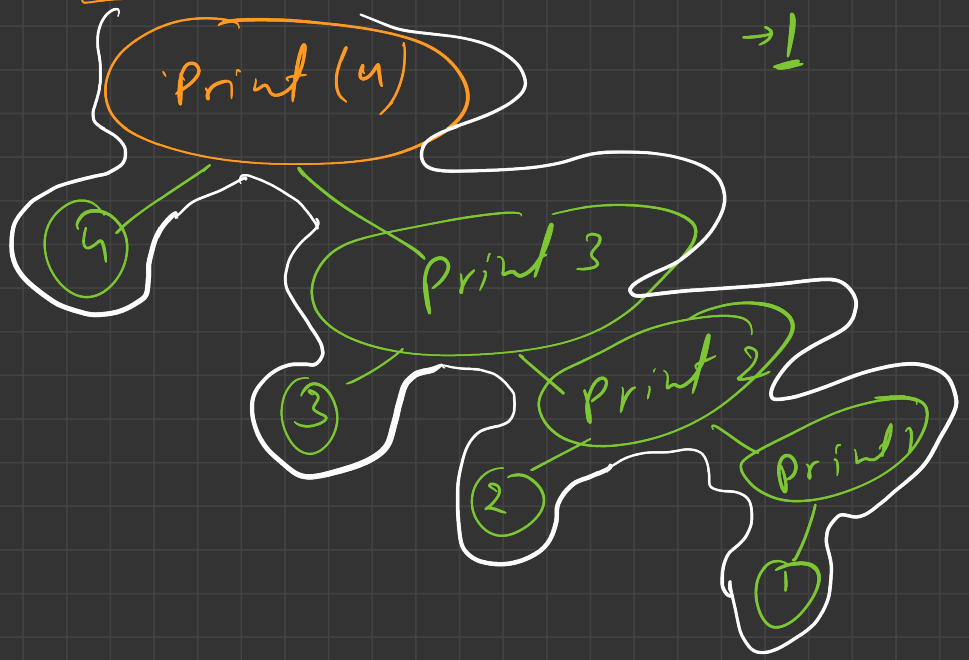
```
void Print(int n){
    if(n==1){
        cout<<1;
        return;
    }
    cout<<n;
    Print(n-1);
}
```

4
3
2
1

→ Print(1) = 1
Print(2) = 2, Print(1)
Print(3) = 3, Print(2)

→ Print(n) = n, Print(n-1)

6
5
4
3
2
→1



n to 1 even num

↳ even nu

Print(2) = 2

Print(4) = 4, Print(2)

Print(6) = 6, Print(4)

Print(n) = n, Print(n-2)

```
void print(int n) {
    if (n==2){
        cout<<2;
        return;
    }
    cout<<n;
    print(n-2);
}
```

10
8
6
4
⇒ 2