Recursion

1

```cpp
#include <bits/stdc++.h>
using namespace std;
int fact(int n){
    //base case
    if(n==0){
        return 1;
    }
    //rec case
    int ans = n*fact(n-1);
    return ans;
}
int main(){
    int n;
    cin >> n;
    cout << fact(n) << endl;
}
```

2

```cpp
#include <bits/stdc++.h>
using namespace std;
int fib(int n){
    if(n==0 or n==1){
        return n;
    }
    int f1 = fib(n-1);
    int f2 = fib(n-2);
    return f1 + f2;
}
int main(){
    int n;
    cin >> n;
    cout << fib(n) << endl;

}
```

3

```cpp
//write a function to check whether array is sorted or not
#include <bits/stdc++.h>
using namespace std;
bool check(int arr[],int n){
    if(n==0 or n==1){
        return true;
    }
```

```
    if(arr[0] < arr[1] and check(arr+1,n-1)){
        return true;
    }
    return false;
}
int main(){
    int n;
    cin >> n;
    int arr[n];
    for (int i = 0; i < n ; ++i)
    {
        cin >> arr[i];
    }
    bool result = check(arr,n);
    cout << result;
}
```

4

```
#include <bits/stdc++.h>
using namespace std;
void method_increasing(int n){
    if(n==0){
        return;
    }
    method_increasing(n-1);
    cout<<n;
}
void method_decreasing(int n){
    if(n==0){
        return;
    }
    cout<<n;
    method_decreasing(n-1);
}
int main(){
    int n;
    cin >> n;
    method_increasing(n);
    cout << endl;
    method_decreasing(n);

}
```

5

```
#include <bits/stdc++.h>
```

```cpp
using namespace std;
int first_element(int arr[],int n , int s){
    if(n==0){
        return -1;
    }
    if(arr[0]==s){
        return 0;
    }
    int subIndex = first_element(arr+1,n-1,s);
    if(subIndex!=-1){
        return subIndex+1;
    }
    return -1;
}
int main(){
    int n;
    cin >> n;
    int arr[n];
    for (int i = 0; i < n ; ++i)
    {
        cin >> arr[i];
    }
    int s;
    cin >> s;
    cout<<first_element(arr,n,s);
}
```

6

```cpp
//last occurence of the element
#include <bits/stdc++.h>
using namespace std;
int last_element(int arr[],int n , int s){
    if(n==0){
        return -1;
    }
    //rec case
    int subIndex = last_element(arr +1,n-1,s);
    if(subIndex==-1){
        if(arr[0]==s){
            return 0;
        }
        else{
            return -1;
        }
    }
    else{
        return subIndex + 1;
```

```
    }
}
int main(){
    int n;
    cin >> n;
    int arr[n];
    for (int i = 0; i < n ; ++i)
    {
        cin >> arr[i];
    }
    int s;
    cin >> s;
    cout<<last_element(arr,n,s);
}
```

7

```
//power function optimized
// a^n =(a^n/2)^2 and if it was odd then a^n =a(a^n/2)^2
#include <bits/stdc++.h>
using namespace std;
int power(int a , int n){
    if(n==0){
        return 1;
    }
    int subPro = power(a,n/2);
    int subProSq = subPro*subPro;
    if(n&1){
        return a*subProSq;
    }
    return subProSq;
}
int main(){
    int a,n;
    cin >> a >> n;
    cout << power(a,n) << endl;
    return 0;
}
```

8

```
//bubble sort
#include <bits/stdc++.h>
using namespace std;
void bubble_sort(int a[],int n){
    //base case
    if(n==1){
        return;
```

```
    }
    //rec case
    for (int j = 0; j < n-1; ++j)
    {
        if(a[j] > a[j+1]){
            swap(a[j],a[j+1]);
        }
    }
    bubble_sort(a,n-1);
}
int main(){
    int arr[] = {3,4,6,3,22,5,22,46,7,54};
    int n = sizeof(arr)/sizeof(int);
    cout << n << endl;
    bubble_sort(arr,n);
    for(int x :arr){
        cout << x <<" ";
    }
}
```

9

```
//print 2021 in two zero two one
#include <bits/stdc++.h>
using namespace std;
string spell[] =
{"zero","one","two","three","four","five","six","seven","eight","nine"};
void toChar(int n){
    if(n==0){
        return;
    }
    int last_digit = n%10;
    toChar(n/10);
    cout << spell[last_digit] << " ";

}
int main(){
    int n;
    cin >> n;
    toChar(n);
    return 0;
}
```