RECURSION

1. Recursion for printing numbers

#include<bits/stdc++.h>

using namespace std;

int cnt = 0;

void print(){

// Base Condition.

if(cnt == 3) return;

cout<<cnt<<endl;

// Count Incremented

cnt++;

print();

}

int main(){

print();

return 0;

}

Output

0

1

2

1. Print name N times

#include<bits/stdc++.h>

using namespace std;

void func(int i, int n){

// Base Condition.

if(i>n) return;

cout<<"Raj"<<endl;

// Function call to print till i increments.

func(i+1,n);

}

int main(){

// Here, let’s take the value of n to be 4.

int n = 4;

func(1,n);

return 0;

}

Output

Raj

Raj

Raj

Raj

1. Print from 1 to N

#include<bits/stdc++.h>

using namespace std;

void func(int i, int n){

// Base Condition.

if(i>n) return;

cout<<i<<endl;

// Function call to print i till i increments to n.

func(i+1,n);

}

int main(){

// Here, let’s take the value of n to be 4.

int n = 4;

func(1,n);

return 0;

}

Output

1

2

3

4

1. Print N to 1

#include<bits/stdc++.h>

using namespace std;

void func(int i, int n){

// Base Condition.

if(i<1) return;

cout<<i<<endl;

// Function call to print i till i decrements to 1.

func(i-1,n);

}

int main(){

// Here, let’s take the value of n to be 4.

int n = 4;

func(n,n);

return 0;

}

Output

4

3

2

1

1. Sum of N numbers

#include<bits/stdc++.h>

using namespace std;

void solve(int n) {

int sum = 0;

for (int i = 1; i <= n; i++) {

sum += i;

}

cout<<"The sum of the first "<<n<<" numbers is: "<<sum<<endl;

}

int main() {

solve(5);

solve(6);

}

Output:

The sum of the first 5 numbers is: 15

The sum of the first 6 numbers is: 21

1. Factorial of a number

#include<bits/stdc++.h>

using namespace std;

int factorial(int n){

// Base Condition.

if(n == 0)

{

return 1;

}

// Problem broken down into 2 parts and then combined.

return n \* factorial(n-1);

}

int main(){

// Here, let’s take the value of n to be 3.

int n = 3;

cout<<factorial(n)<<endl;

return 0;

}

Output

6

1. Reverse an array

#include <iostream>

using namespace std;

//Function to print array

void printArray(int arr[], int n) {

cout << "The reversed array is:- " << endl;

for (int i = 0; i < n; i++) {

cout << arr[i] << " ";

}

}

//Function to reverse array using recursion

void reverseArray(int arr[], int start, int end) {

if (start < end) {

swap(arr[start], arr[end]);

reverseArray(arr, start + 1, end - 1);

}

}

int main() {

int n = 5;

int arr[] = { 5, 4, 3, 2, 1 };

reverseArray(arr, 0, n - 1);

printArray(arr, n);

return 0;

}

Output:

The reversed array is:-

1 2 3 4 5

1. String is palindrome or not

#include <iostream>

using namespace std;

bool palindrome(int i, string& s){

// Base Condition

// If i exceeds half of the string means all the elements

// are compared, we return true.

if(i>=s.length()/2) return true;

// If the start is not equal to the end, not the palindrome.

if(s[i]!=s[s.length()-i-1]) return false;

// If both characters are the same, increment i and check start+1 and end-1.

return palindrome(i+1,s);

}

int main() {

// Example string.

string s = "madam";

cout<<palindrome(0,s);

cout<<endl;

return 0;

}

Output:

True

1. Fibonacci series upto the nth term

#include<bits/stdc++.h>

using namespace std;

int main() {

int n = 5;

if (n == 0) {

cout << 0;

} else if (n == 1) {

cout << 0 << " " << 1 << "\n";

} else {

int fib[n + 1];

fib[0] = 0;

fib[1] = 1;

for (int i = 2; i <= n; i++) {

fib[i] = fib[i - 1] + fib[i - 2];

}

cout<<"The Fibonacci Series up to "<<n<<"th term:"<<endl;

for (int i = 0; i <= n; i++) {

cout << fib[i] << " ";

}

}

}

Output:

The Fibonacci Series up to 5th term:

0 1 1 2 3 5

1. Print Fibonacci number for given number #Asli Recursion

#include<bits/stdc++.h>

using namespace std;

int fibonacci(int N){

// Base Condition.

if(N <= 1)

{

return N;

}

// Problem broken down into 2 functional calls

// and their results combined and returned.

int last = fibonacci(N-1);

int slast = fibonacci(N-2);

return last + slast;

}

int main(){

// Here, let’s take the value of N to be 4.

int N = 4;

cout<<fibonacci(N)<<endl;

return 0;

}

Output

3