Name-Abhisek Roy

Department- B.Tech in Information Technology

Year-3rd

Semester-5th

Roll No-10000219062

Registration No-042888

Paper code :PCCIT591

1. **Write a recursive function to compute Max and Min of n elements.**

**Algorithm:**

Get a array of numbers

Define a recursive function:

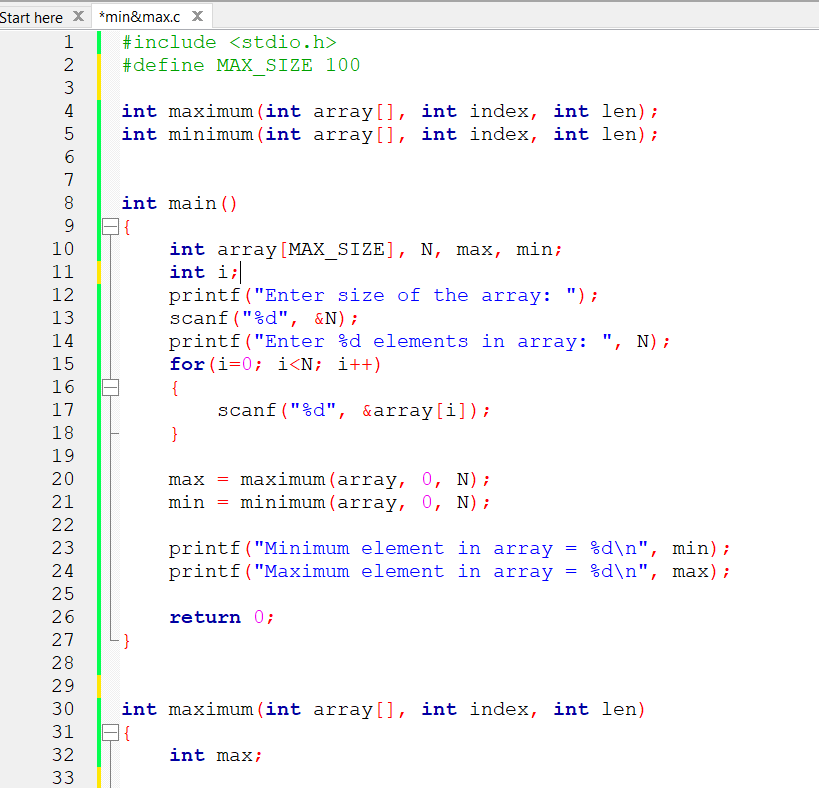
Base Case: if n==1, return arr[0]

Recursive Case: return max(arr[n-1], findMax(arr,n-1))

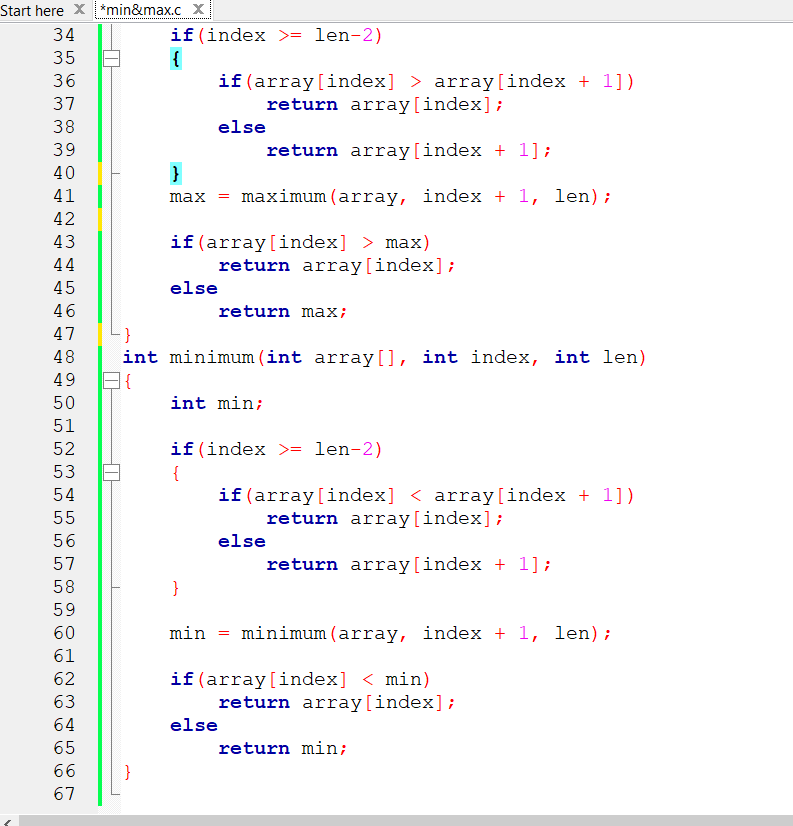
For minimum:

Recursive Case: return min(arr[n-1], findMin(arr,n-1)

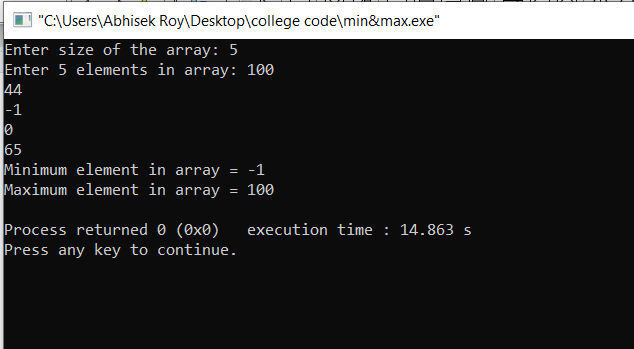
**Code:**



\*-+



**Output**



1. **write a recursive function to find the binary equivalent of a positive integer number.**

**Algorithm:**

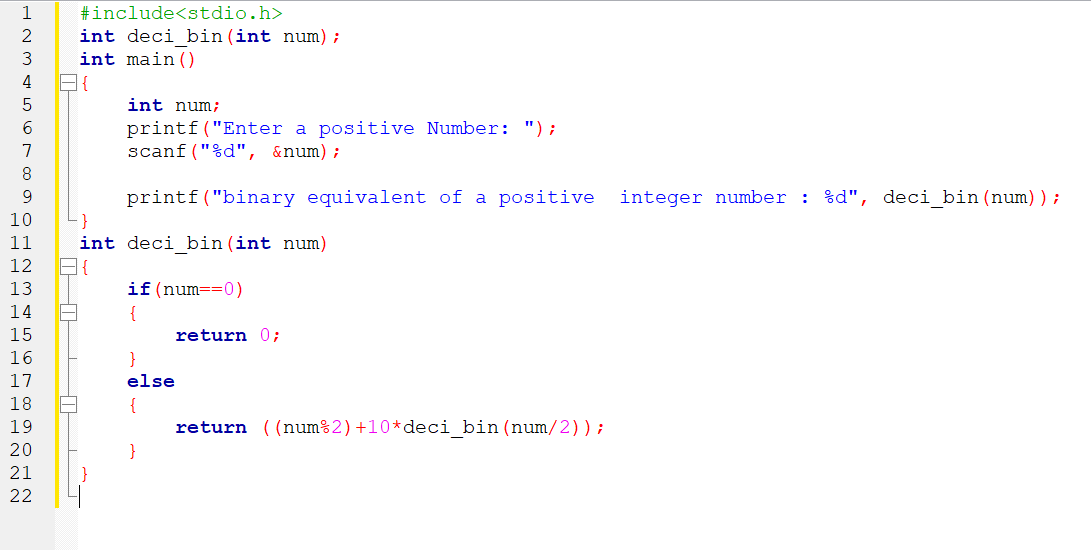
int deci\_bin(int num)

if(num==0)

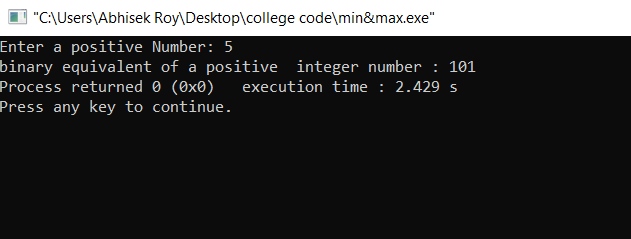
return 0;

else

return ((num%2)+10\*deci\_bin(num/2));



**Output**



1. **Write a recursive function that adds up the numbers present within an array.**

**Algorithm :**

int sum( int arr[], int n )

{

int sum = 0;

//base case:

if (n < 0) {

return sum;

} else{

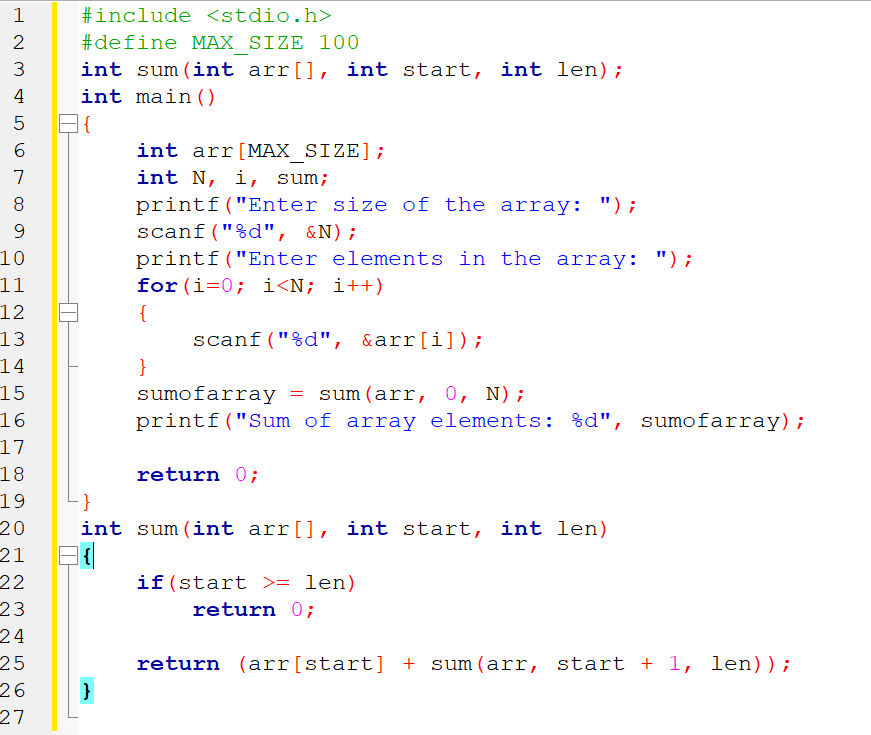
sum = sum + arr[n];

}

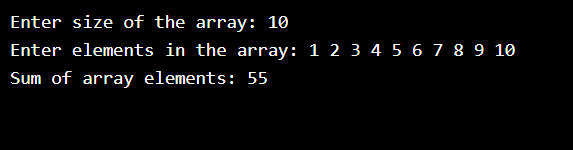
sum(arr,n-1);

}

**code**



**Output**



1. **Write a recursive function to compute the binomial coefficients.**

**Algorithm:**

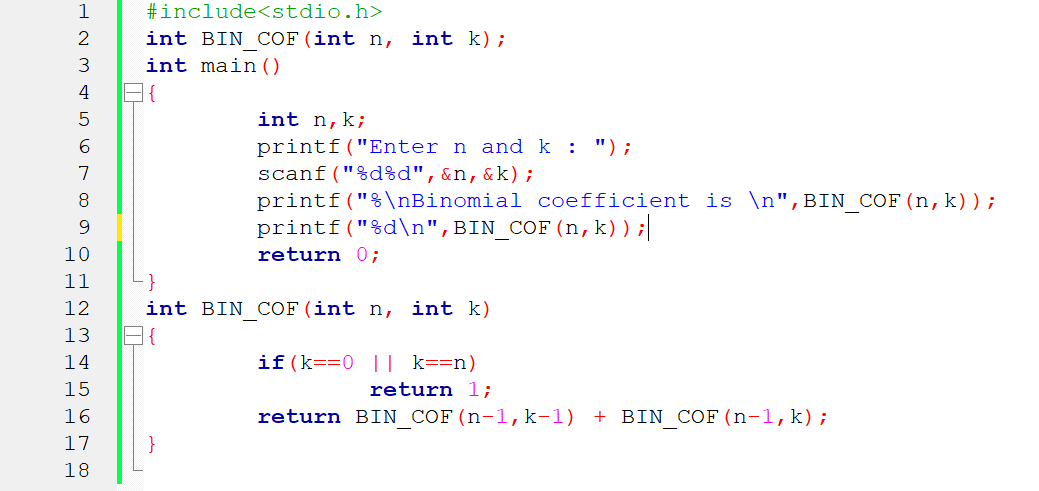
int BIN\_COF(int n, int k)

if (k==0 || k==n)

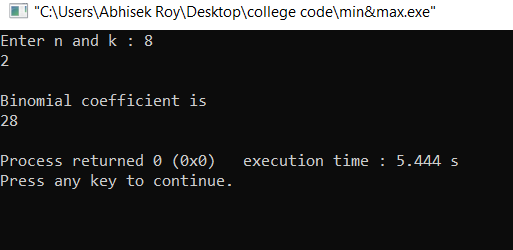
return 1;

return BIN\_COF(n-1, k-1) + BIN\_COF(n-1, k);

**code :**



**Output**



1. **Write a program that reads number and outputs all the possible sums that can be formed by digits of the number.**

**Algorithm:**

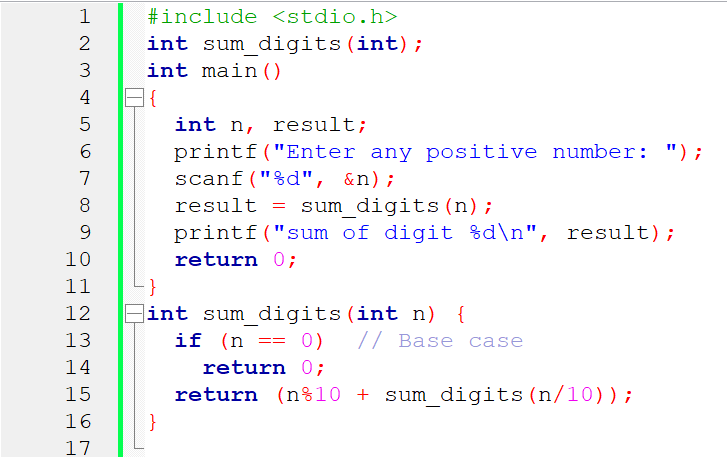
int sum\_digits(int n)

if (n == 0)

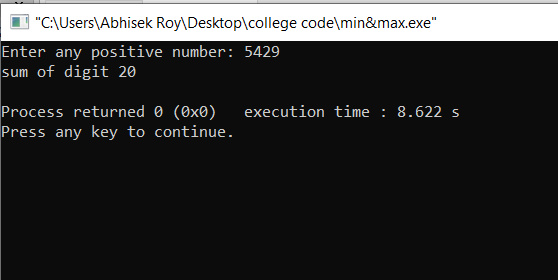
return 0;

return (n%10 + sum\_digits(n/10));

**Code:**



**Output**



1. **Consider a binary tree and write recursive functions to compute the no. of nodes present in the tree.**

**Algorithm :**

if(root != NULL)

{

countnodes(root->left);

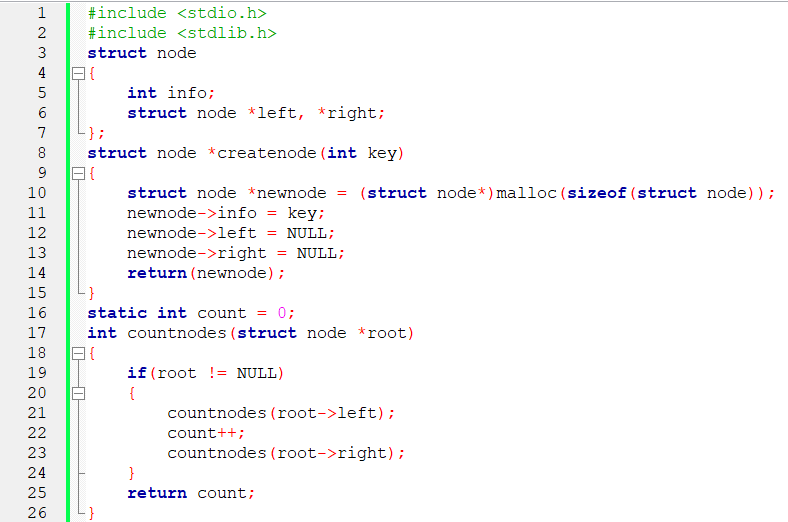
count++;

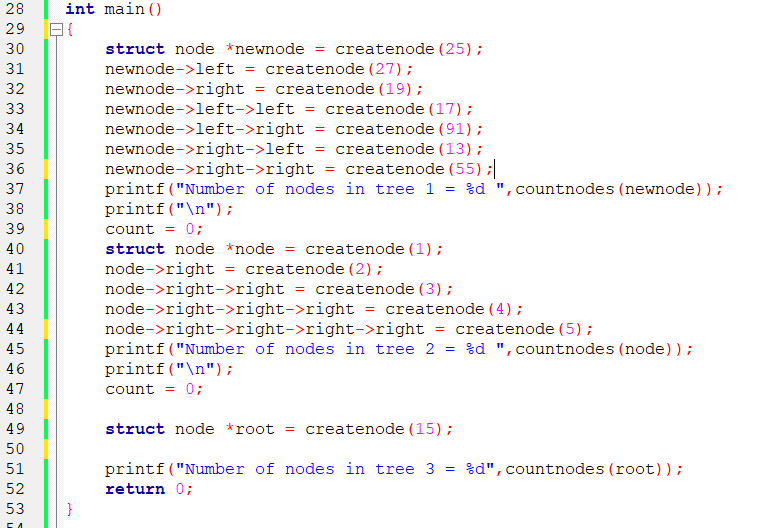
countnodes(root->right);

}

return count;

**Code:**





**Output**

