**CPP LAB ASSIGNMENT**

**Ques 1. write a program to find the given number is even or odd**

#include<iostream>

using namespace std;

int main(){

    int n;

    cout<<"enter an integer:";

    cin>>n;

    if(n%2==0)

    cout<<n<<" is even.";

    else

    cout<<n<<" is odd.";

    return 0;

}

**OUTPUT**

****

**Ques 2. write a program to find whether the given number is prime or composite**

#include<iostream>

using namespace std;

int main(){

    int num,i,count=0;

    cout<<" enter the number:";

    cin>>num;

    for(i=2;i<num;i++)

    {

        if(num%i==0)

         count++;

    }

if(count==1)

{

    cout<<"\n prime number";

}

else

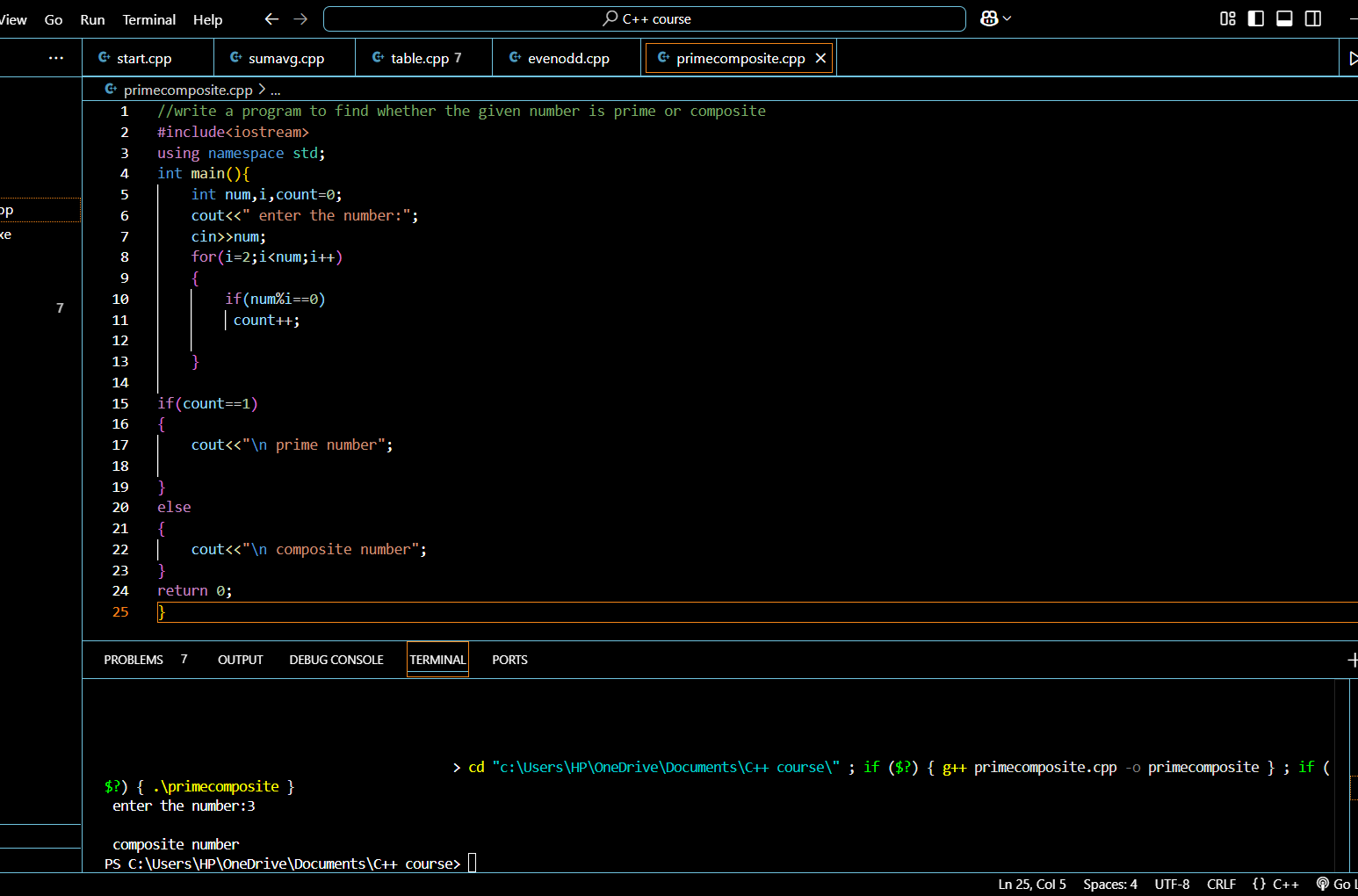
{

    cout<<"\n composite number";

}

return 0;

}



**Ques 3. write a program to print table of a given number up to n number/range**

#include <iostream>

using namespace std;

int main() {

    int n, range;

    cout << "Enter an integer: ";

    cin >> n;

    cout << "Enter range: ";

    cin >> range;

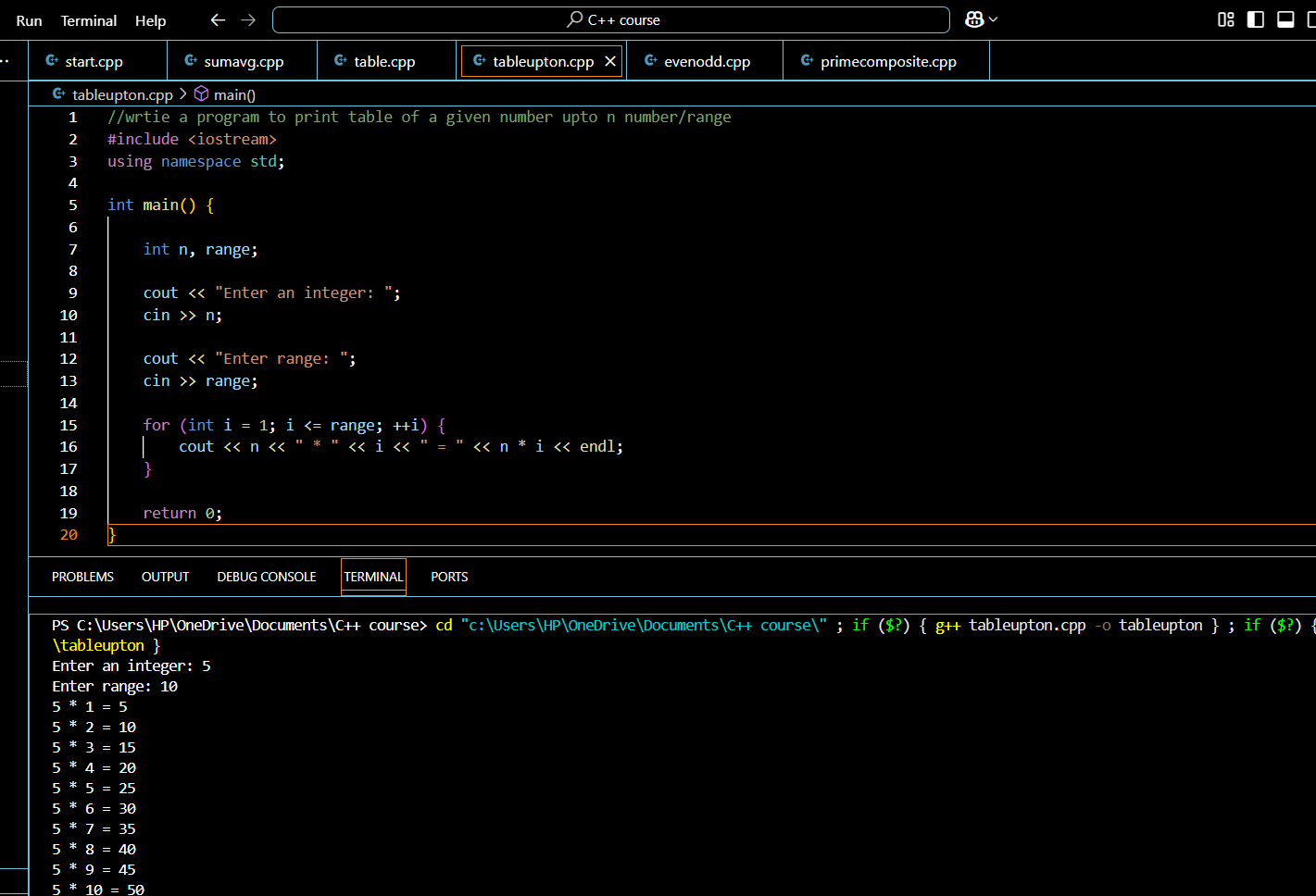
    for (int i = 1; i <= range; ++i) {

        cout << n << " \* " << i << " = " << n \* i << endl;

    }

    return 0;

}



**Ques 4. 1) write a program to find greater of the two numbers**

#include <iostream>

using namespace std;

int main()

{

    int num1, num2;

    cout<<"Enter first number:";

    cin>>num1;

    cout<<"Enter second number:";

    cin>>num2;

    if(num1>num2)

    {

    cout<<"First number "<<num1<<" is the largest";

    }

    else

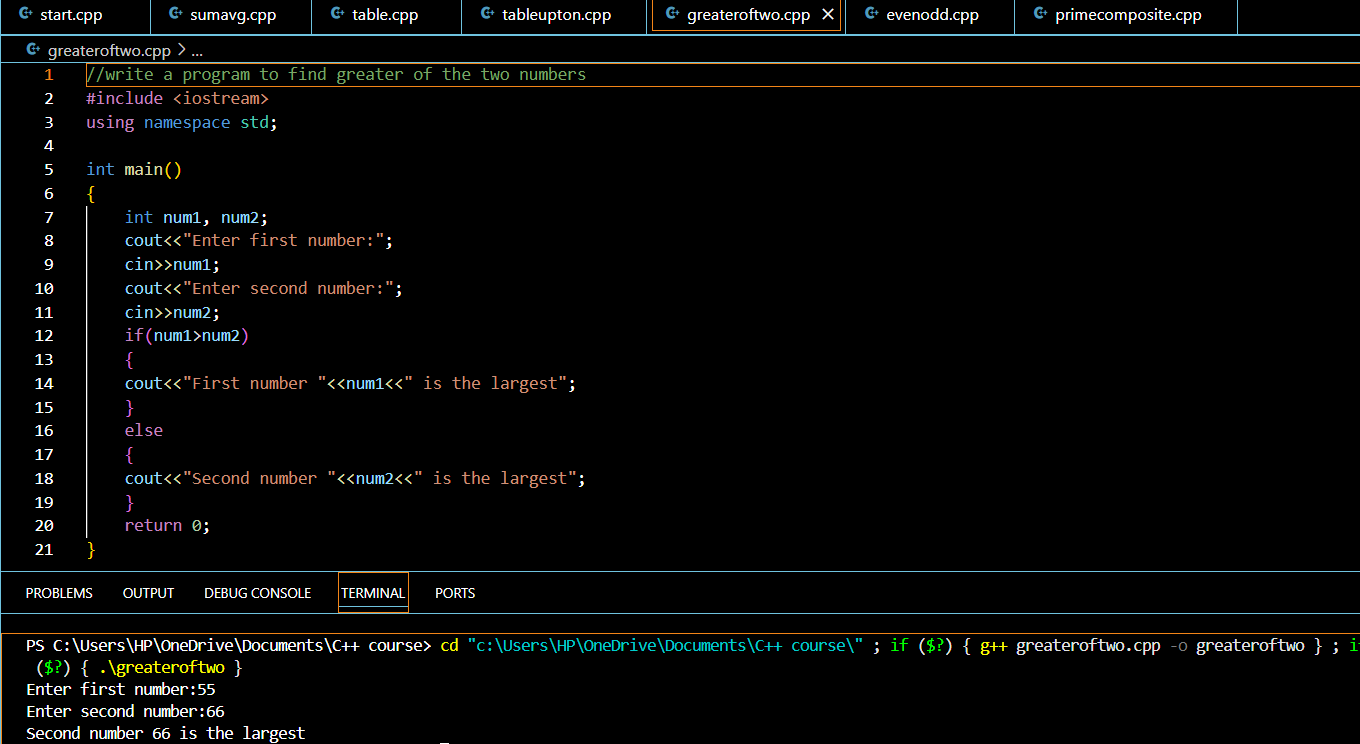
    {

    cout<<"Second number "<<num2<<" is the largest";

    }

    return 0;

}



**2) wrtie a program to find out greatest of the three numbers**

#include<iostream>

using namespace std;

int main(){

    int a,b,c;

    cout<<"enter three numbers";

    cin>>a>>b>>c;

    if(a>b){

        if(a>c){

            cout<<a<<" is the greatest";

        }

        else{

            cout<<c<<" is the greatest";

        }

          }

          else {

            if(b>c){

                cout<<b<<" is the greatest";

            }

            else{

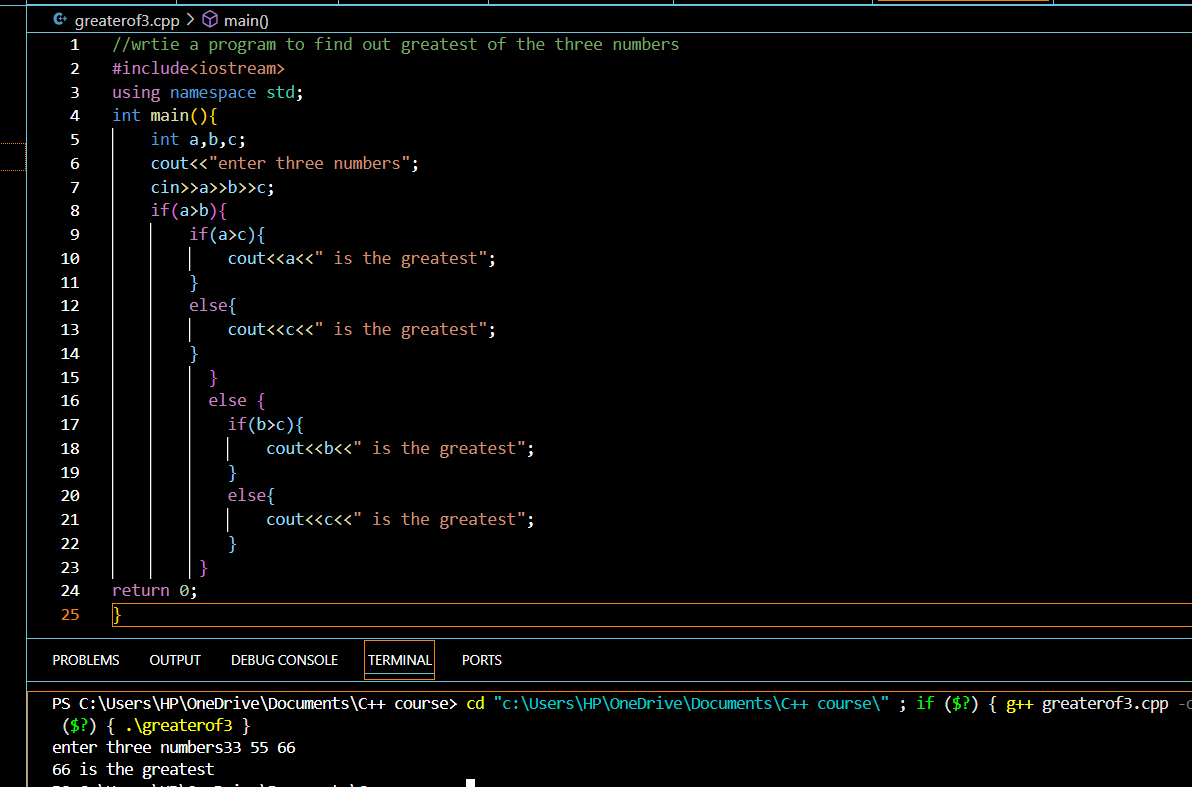
                cout<<c<<" is the greatest";

            }

         }

return 0;

}



**Ques 5. write a program to find sum of n natural numbers**

#include <iostream>

using namespace std;

int main() {

    int n, sum = 0;

    cout << "Enter a positive integer: ";

    cin >> n;

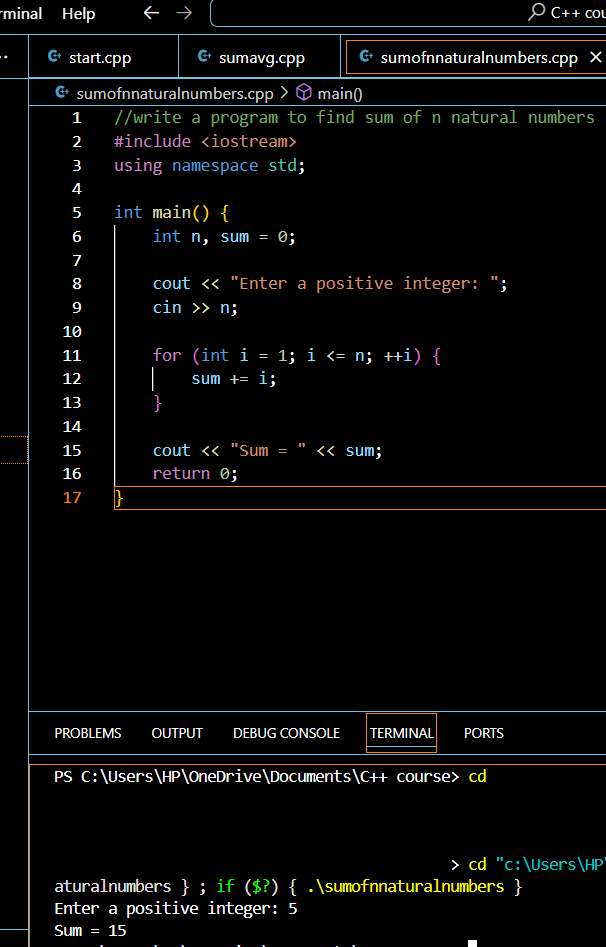
    for (int i = 1; i <= n; ++i) {

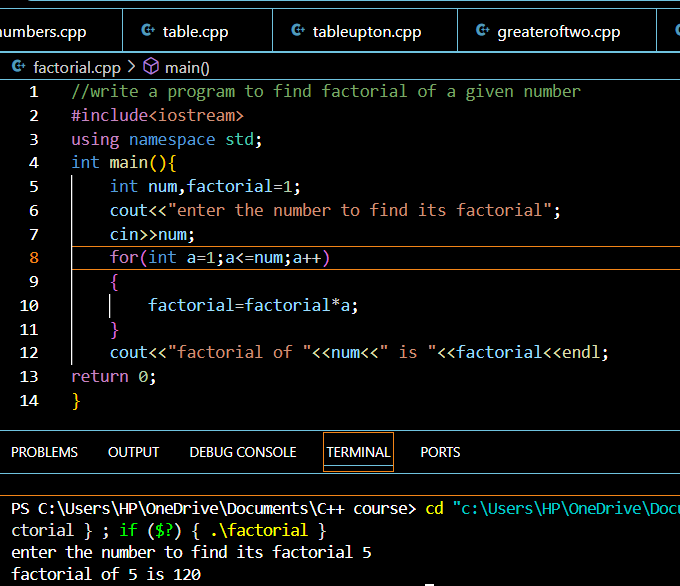
        sum += i;

    }

    cout << "Sum = " << sum;

    return 0;

}

Ques6.

**Ques 7. write a program to find sum of digits of n digit number**

#include<iostream>

using namespace std;

int main(){

    int num,sum=0;

    cout<<"enter the number:";

    cin>>num;

    while(num!=0){

        sum+=num%10;

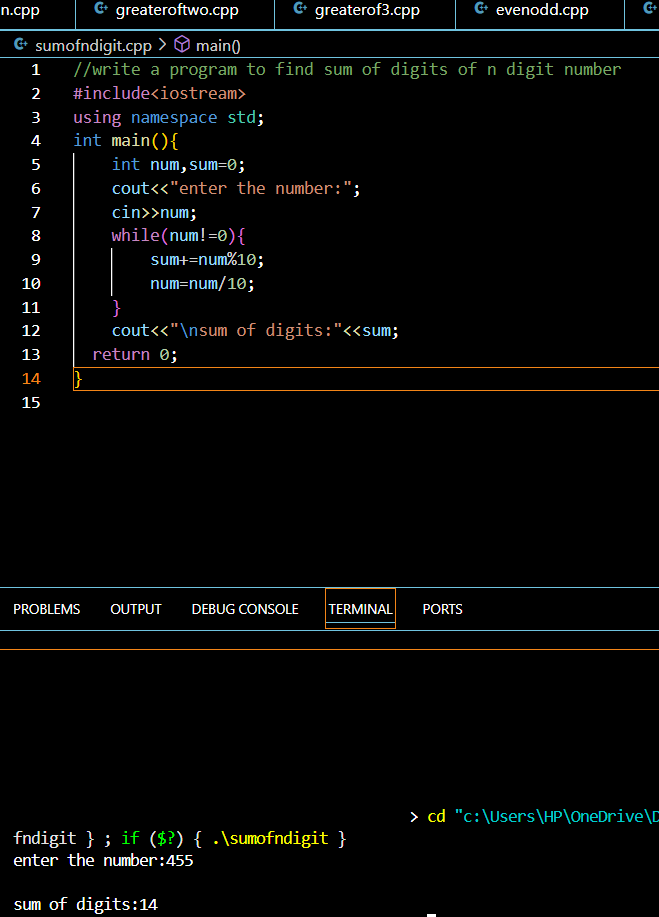
        num=num/10;

    }

    cout<<"\nsum of digits:"<<sum;

  return 0;

}



Q**ues 8. write a program to find reverse of a number**

#include <iostream>

using namespace std;

int main() {

  int n, reversed\_number = 0, remainder;

  cout << "Enter an integer: ";

  cin >> n;

  while(n != 0) {

    remainder = n % 10;

    reversed\_number = reversed\_number \* 10 + remainder;

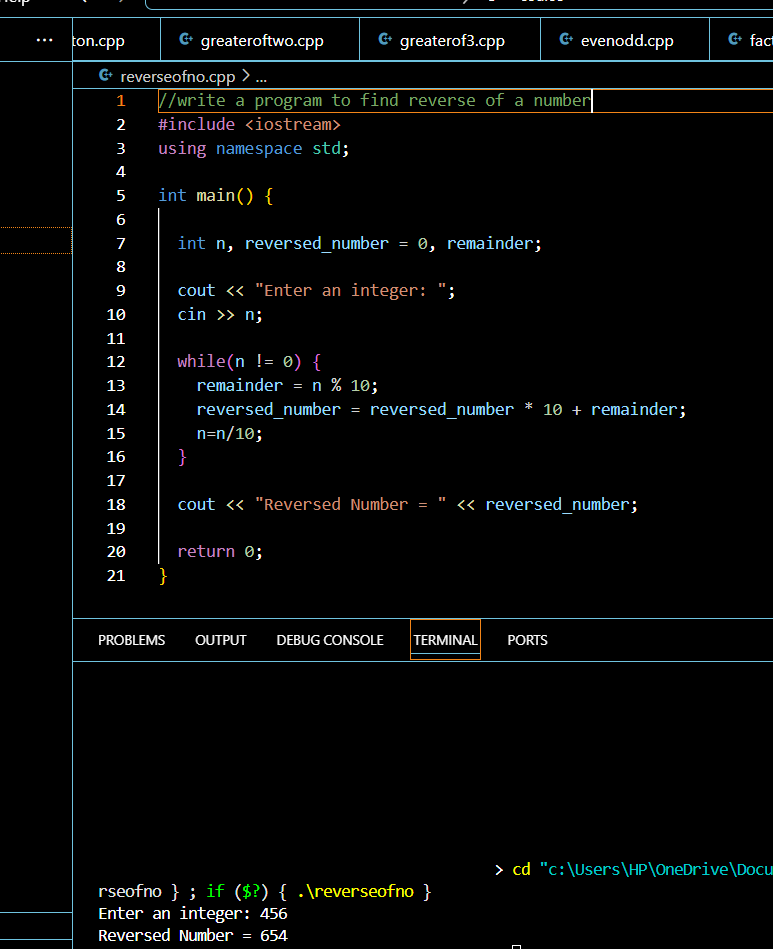
    n=n/10;

  }

  cout << "Reversed Number = " << reversed\_number;

  return 0;

}



**Ques 9. write a program to check whether a number is palindrome or not**

#include <iostream>

using namespace std;

int main()

{

     int n, num, digit, rev = 0;

     cout << "Enter a positive number: ";

     cin >> num;

     n = num;

     do

     {

         digit = num % 10;

         rev = (rev \* 10) + digit;

         num = num / 10;

     } while (num != 0);

     cout << " The reverse of the number is: " << rev << endl;

     if (n == rev and n > 0)  // Negative numbers are not palindromic

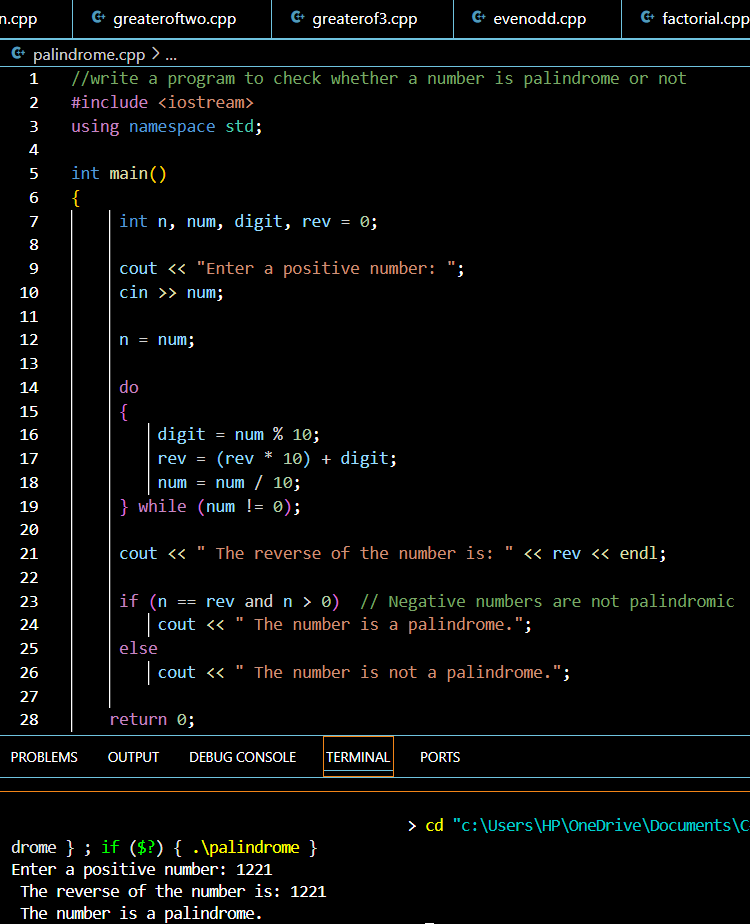
         cout << " The number is a palindrome.";

     else

         cout << " The number is not a palindrome.";

    return 0;

}



**Ques 10.Program to print Fibonacci series up to n numbers.**

#include <iostream>

using namespace std;

int main() {

    int n;

    cout << "Enter the number of terms for the Fibonacci series: ";

    cin >> n;

    int n1 = 0, n2 = 1, next;

    cout << "Fibonacci Series: ";

    for (int i = 1; i <= n; i++) {

        if (i == 1) {

            cout << n1 << " ";

            continue;

        }

        if (i == 2) {

            cout << n2 << " ";

            continue;

        }

        next = n1 + n2;

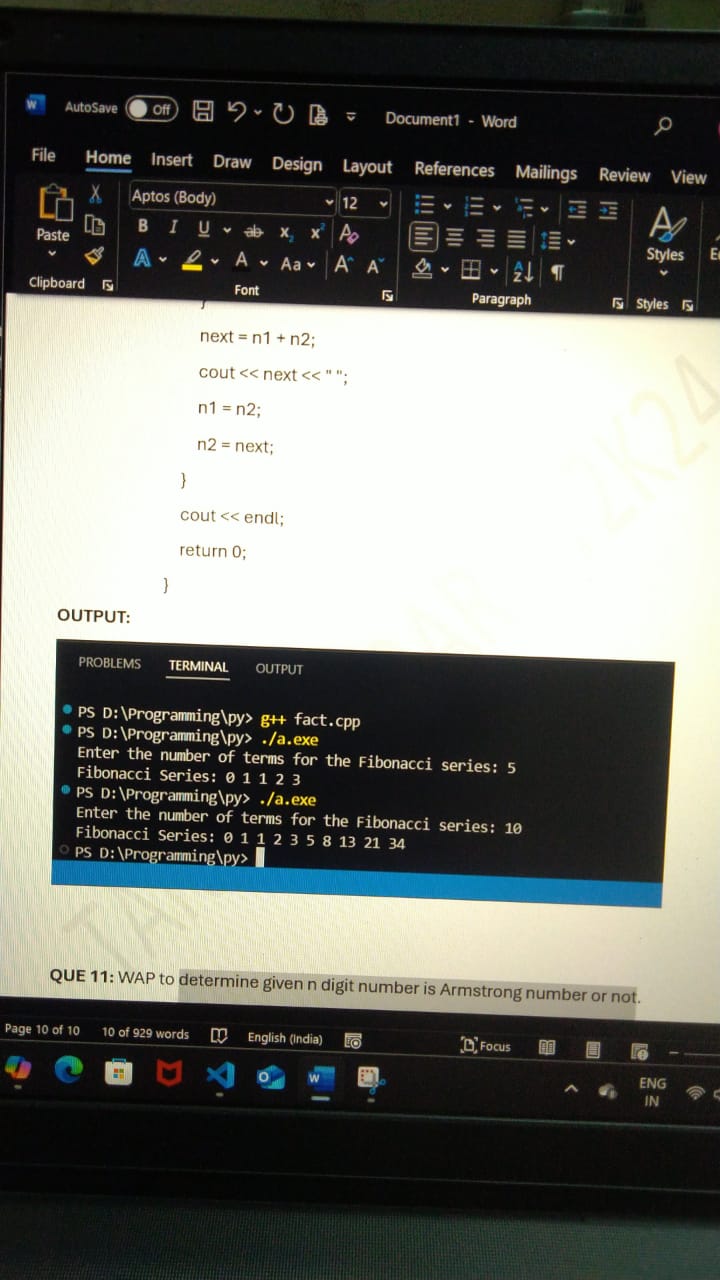
        cout << next << " ";

        n1 = n2;

        n2 = next;

    } cout << endl;

    return 0; }

}

**Ques 11. Determine n digit number is Armstrong or not.**

#include <iostream>

using namespace std;

int main() {

    int number, original, sum = 0, n = 0;

    cout << "Enter the number: ";

    cin >> number;

    original = number; // Store the original number

    // Count the number of digits

    int temp = number;

    while (temp != 0) {

        temp /= 10;

        n++;

    }

    // Calculate the sum of the n-th power of each digit without using the math library

    temp = number;

    for (; temp != 0; temp /= 10) {

        int digit = temp % 10; // Extract the last digit

        int power = 1; // To calculate digit raised to the power n

        for (int i = 0; i < n; i++) {

            power \*= digit;

        }

        sum += power; // Add the n-th power of the digit to the sum

    }

    // Check if the original number equals the sum

    if (original == sum) {

        cout << "The number is an Armstrong number." << endl;

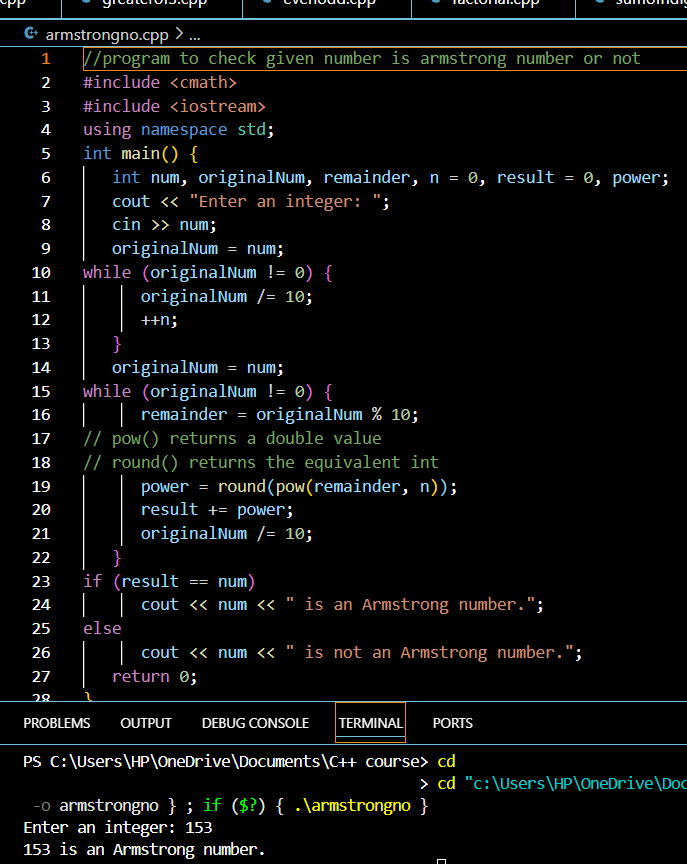
    } else {

        cout << "The number is not an Armstrong number." << endl;

    }

    return 0;

}



**Ques 12 . write a program to print even number between 100 and 200**

#include <iostream>

using namespace std;

int main()

{

    int start = 100, end = 200;

    cout << "The even numbers between " << start << " and " << end

            << " is "<<endl;

    for (int i = start; i <end; i++)

        if (i % 2 == 0)

            cout << i << "\t";

    return 0;

}



**Ques 13. Write a program to print first 50 prime numbers**

#include<iostream>

using namespace std;

int main(){

int i, chk=0, j;

    cout<<"Prime Numbers Between 1 to 50 are:\n";

    for(i=1; i<=50; i++)

    {

        for(j=2; j<i; j++)

        {

           if(i%j==0)

           {

               chk++;

               break;

           }

        }

        if(chk==0 && i!=1)

            cout<<i<<endl;

        chk = 0;

    }

    cout<<endl;

    return 0;

}



**Question 14. Write program to print all 4 digit Armstrong number.**

#include <iostream>

using namespace std;

int main()

{

int counter,sum=0;

cout << "4-digit Armstrong numbers are:" << endl;

for (int num = 1000; num < 10000; num++) {

counter = num;

sum=0;

while (counter != 0) {

int digit = counter % 10;

int power = 1;

for (int i = 0; i < 4; i++) {

power \*= digit;

}

sum += power;

counter /= 10;

}

if (sum == num) {

cout << num << " ";

}

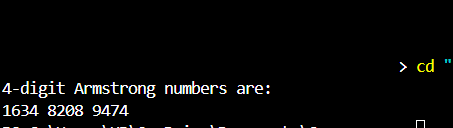
}

cout << endl;

return 0;

}

**OUTPUT:**



**Ques 15. 1) right half pyramid pattern program**

#include<iostream>

using namespace std;

int main(){

int rows=5;

// first loop for printing rows

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

    // second loop for printing character in each rows

    for (int j = 0; j <= i; j++){

        cout<<"\* ";

    }

    cout<<"\n";

}

return 0;

}

**2)program to print inverted right half pyramid**

#include<iostream>

using namespace std;

int main(){

   int rows = 5;

    // first loop to print all rows

    for (int i = 0; i < rows; i++) {

        // first inner loop to print the \* in each row

        for (int j = 0; j < rows - i; j++) {

            cout<<"\* ";

        }

        cout<<"\n";

}

return 0;

}

**3)**

#include <iostream>

using namespace std;

void main(){

for(int i = 1 ; i <= 3 ; i++){

for(int j = 5-i ; j > 0 ; j--){

cout << " ";

}

for(int k = 1 ; k <= (2\*i)- 1 ; k++){

cout << "\*";

}

cout << "\n";

}

}

**4)**

#include <iostream>

using namespace std;

int main() {

    for (int i = 1; i <= 4; i++) {   // Loop for 4 rows

        for (int j = 1; j <= i; j++) {  // Loop for printing the numbers in each row

            cout << i;  // Print the number i

        }

        cout << endl;

    }

    return 0;

}

**5) Pascal’s triangle**

#include<iostream>

using namespace std;

int fact(int x){

int f=1;

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

f\*=i;

}

return f;

}

int combi(int n,int r){

int c;

c= fact(n) /(fact(r) \* fact(n-r) );

return c;

}

int main() {

int n;

cout<<"Enter a number: ";

cin>>n;

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

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

cout<<" ";

}

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

cout<<combi(i,j)<<" ";

}

cout<<endl;

}

return 0;

}

**6)Floyd’s triangle**

#include <iostream>

using namespace std;

int main()

{

    int n, i,  c, a = 1;

    cout << "Enter the number of rows of Floyd's triangle to print: "; cin >> n;

    for (i = 1; i <= n; i++)

    {

        for (c = 1; c <= i; c++)

        {

            cout << a;

            a++;

        }

        cout << endl;

    }

    return 0;

}

**Q16. Using functions write following c++ programs.**

**1.To print palindrome numbers from range 500 to 1000**

#include <iostream>

using namespace std;

void Palindrome()

{

for (int i = 500; i <= 1000; i++)

{

int num = i;

int rev = 0;

while (num != 0)

{

int digit = num % 10;

rev = rev \* 10 + digit;

num /= 10;

}

if (rev == i)

{

cout << i << endl;

}

}

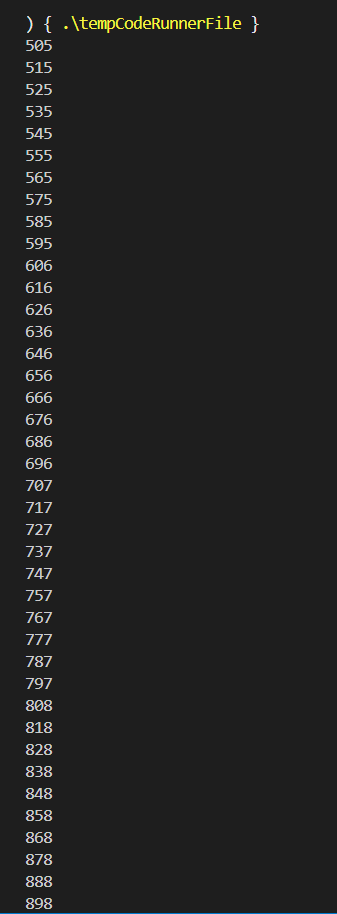
}

int main() {

Palindrome ();

return0;

}



**ii ) To print first 100 odd numbers.**

#include <iostream>

using namespace std;

void Odd()

{

int n;

for (n=1; n<=200;n++)

{

if (n%2!=0)

{

cout << “\n”<<n;

}

}

}

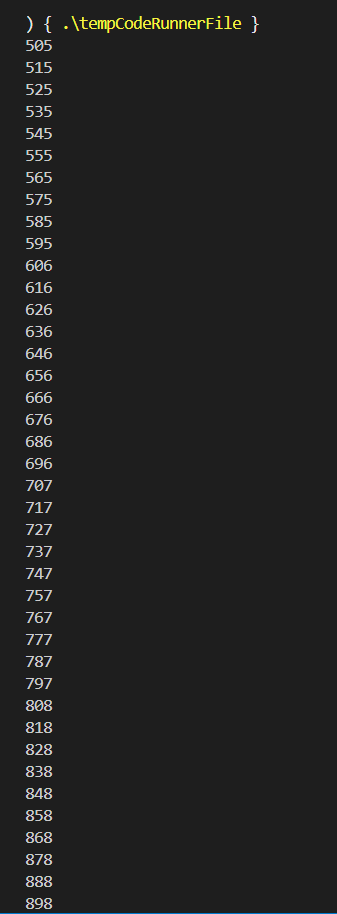
int main()

{

Odd();

return0;

}



**iii ) To find binary, octal, hexadecimal equivalent of a given decimal number.**

#include <iostream>

using namespace std;

void binary(int n)

{

int org\_num = n;

int factor = 1;

int bin = 0;

while (n != 0)

{

bin = bin + (n % 2) \* factor;

n = n / 2;

factor = factor \* 10;

}

cout << "The binary number for " << org\_num << " is " << bin << "\n";

}

void octal(int n)

{

int org\_num = n;

int factor = 1;

int oct = 0;

while (n != 0)

{

oct = oct + (n % 8) \* factor;

n = n / 8;

factor = factor \* 10;

}

cout << "The octal number for " << org\_num << " is " << oct << "\n";

}

void hexadecimal(int n)

{

int org\_num = n;

int factor = 1;

int hexa = 0;

while (n != 0)

{

hexa = hexa + (n % 16) \* factor;

n = n / 16;

factor = factor \* 10;

}

cout << "The hexadecimal number for " << org\_num << " is " << hexa << "\n";

}

int main()

{

int num;

cout << "Enter a number:";

cin >> num;

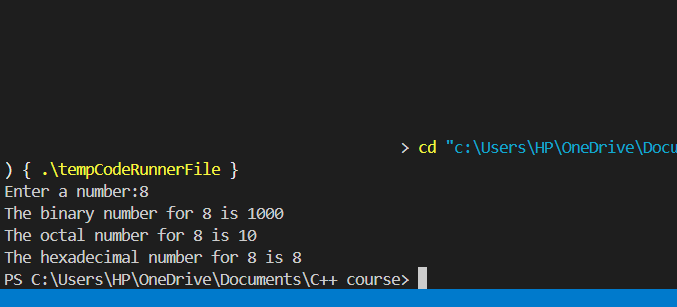
binary(num);

octal(num);

hexadecimal(num);

return 0;

}



**iv ) To find decimal equivalents for given binary, hexadecimal and octal numbers.**

#include <iostream>

#include <cmath>

using namespace std;

void bin(int n)

{

int org\_num = n;

int deci = 0;

int power = 0;

while (n != 0)

{

deci = deci + (n % 10) \* pow(2, power);

n = n / 10;

power++;

}

cout << "The decimal number for binary " << org\_num << " is " << deci << "\n";

}

void oct(int n)

{

int org\_num = n;

int deci = 0;

int power = 0;

while (n != 0)

{

deci = deci + (n % 10) \* pow(8, power);

n = n / 10;

power++;

}

cout << "The decimal number for octal " << org\_num << " is " << deci << "\n";

}

void hex(int n)

{

int org\_num = n;

int deci = 0;

int power = 0;

while (n != 0)

{

deci = deci + (n % 10) \* pow(16, power);

n = n / 10;

power++;

}

cout << "The decimal number for hexadecimal " << org\_num << " is " << deci << "\n";

}

int main()

{

int num, choice = 0;

cout << "Enter 1 if number is binary, 2 if it is octal and 3 if it is hexadecimal:";

cin >> choice;

cout << "Enter a number:";

cin >> num;

if (choice == 1)

{

bin(num);

}

else if (choice == 2)

{

oct(num);

}

else if (choice == 3)

{

hex(num);

}

else

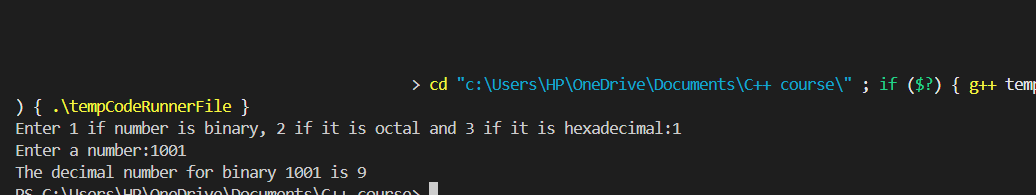
{

cout << "Invalid choice.";

}

return 0;

}



**v) To calculate geometric sum upto n terms.**

#include <iostream>

using namespace std;

double geometricSum(double a, double r, int n) {

double sum = 0;

double term = a;

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

sum += term;

term \*= r;

}

return sum;

}

int main() {

double a, r;

int n;

cout << "Enter the first term (a): ";

cin >> a;

cout << "Enter the common ratio (r): ";

cin >> r;

cout << "Enter the number of terms (n): ";

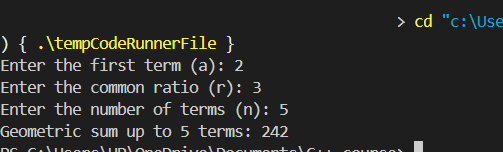
cin >> n;

double sum = geometricSum(a, r, n);

cout << "Geometric sum up to " << n << " terms: " << sum << endl;

return 0;

**}**

****

**Ques 17**

1. **Print binary for decimal using recursion**

#include <iostream>

using namespace std;

// Function to convert decimal to binary using recursion

void decimalToBinary(int n) {

    // Base case: if the number is 0 or 1, print the number

    if (n <= 1) {

        cout << n;

        return;

    }

    // Recursive case: divide the number by 2 and call the function recursively

    decimalToBinary(n / 2);

    // Print the remainder (either 0 or 1)

    cout << n % 2;

}

int main() {

    int num;

    cout << "Enter a decimal number: ";

    cin >> num;

    cout << "Binary representation: ";

    if (num == 0) {

        cout << 0; // Special case for zero

    } else {

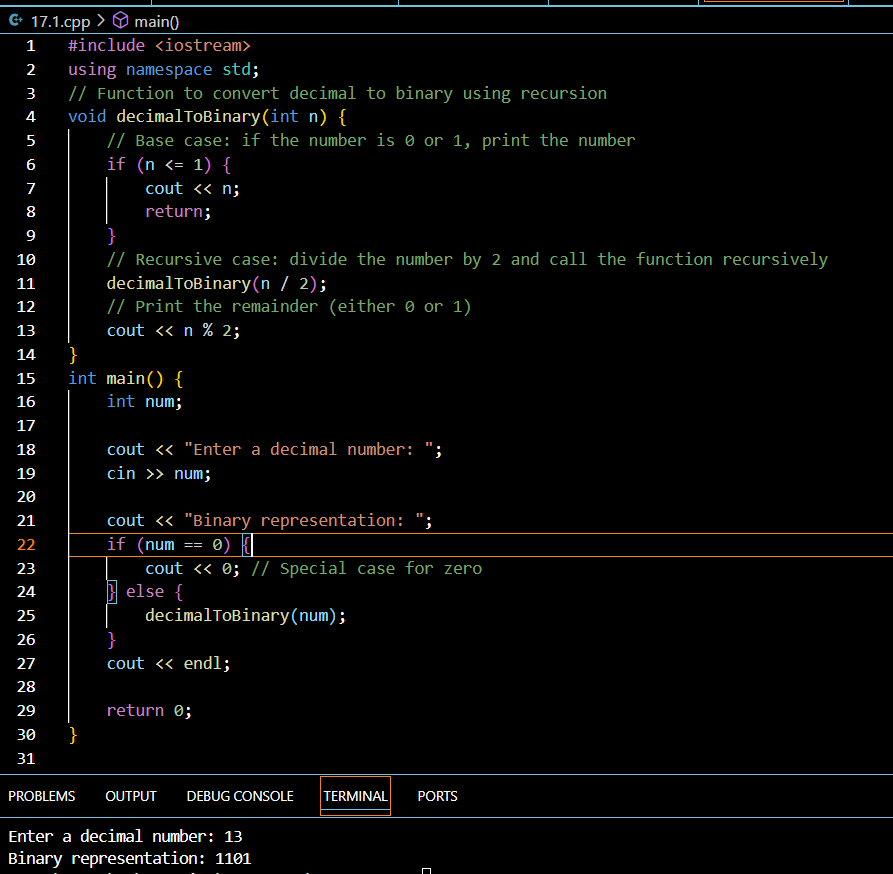
        decimalToBinary(num);

    }

    cout << endl;

    return 0;

**}**



1. **print octal for a decimal using recursion**

#include <iostream>

using namespace std;

// Function to convert decimal to octal using recursion

void decimalToOctal(int n) {

    // Base case: if the number is less than 8, print the number

    if (n < 8) {

        cout << n;

        return;

    }

 // Recursive case: divide the number by 8 and call the function recursively

    decimalToOctal(n / 8);

// Print the remainder (either a digit between 0 and 7)

    cout << n % 8;

}

int main() {

    int num;

    cout << "Enter a decimal number: ";

    cin >> num;

    cout << "Octal representation: ";

    if (num == 0) {

        cout << 0; // Special case for zero

    } else {

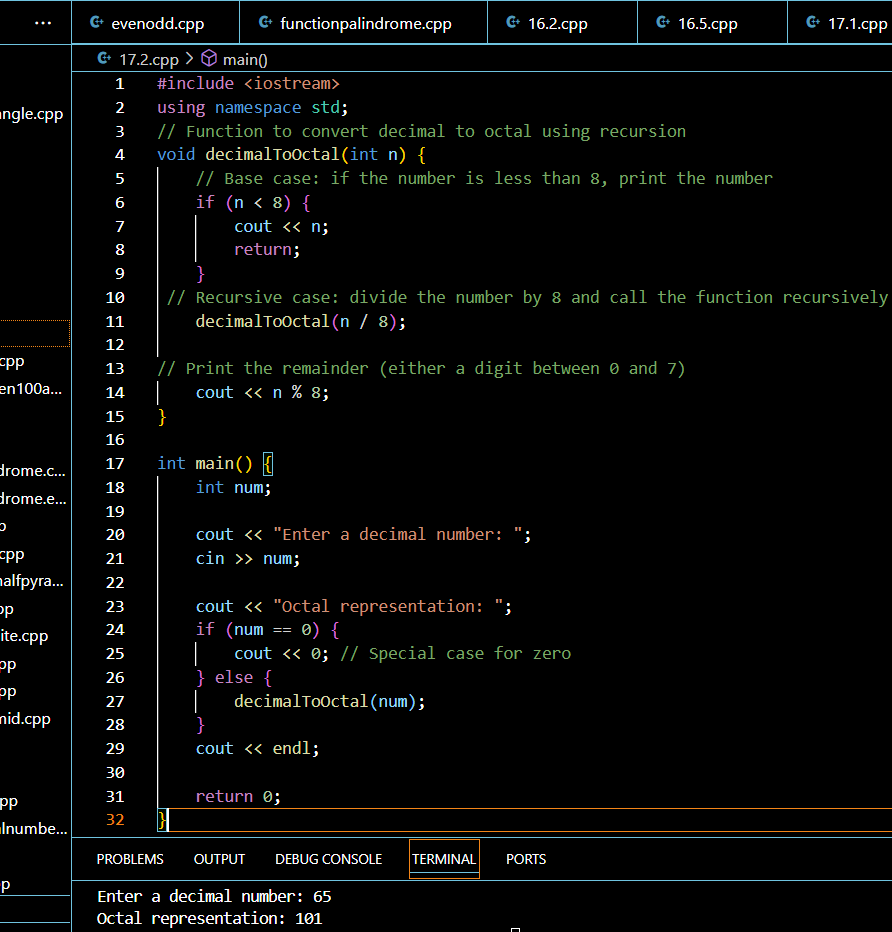
        decimalToOctal(num);

    }

    cout << endl;

    return 0;

**}**



1. **print factorial for a given range using recursion**

#include <iostream>

using namespace std;

// Function to calculate factorial recursively

long long factorial(int n) {

if (n <= 1) {

return 1; // Base case: factorial of 0 or 1 is 1

} else {

return n \* factorial(n - 1); // Recursive case

}

}

// Function to print factorials for numbers in the given range

void printFactorialsInRange(int start, int end) {

// Base case: if start is greater than end, stop the recursion

if (start > end) {

return;

}

// Print the factorial for the current number

cout << "Factorial of " << start << " is " << factorial(start) << endl;

// Recursively call the function for the next number

printFactorialsInRange(start + 1, end);

}

int main() {

int start, end;

cout << "Enter the range (start and end): ";

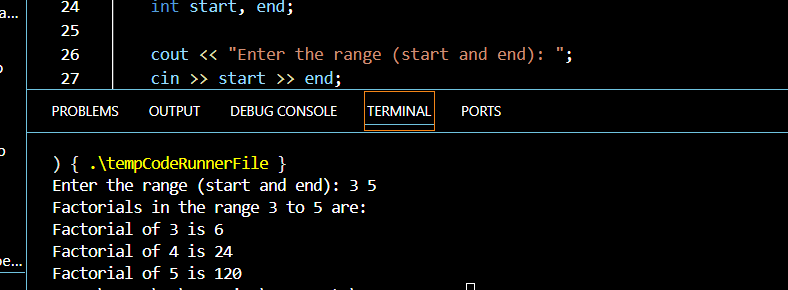
cin >> start >> end;

cout << "Factorials in the range " << start << " to " << end << " are:\n";

printFactorialsInRange(start, end);

return 0;

}



1. **print first n terms of Fibonacci series.**

#include <iostream>

using namespace std;

// Function to calculate Fibonacci number recursively

int fibonacci(int n) {

if (n <= 1) {

return n; // Base case: fibonacci(0) = 0, fibonacci(1) = 1

}

return fibonacci(n - 1) + fibonacci(n - 2); // Recursive case

}

// Function to print first n Fibonacci numbers

void printFibonacciSeries(int n) {

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

cout << fibonacci(i) << " "; // Print the ith Fibonacci number

}

cout << endl;

}

int main() {

int n;

cout << "Enter the number of terms in the Fibonacci series: ";

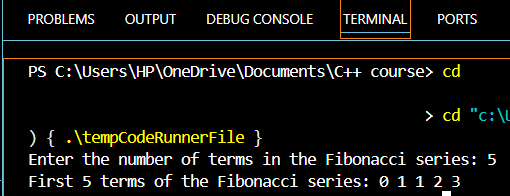
cin >> n;

cout << "First " << n << " terms of the Fibonacci series: ";

printFibonacciSeries(n);

return 0;

}



**Ques 18. Calculate avg of all elements of array**

#include <iostream>

using namespace std;

int main() {

int n;

// Input: size of the array

cout << "Enter the number of elements in the array: ";

cin >> n;

int arr[n]; // Declare array of size 'n'

int sum = 0;

// Input: elements of the array

cout << "Enter " << n << " elements of the array:\n";

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

cin >> arr[i]; // Take each element as input

sum += arr[i]; // Add the element to the sum

}

// Calculate the average

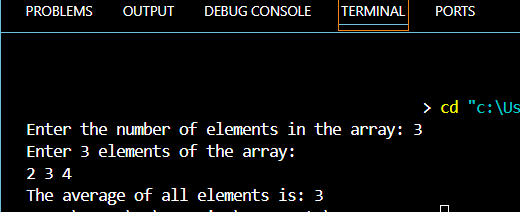
double average = static\_cast<double>(sum) / n;

// Output: the average

cout << "The average of all elements is: " << average << endl;

return 0;

}



**Ques 19. Find maximum minimum value of 1d array**

#include <iostream>

using namespace std;

int main() {

int n;

// Input: size of the array

cout << "Enter the number of elements in the array: ";

cin >> n;

int arr[n]; // Declare the array with size 'n'

// Input: elements of the array

cout << "Enter " << n << " elements of the array:\n";

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

cin >> arr[i]; // Input each element into the array

}

// Initialize max and min values with the first element of the array

int maxVal = arr[0];

int minVal = arr[0];

// Traverse the array to find the max and min values

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

if (arr[i] > maxVal) {

maxVal = arr[i]; // Update max value

}

if (arr[i] < minVal) {

minVal = arr[i]; // Update min value

}

}

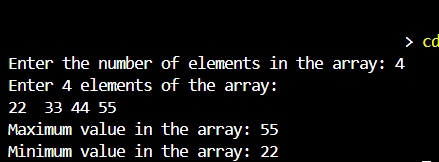
// Output: maximum and minimum values

cout << "Maximum value in the array: " << maxVal << endl;

cout << "Minimum value in the array: " << minVal << endl;

return 0;

}



**Ques 20. Print transpose of 2D matrix**

#include <iostream>

using namespace std;

int main() {

int rows, cols;

// Input: dimensions of the matrix

cout << "Enter the number of rows: ";

cin >> rows;

cout << "Enter the number of columns: ";

cin >> cols;

// Declare the matrix

int matrix[rows][cols];

// Input: elements of the matrix

cout << "Enter the elements of the matrix:\n";

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

for (int j = 0; j < cols; ++j) {

cin >> matrix[i][j];

}

}

// Create a matrix for the transpose

int transpose[cols][rows];

// Compute the transpose

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

for (int j = 0; j < cols; ++j) {

transpose[j][i] = matrix[i][j];

}

}

// Output: the original matrix

cout << "\nOriginal Matrix:\n";

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

for (int j = 0; j < cols; ++j) {

cout << matrix[i][j] << " ";

}

cout << endl;

}

// Output: the transposed matrix

cout << "\nTransposed Matrix:\n";

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

for (int j = 0; j < rows; ++j) {

cout << transpose[i][j] << " ";

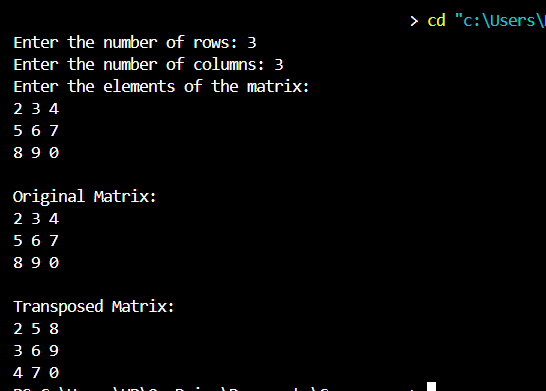
}

cout << endl;

}

return 0;

**}**



**Ques 21. Program to add 2D MATRIX**

#include <iostream>

using namespace std;

int main() {

int rows, cols;

// Input: dimensions of the matrices

cout << "Enter the number of rows: ";

cin >> rows;

cout << "Enter the number of columns: ";

cin >> cols;

// Declare two matrices (Matrix A and Matrix B) and a result matrix

int matrixA[rows][cols], matrixB[rows][cols], result[rows][cols];

// Input: elements of Matrix A

cout << "Enter the elements of Matrix A:\n";

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

for (int j = 0; j < cols; ++j) {

cin >> matrixA[i][j];

}

}

// Input: elements of Matrix B

cout << "Enter the elements of Matrix B:\n";

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

for (int j = 0; j < cols; ++j) {

cin >> matrixB[i][j];

}

}

// Add Matrix A and Matrix B and store the result in the result matrix

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

for (int j = 0; j < cols; ++j) {

result[i][j] = matrixA[i][j] + matrixB[i][j];

}

}

// Output: the sum of Matrix A and Matrix B

cout << "\nSum of Matrix A and Matrix B:\n";

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

for (int j = 0; j < cols; ++j) {

cout << result[i][j] << " ";

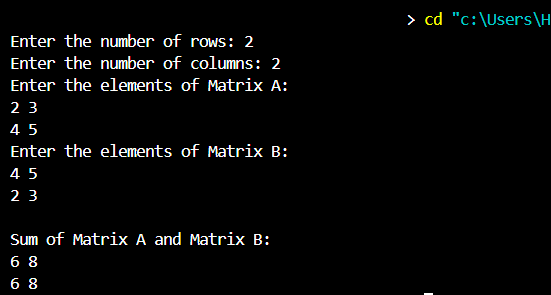
}

cout << endl;

}

return 0;

}



**Ques 22. Multiply 2D matrix**

#include <iostream>

using namespace std;

int main() {

int rowsA, colsA, rowsB, colsB;

// Input: dimensions of the first matrix

cout << "Enter the number of rows and columns for Matrix A: ";

cin >> rowsA >> colsA;

// Input: dimensions of the second matrix

cout << "Enter the number of rows and columns for Matrix B: ";

cin >> rowsB >> colsB;

// Matrix multiplication is only possible if colsA == rowsB

if (colsA != rowsB) {

cout << "Matrix multiplication is not possible. The number of columns of Matrix A must be equal to the number of rows of Matrix B." << endl;

return 0;

}

// Declare the matrices

int matrixA[rowsA][colsA], matrixB[rowsB][colsB], result[rowsA][colsB];

// Input: elements of Matrix A

cout << "Enter the elements of Matrix A:\n";

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

for (int j = 0; j < colsA; ++j) {

cin >> matrixA[i][j];

}

}

// Input: elements of Matrix B

cout << "Enter the elements of Matrix B:\n";

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

for (int j = 0; j < colsB; ++j) {

cin >> matrixB[i][j];

}

}

// Initialize the result matrix with zeros

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

for (int j = 0; j < colsB; ++j) {

result[i][j] = 0;

}

}

// Matrix multiplication

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

for (int j = 0; j < colsB; ++j) {

for (int k = 0; k < colsA; ++k) {

result[i][j] += matrixA[i][k] \* matrixB[k][j];

}

}

}

// Output: the resulting matrix after multiplication

cout << "\nResultant Matrix after multiplication:\n";

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

for (int j = 0; j < colsB; ++j) {

cout << result[i][j] << " ";

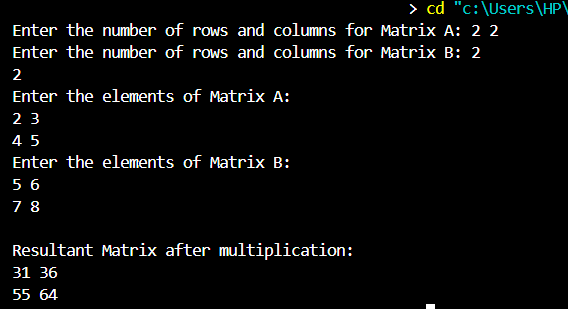
}

cout << endl;

}

return 0;

}



**Ques 23. Sort an array In ascending order**

#include <iostream>

#include <algorithm> // For the sort function

using namespace std;

int main() {

int n;

// Input: size of the array

cout << "Enter the number of elements in the array: ";

cin >> n;

int arr[n]; // Declare array of size 'n'

// Input: elements of the array

cout << "Enter " << n << " elements of the array:\n";

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

cin >> arr[i]; // Input each element into the array

}

// Sorting the array in ascending order

sort(arr, arr + n); // This function from <algorithm> sorts the array

// Output: sorted array

cout << "Array in ascending order:\n";

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

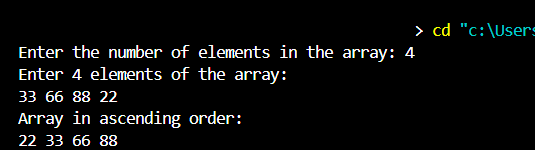
cout << arr[i] << " ";

}

cout << endl;

return 0;

}



**Ques 24. WAP to reverse a given string**

#include <iostream>

#include <string> // For using the string class

#include <algorithm> // For using the reverse function

using namespace std;

int main() {

string str;

// Input: Enter a string

cout << "Enter a string: ";

cin >> str; // Read a string from the user

// Reverse the string using the reverse function

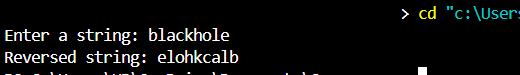
reverse(str.begin(), str.end());

// Output: The reversed string

cout << "Reversed string: " << str << endl;

**return 0;**

**}**



**Ques 25. WAP to count all vowels in a given string**

#include <iostream>

#include <string>

using namespace std;

int main() {

string str;

int vowelCount = 0;

// Input: Enter a string

cout << "Enter a string: ";

getline(cin, str); // Read the entire line including spaces

// Loop through the string and count vowels

for (char c : str) {

// Check if the character is a vowel (both uppercase and lowercase)

if (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u' ||

c == 'A' || c == 'E' || c == 'I' || c == 'O' || c == 'U') {

vowelCount++;

}

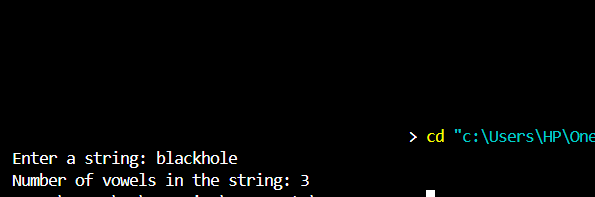
}

// Output: The number of vowels

cout << "Number of vowels in the string: " << vowelCount << endl;

return 0;

}



**Q26. WAP to check if a given string is palindrome or not.**

#include<iostream>

using namespace std;

int main()

string st:

cout<<"Enter a string \n";

cin>>st:

int flag=0;

int len=st.size();

for (int i=0;i<len/2;i++)

if(st[i]! = st[len-1-i])

{

flag=1;

}

if(flag==0)

cout<<"Palindrome Word":

else

cout<<" Not Palindrome Word":

return 0;

}



**Q27. WAP to check if a given string is anagram or not.**

#include<iostream>

using namespace std;

int main(){

int arr[26]={0};

cout << "enter a size:";

int size;

cin >> size;

cout << "s1:";

char s1[size];

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

cin >> s1[i];

}

char s2[size];

cout << "s2:";

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

cin >> s2[i];

}

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

int a = s1[i]-'a';

arr[a]=arr[a]+1;

}

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

int a = s2[i]-'a';

arr[a]=arr[a]-1;;

// cout << a << endl;

}

int flag=0;

for(int i=0;i<26;i++){

if(arr[i]!=0){

flag=1;

break;

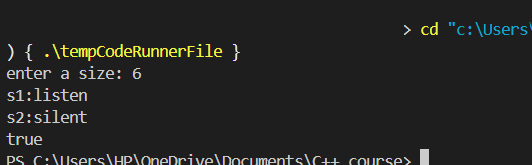
}

}

(flag==0)? cout << "true" : cout << "false";

return 0;

}

****

**Q28. Define a class called Car with attributes such as make, model, and year. Include member functions to set and get these attributes. Create an object of the Car class and demonstrate the use of its member functions.**

#include<iostream>

using namespace std;

class car{

string make;

string model;

int year;

public:

void setData(){

cout << "enter make:";

cin >> make;

cout << "enter model:";

cin >> model;

cout << "year:";

cin >> year;

}

void getData(){

cout << "make:" << make << endl;

cout << "model:" << model << endl;

cout << "year:" << year << endl;

}

};

int main(){

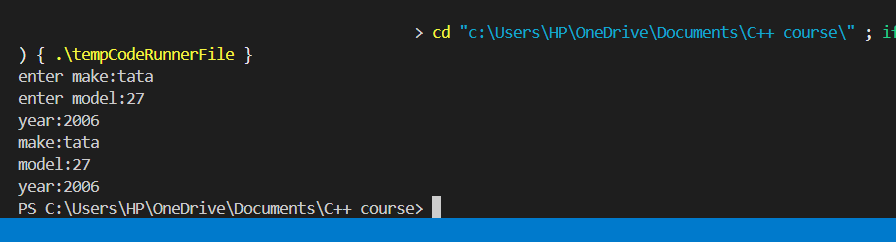
car c1;

c1.setData();

c1.getData();

return 0;

}

****

**Q29. Define a class called Address with attributes such as street, city, and zipCode. Create a class called Person that has an Address object as a member variable. Demonstrate composition by creating a Person object and accessing its Address attributes.**

#include <iostream>

#include <string>

using namespace std;

class Address {

public:

string street;

string city;

string zipCode;

Address(string s, string c, string z)

{

street = s;

city = c;

zipCode = z;

}

};

class Person {

public:

string name;

Address address;

Person(string n, string s, string c, string z) {

name = n;

address = Address(s, c, z); // Initialize Address object

}

// Function to display Person details

void displayDetails() {

cout << "Name: " << name << endl;

cout << "Address: " << address.street << ", " << address.city << " " << address.zipCode << endl;

}

};

int main() {

Person person("John Doe", "123 Main St", "Anytown", "12345");

person.displayDetails();

return 0;

**Q30. Write a program to display the minimum, maximum, sum, search and average of elements of an array.**

#include <iostream>

using namespace std;

int main() {

int n;

cout << "Enter the number of elements: ";

cin >> n;

int arr[n];

cout << "Enter the elements: ";

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

cin >> arr[i];

}

// Find minimum

int minVal = arr[0];

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

if (arr[i] < minVal) {

minVal = arr[i];

}

}

// Find maximum

int maxVal = arr[0];

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

if (arr[i] > maxVal) {

maxVal = arr[i];

}

}

// Calculate sum

int sum = 0;

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

sum += arr[i];

}

// Search for an element

int searchVal;

cout << "Enter the value to search: ";

cin >> searchVal;

bool found = false;

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

if (arr[i] == searchVal) {

found = true;

break;

}

}

if (found) {

cout << "Value found in the array." << endl;

} else {

cout << "Value not found in the array." << endl;

}

// Calculate average

double average = (double)sum / n;

// Display results

cout << "Minimum value: " << minVal << endl;

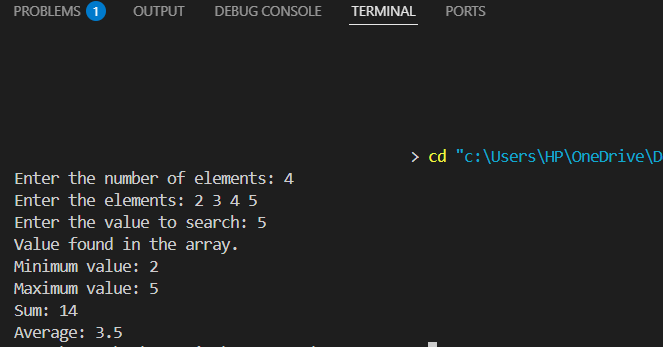
cout << "Maximum value: " << maxVal << endl;

cout << "Sum: " << sum << endl;

cout << "Average: " << average << endl;

return 0;

**}**

****

**Q31. Define a class student with the following specification**

**Private members of class student**

**admno integer**

**sname 20 character**

**eng. math, science float**

**total float**

**Public member function of class student**

**ctotal() a function to calculate eng + math + science with float return type.**

**Takedata() Function to accept values for admno, sname, eng, science Showdata() Function to display all the data members on the screen**

#include <iostream>

using namespace std;

class Student {

private:

int admno;

char sname[20];

float eng, math, science;

float total;

public:

// Function to calculate total

float ctotal() {

total = eng + math + science;

return total;

}

void takeData() {

cout << "Enter admission number: ";

cin >> admno;

cout << "Enter student name: ";

cin >> sname;

cout << "Enter English marks: ";

cin >> eng;

cout << "Enter Math marks: ";

cin >> math;

cout << "Enter Science marks: ";

cin >> science;

}

void showData() {

cout << "Admission Number: " << admno << endl;

cout << "Student Name: " << sname << endl;

cout << "English Marks: " << eng << endl;

cout << "Math Marks: " << math << endl;

cout << "Science Marks: " << science << endl;

cout << "Total Marks: " << ctotal() << endl;

}

};

int main() {

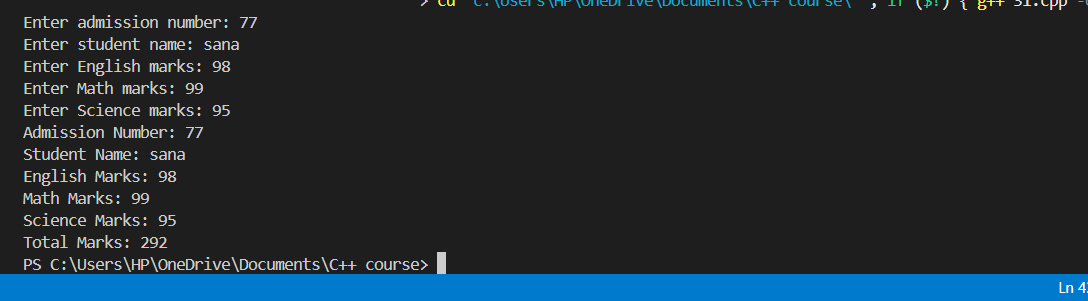
Student student;

student.takeData();

student.showData();

return 0;

}

****

**Q32. Define a class in C++ with following description:**

**Private Members**

**A data member Flight number of type integer**

**A data member Destination of type string**

**A data member Distance of type float**

**A data member Fuel of type float**

**A member function CALFUEL() to calculate the value of Fuel as per the following criteria**

**Distance Fuel**

**<=1000 500**

**more than 1000 and <=2000 1100**

**more than 2000 2200**

**Public Members**

**A function FEEDINFO() to allow user to enter values for Flight Number, Destination, Distance & call function CALFUEL() to calculate the quantity of Fuel.**

**A function SHOWINFO() to allow user to view the content of all the data members.**

#include<iostream>

using namespace std;

class travel{

int flightNumber;

string destination;

int distance;

float fuel;

void calFuel(){

if(distance<=1000) fuel=500;

else if(distance>1000 && distance<=2000) fuel=1100;

else fuel=2200;

}

public :

void feedInfo(int fn,string des,int dist){

flightNumber=fn;

destination=des;

distance=dist;

calFuel();

}

void showInfo(){

cout << "flight number:" << flightNumber << endl;

cout << "destination:" << destination << endl;

cout << "distance:" << distance << endl;

cout << "fuel:" << fuel << endl;

}

};

int main(){

travel t1;

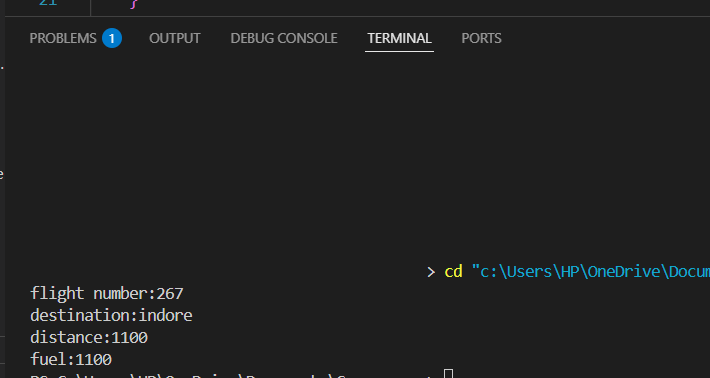
t1.feedInfo(267,"indore",1100);

t1.showInfo();

return 0;

}

}



**Q33. Write a menu driven program to perform following:**

**a)Input a matrix**

**b)Display matrix**

**c)Add two matrix**

**d)Multiply two matrix**

**e)Transpose a matrix**

#include<iostream>

using namespace std;

class matrix{

int arr1[3][3];

int arr2[3][3];

public :

void Switch(int button){

switch (button){

case 1 :

inputdata();

break;

case 2 :

displaydata();

break;

case 3 :

add();

break;

case 4:

multiply();

break;

case 5:

transpose();

break;

default:

printf("Default case is Matched.");

break;

}

}

void inputdata(){

cout << "array 1:";

for(int i=0;i<3;i++){

for(int j=0;j<3;j++) {

cin >> arr1[i][j];

}

}

cout << "array 2:";

for(int i=0;i<3;i++){

for(int j=0;j<3;j++){

cin >> arr2[i][j];

}

}

}

void displaydata(){

cout << "array 1:";

for(int i=0;i<3;i++){

for(int j=0;j<3;j++) {

cout << arr1[i][j] << " ";

}

cout << endl;

}

cout << "array 2:";

for(int i=0;i<3;i++){

for(int j=0;j<3;j++) {

cout << arr2[i][j] << " ";

}

cout << endl;

}

}

void add(){

cout << "sum of two matrix";

for(int i=0;i<3;i++){

for(int j=0;j<3;j++){

int r= arr1[i][j]+arr2[i][j];

cout << r << " ";

}

cout << endl;

}

}

void multiply(){

printf("the resultant matrix\n");

for(int i=0;i<3;i++){

int d=0;

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

d=arr2[i][j];

arr2[i][j]=arr2[j][i];

arr2[j][i]=d;

}

}

int r=0;

for(int i=0;i<3;i++){

for(int j=0;j<3;j++){

for(int k=0;k<3;k++){

r=arr1[i][k]\*arr2[j][k]+r;

}

cout << r << " ";

}

cout << endl;

}

}

void transpose(){

cout << "transpose of both matrix:";

for(int i=0;i<3;i++){

int d=0;

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

d=arr1[i][j];

arr1[i][j]=arr1[j][i];

arr1[j][i]=d;

}

}

for(int i=0;i<3;i++){

int d=0;

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

d=arr2[i][j];

arr2[i][j]=arr2[j][i];

arr2[j][i]=d;

}

}

displaydata();

}

};

int main(){

matrix m1;

cout << "enter 1 for input matrix:" << endl;

cout << "enter 2 for output matrix:" << endl;

cout << "enter 3 for add two matrix:" << endl;

cout << "enter 4 for multiply two matrix:" << endl;

cout << "enter 5 for transpose of matrix:" << endl;

int button;

cout << "enter button:";

cin >> button;

m1.Switch(button);

cout << "enter button:";

cin >> button;

m1.Switch(button);

cout << "enter button:";

cin >> button;

m1.Switch(button);

return 0;

}

