**NAME – KHUSHI PANWAR**

**ROLL NO – 2021334**

**C++ ASSIGNMENT**

**1. Write a program to compute the sum of the first n terms of the following series: S = 1 - 1 / (2 ^ 2) + 1 / (3 ^ 3) - ... 1 / (n ^ n) where ^ is exponentiation. The number of terms n is to be taken from user through command line. If command line 12 argument is not found then prompt the user to enter the value of n.**

#include <iostream>

#include <math.h>

using namespace std;

int main(int arg, char \*arr[])

{

    int n;

    cout << "We have to perform of Sum of Series. " << endl;

    cout << "S = 1 - 1 / (2 ^ 2) + 1 / (3 ^ 3) - ... 1 / (n ^ n)" << endl;

    cout << "where n is number of terms. " << endl;

    if (arg == 2)

    {

        // atoi is a predefined function used to convert a string value to an integer value.

        n = atoi(arr[1]);

    }

    else if (arg != 2)

    {

        cout << "\nEnter the value of n here: ";

        cin >> n;

    }

    float sum = 0.0;

    float i = 1;

    while (i <= n)

    {

        sum = sum + pow(-1, i + 1) \* (1.0 / (i \* i));

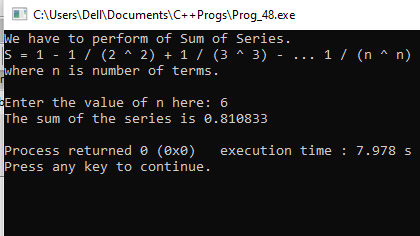
        i = i + 1;

    }

    cout << "The sum of the series is " << sum << endl;

    return 0;

}

****

**2. Write a program to remove the duplicates from an array.**

#include <iostream>

#include <iomanip>

using namespace std;

int main(){

int size;

    cout<<endl;

    cout<<setw(15)<< "\_\_ CREATE & DISPLAY THE ARRAY \_\_"<<endl<<endl;

//taking size of the 1D array

    cout<<"\t ENTER THE SIZE OF REQUIRED ARRAY : ";

    cin>>size;

    cout<<endl;

//declare array

    int arrayNum[size];

//taking inputs for array

    cout<<"\t Now Enter "<<size<<" elements for the array"<<endl<<endl;

    for (int i=0; i<size; i++){

        cout<<"Enter the element : ";

        cin>>arrayNum[i];

    }

//displaying array

    cout<<endl<<"-> ELEMENTS OF THE GIVEN ARRAY ARE : ";

    for (int i=0; i<size; i++){

        cout<<setw(4)<<arrayNum[i];

    }

    cout<<endl<<endl<< "\_\_ DELETE DUPLICATE ELEMENTS FROM THE ARRAY \_\_"<<endl;

//Delete duplicates from the array

    int duplicate=arrayNum[0];

    for (int i=0; i<=size; i++) {

        for(int j=i+1; j<size;){

            if(arrayNum[i]==arrayNum[j]) {

                for (int k=j; k<size-1; ++k) {

                    arrayNum[k]=arrayNum[k+1];

                    --size;

                } }

            else ++j;

        }

    }

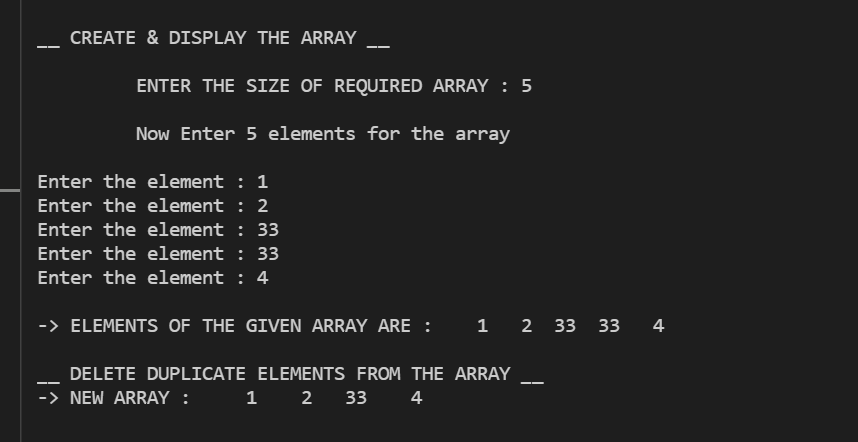
    cout<<"-> NEW ARRAY : ";

    for (int i=0; i<size; ++i)

           cout<<setw(5)<<arrayNum[i];

    return 0;

}



**3. Write a program that prints a table indicating the number of occurrences of each alphabet in the text entered as command line arguments.**

Ans -

#include <iostream>

#include <string.h>

    using namespace std;

int main(int argc, char \*argv[])

{

    int i;

    int count = 0;

    cout << "Parameters are: ";

    for (int i = 0; i < argc; i++)

        cout << argv[i] << " ";

    for (char k = 'a'; k <= 'z'; k++)

    {

        for (i = 0; i < argc; i++)

        {

            for (int j = 0; argv[i][j] != '\0'; j++)

            {

                argv[i][j] = tolower(argv[i][j]);

                if (argv[i][j] == k)

                    count++;

            }

        }

        if (count > 0)

            cout << "\n"

                 << k << " occurs " << count << " times.";

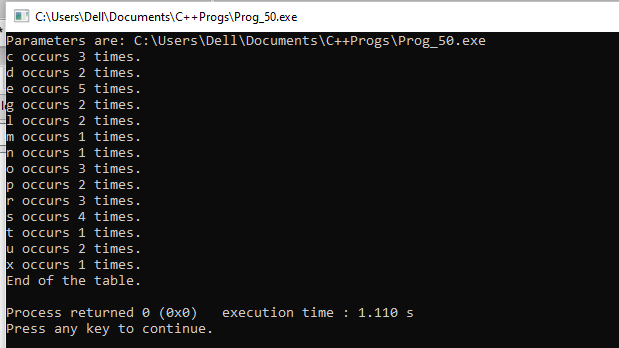
        count = 0;

    }

    cout << "\nEnd of the table." << endl;

    return 0;

}

****

**4. Write a menu driven program to perform following operations on strings (without using inbuilt string functions):**

**a) Show address of each character in string**

**b) Concatenate two strings.**

**c) Compare two strings**

**d) Calculate length of the string (use pointers)**

**e) Convert all lowercase characters to uppercase**

**f) Reverse the string**

#include <iostream>

#include<cstdlib>

#include<string.h>

#include<iomanip>

using namespace std;

void printaddress(char a[])

{

    void\* p;

    for(int i=0;i<strlen(a);i++)

    {

        p= &a[i];

        cout<<"Character"<<setw(2)<<a[i]<<setw(9)<<" address is: "<<p<<endl;

    }

}

int length(char\* p)

{

    int i=0;

    while(\*p)

    {

        i++;

        p=p+1;

    }

    return i;

}

void concatenate(char\* p, char\* p1)

{

    while(\*p!='\0')

    p++;

    while(\*p1!='\0')

    {

        \*p=\*p1;

        p++;

        p1++;

    }

    \*p='\0';

}

void relationalOperation(char s1[],char s2[])

{

}

void reverse(char p[])

{

    char temp;

    for(int i=0,j=length(p)-1;i<length(p)/2;i++,j--)

    {

        temp=p[i];

        p[i]=p[j];

        p[j]=temp;

    }

}

void case\_changer(char s[])

{

    int i;

    for(i=0;i<strlen(s);i++)

    if(islower(s[i]))

    {

        s[i]=s[i]-32;

    }

    return;

}

int main()

{

    char s1[20],s2[20];

    int count,x1,p1,p2;

    cout<<endl<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

    cout<<"1. Input a string: "<<endl;

    cout<<"2. Print address of each character of the string: "<<endl;

    cout<<"3. Concatenation of two strings: "<<endl;

    cout<<"4. Comparison of two strings: "<<endl;

    cout<<"5. Length of string: "<<endl;

    cout<<"6. Conversion of all lowercase characters to uppercase : "<<endl;

    cout<<"7. Reverse the string: "<<endl;

    cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl<<endl;

    char ch='y';

    while(ch=='y')

    {

        cout<<"Enter your choice (from 1 to 7): ";

        cin>>x1;

        switch(x1)

        {

            case 1: cout<<"Enter the first string: ";

            cin>>s1;

            cout<<"Enter the second string: ";

            cin>>s2;

            break;

            case 2: printaddress(s1);

            printaddress(s2);

            break;

            case 3: concatenate(s1,s2);

            cout<<"Concatenation of two string are: "<<s1<<endl;

            break;

            case 4: relationalOperation(s1,s2);

            if(s1==s2)

            {

            cout<<s1<<" is equal to "<<s2<<endl;

            }

            else

            cout<<s1<<" is not equal to "<<s2<<endl;

            break;

            case 5: p1=length(s1);

            cout<<"Length of the first string is: "<<p1<<endl;

            p2=length(s2);

            cout<<"Length of the second string is: "<<p2<<endl;

            break;

            case 6: case\_changer(s1);

            cout<<"Uppercase of the first string is: "<<s1<<endl;

            case\_changer(s2);

            cout<<"Uppercase of the second string is: "<<s2<<endl;

            break;

            case 7: reverse(s1);

            cout<<"Reverse of the first string is: "<<s1<<endl;

            reverse(s2);

            cout<<"Reverse of the second string is: "<<s2<<endl;

            break;

            default: cout<<"Invalid choice!! "<<endl;

        }

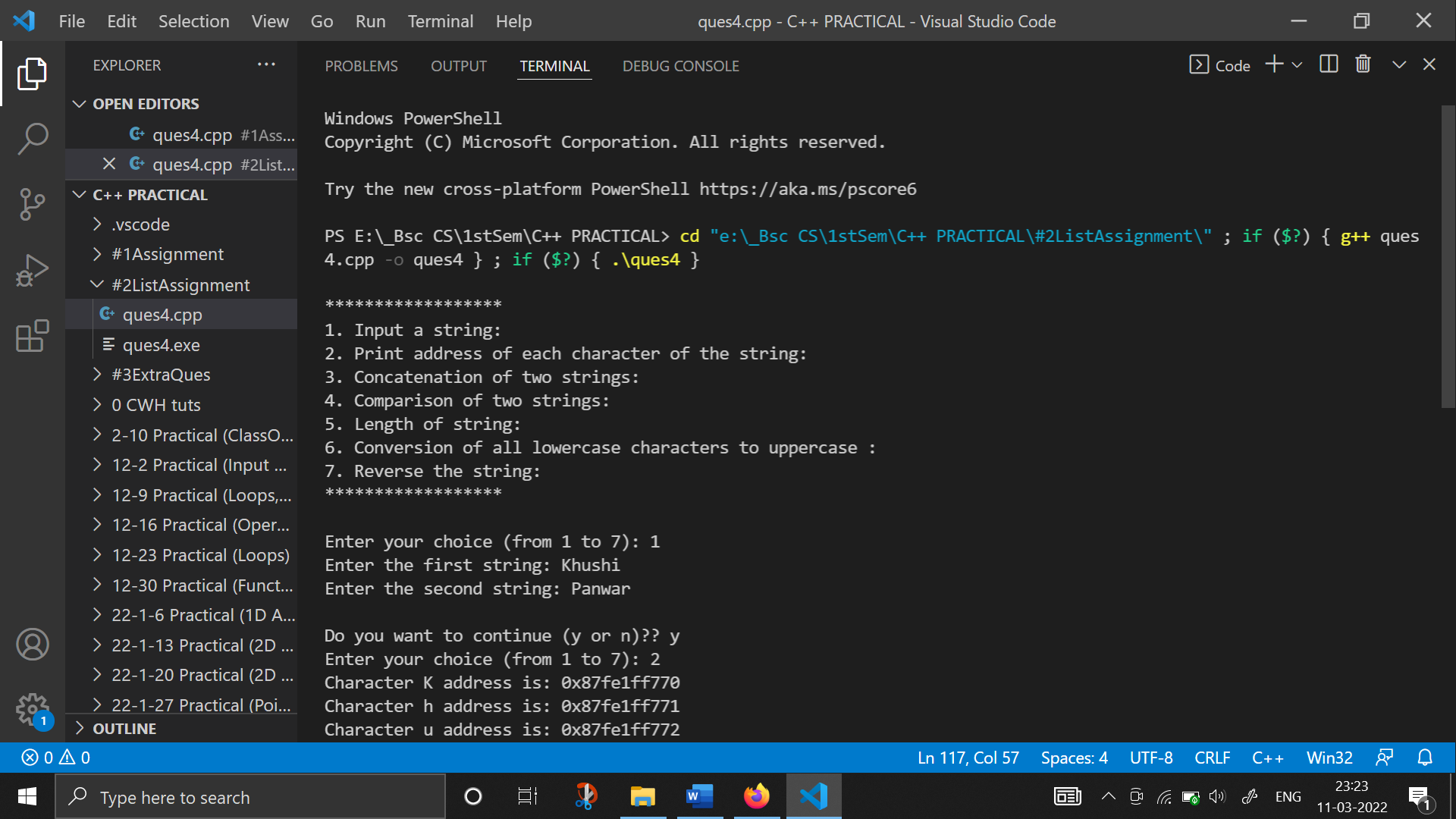
        cout<<endl<<"Do you want to continue (y or n)?? ";

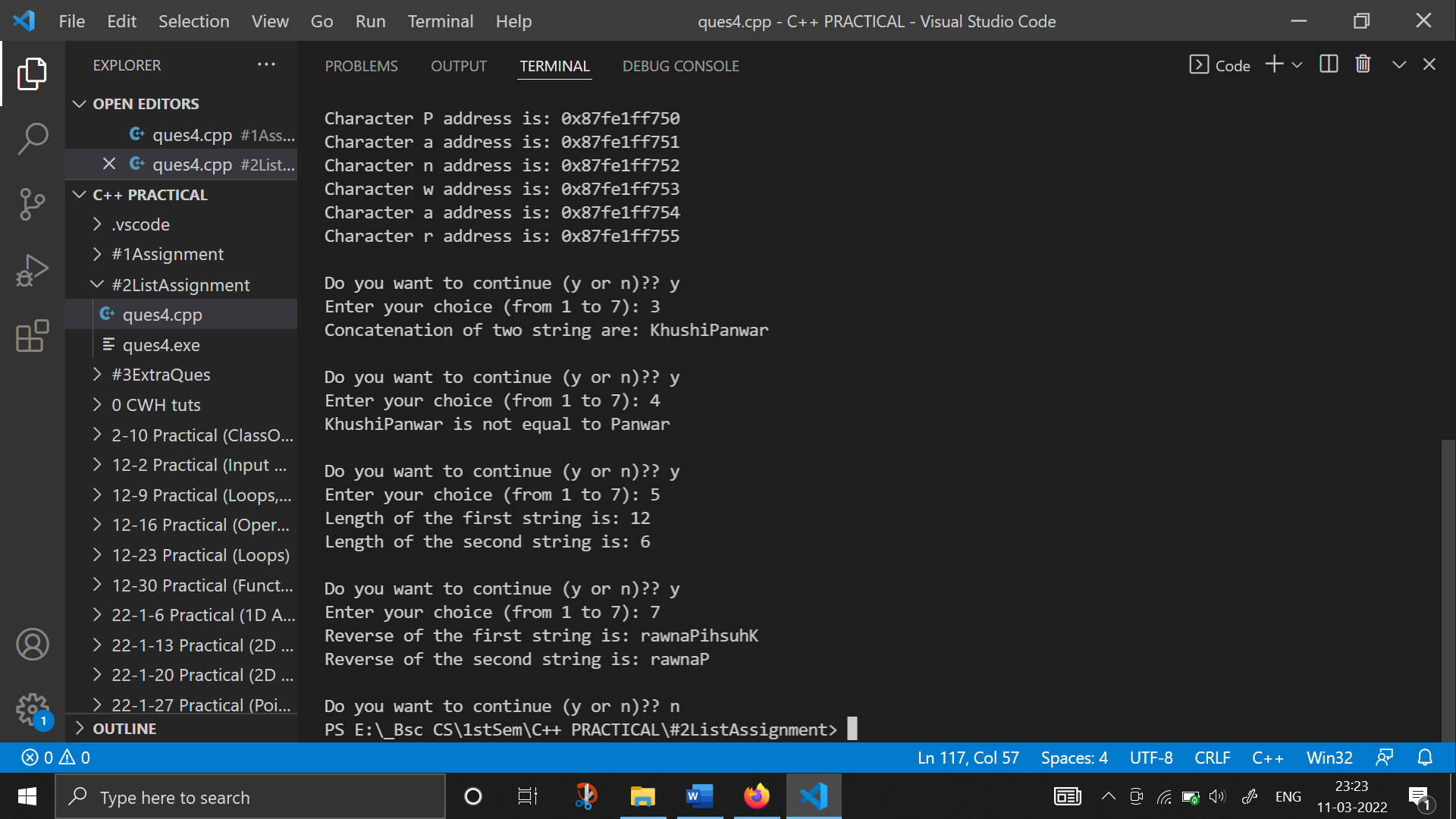
        cin>>ch;

    }

    return 0;

}





**5. Write a program to merge two ordered arrays to get a single ordered array.**

#include<iostream>

using namespace std;

int main()

{

    int arrOne[5], arrTwo[5], arrMerge[10];

    int sizeOne, sizeTwo, i, k;

    cout<<endl<<"\*\* THIS PROGRAM MERGES TWO ARRAYS \*\*"<<endl<<endl;

    cout<<"-> Enter the Size for First Array: ";

    cin>>sizeOne;

    cout<<"Enter "<<sizeOne<<" Elements for First Array: ";

    for(i=0; i<sizeOne; i++)

    {

        cin>>arrOne[i];

        arrMerge[i] = arrOne[i];

    }

    k = i;

    cout<<"\nEnter the Size for Second Array: ";

    cin>>sizeTwo;

    cout<<"Enter "<<sizeTwo<<" Elements for Second Array: ";

    for(i=0; i<sizeTwo; i++)

    {

        cin>>arrTwo[i];

        arrMerge[k] = arrTwo[i];

        k++;

    }

    cout<<"\nThe New Array (Merged Array):\n";

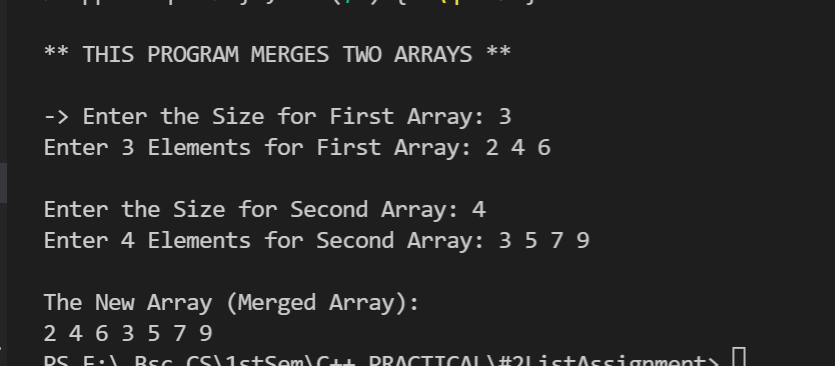
    for(i=0; i<k; i++)

        cout<<arrMerge[i]<<" ";

    cout<<endl;

    return 0;

}



**6. Write a program to search a given element in a set of N numbers using Binary search**

**(i) with recursion**

#include<iostream>

using namespace std;

int binarysearch(int ar[],int l,int h,int n)

{

if(l<=h)

{

int m=(l+h)/2;

if(ar[m]==n)

{

return m;

}

else if(ar[m]<n)

{

return binarysearch(ar,m+1,h,n);

}

else

{

return binarysearch(ar,l,m-1,n);

}

}

return -1;

}

int main()

{

int ar[]={1,2,5,7,9,0};

int n=2,l,h,m,index;

l=0;

h= (sizeof(ar)/sizeof(ar[0]))-1;

index =binarysearch(ar,l,h,n);

if(index==-1)

{

cout<<"Number not found ";

}

else

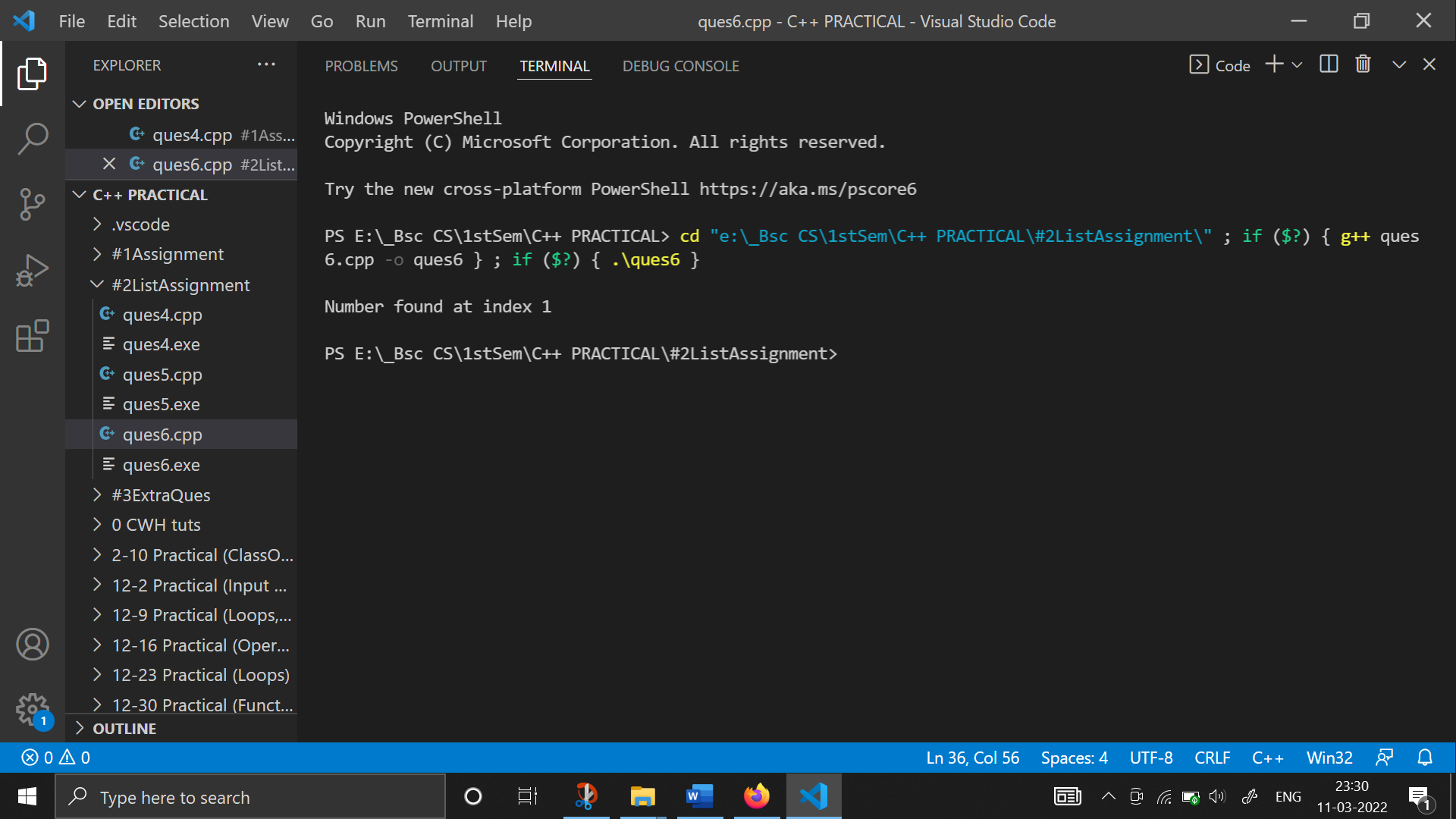
{

cout<<endl<<"Number found at index "<<index<<endl<<endl;

}

return 0;

}



**(ii) without recursion**

#include<iostream>

using namespace std;

int main()

{

    int i, arr[10], num, first, last, middle;

    cout<<endl<<"\t \*\* BINARY SEARCH USING RECURSION \*\*"<<endl<<endl;

    cout<<endl<<"Enter 10 Elements (in ascending order): ";

    for(i=0; i<10; i++)

        cin>>arr[i];

    cout<<"\nEnter Element to be Search: ";

    cin>>num;

    first = 0;

    last = 9;

    middle = (first+last)/2;

    while(first <= last)

    {

        if(arr[middle]<num)

            first = middle+1;

        else if(arr[middle]==num)

        {

            cout<<"\nThe number, "<<num<<" found at Position "<<middle+1;

            break;

        }

        else

            last = middle-1;

        middle = (first+last)/2;

    }

    if(first>last)

        cout<<"\nThe number, "<<num<<" is not found in given Array";

    cout<<endl;

    return 0;

}



**7. Write a program to calculate GCD of two numbers**

**(i) with recursion**

#include <iostream>

using namespace std;

int gcd(int a, int b)

{

    if (a == 0)

        return b;

    if (b == 0)

        return a;

    if (a == b)

        return a;

    if (a > b)

        return gcd(a - b, b);

    return gcd(a, b - a);

}

int main()

{

    int a, b;

    cout << endl

         << "\t\*\* FINDING GCD USING RECUSION \*\*" << endl

         << endl;

    cout << "Enter the two numbers : ";

    cin >> a >> b;

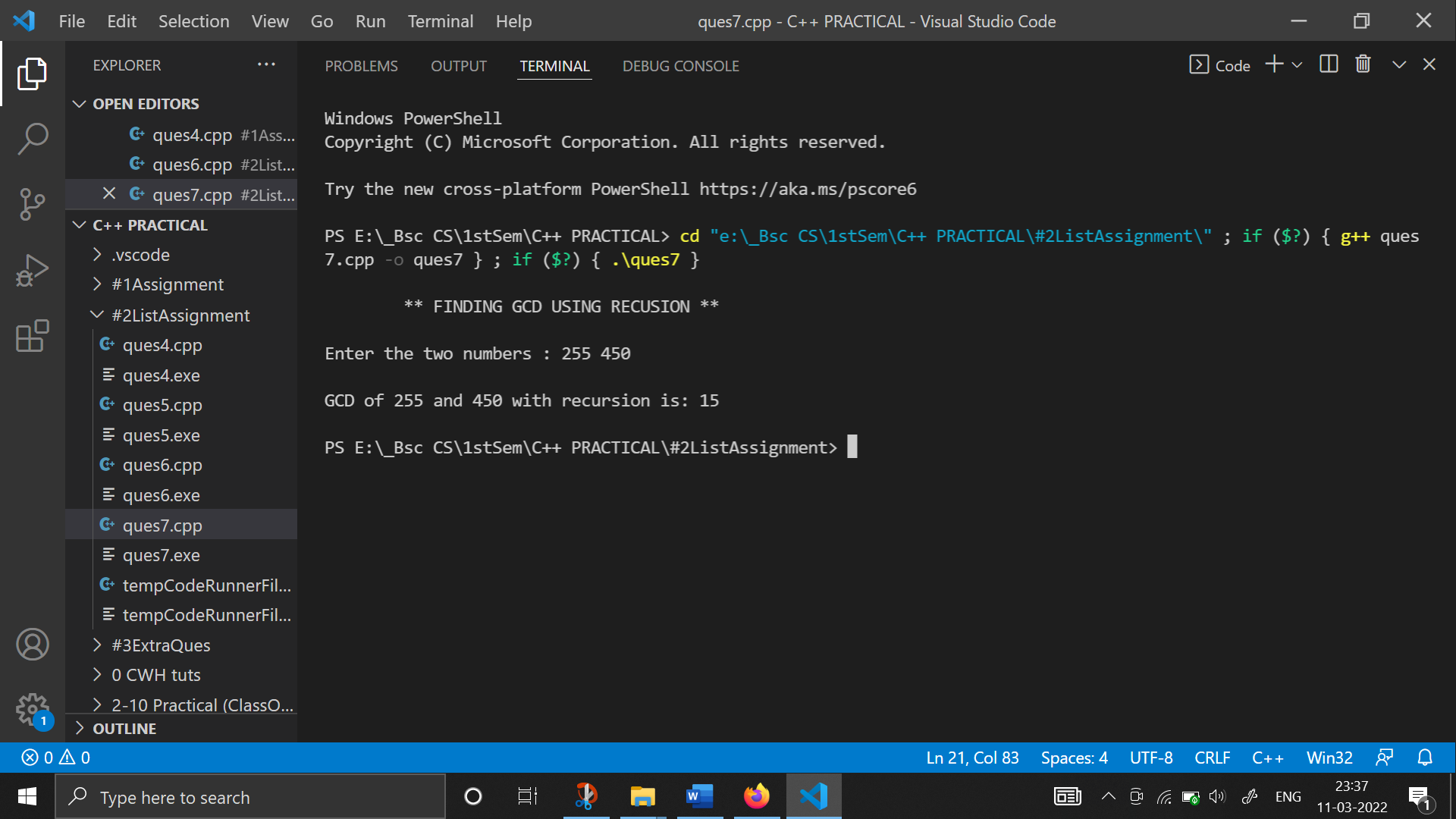
    cout << endl

         << "GCD of " << a << " and " << b << " with recursion is: " << gcd(a, b) << endl

         << endl;

    return 0;

}



**(ii) without recursion**

#include <iostream>

using namespace std;

int main()

{

    int num1, num2, gcd;

    cout << endl

         << "\t \*\*GCD WITHOUT RECURSION \*\*" << endl

         << endl;

    cout << " Input the first number : ";

    cin >> num1;

    cout << " Input the second number : ";

    cin >> num2;

    for (int i = 1; i <= num1 && i <= num2; i++)

    {

        if (num1 % i == 0 && num2 % i == 0)

        {

            gcd = i;

        }

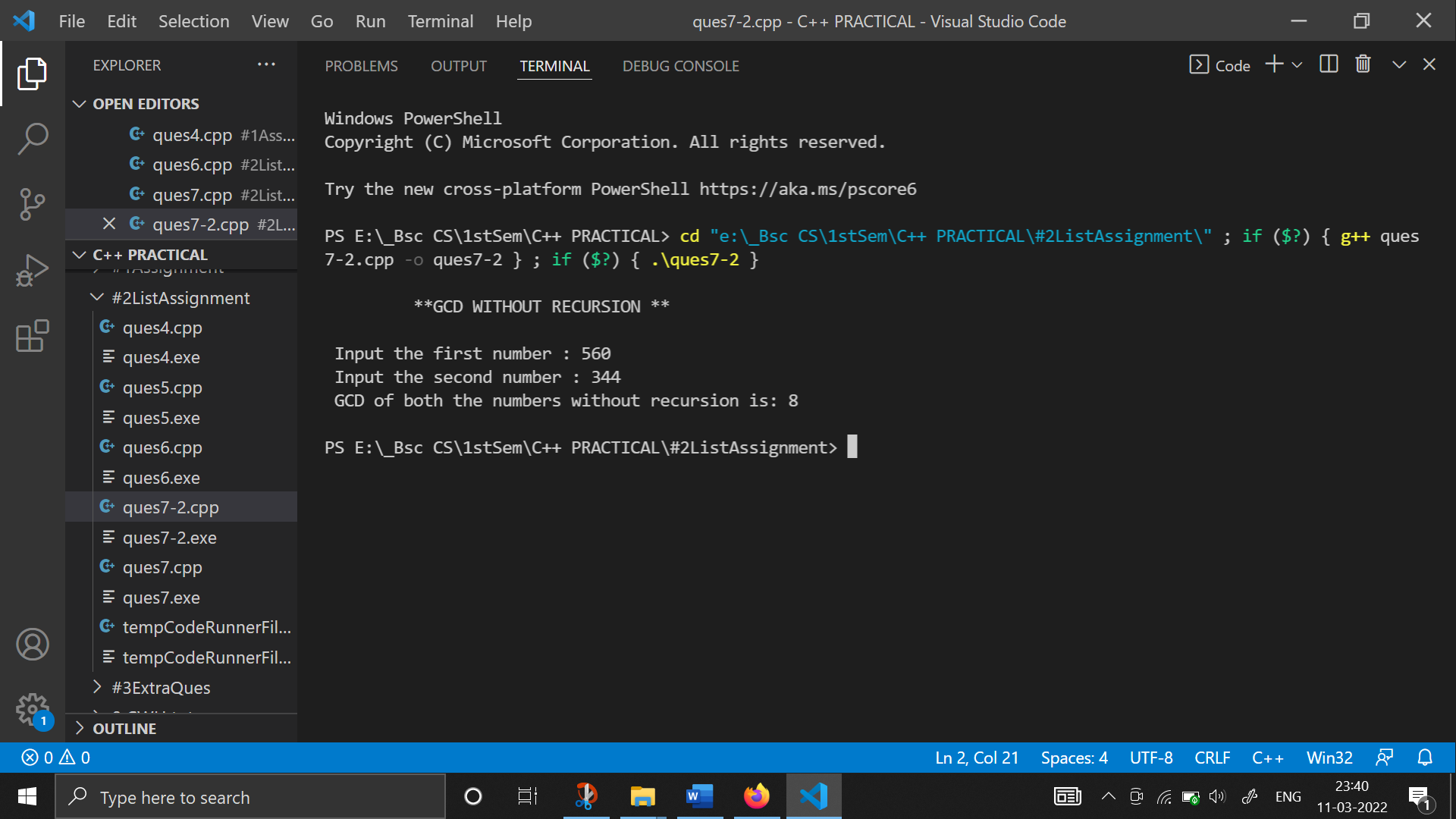
    }

    cout << " GCD of both the numbers without recursion is: " << gcd << endl

         << endl;

    return 0;

}



**8. Create Matrix class. Write a menu-driven program to perform following Matrix operations: a) Sum**

**b) Product**

**c) Transpose**

#include <iostream>

#include <iomanip>

using namespace std;

int inputMatrix(int m[][3]){

    for (int i=0; i<3; i++){

        cout<<"Enter the elements for row "<<i+1<<" : ";

        for (int j=0; j<3; j++)

        cin>>m[i][j];

    }

}

int addition(int m1[][3], int m2[][3], int sum[][3]){

    for (int i=0; i<3; i++){

        for (int j=0; j<3; j++)

            sum[i][j]=m1[i][j]+m2[i][j];

    }

}

int subtraction(int m1[][3], int m2[][3], int difference[][3]){

    for (int i=0; i<3; i++){

        for (int j=0; j<3; j++)

            difference[i][j]=m1[i][j]-m2[i][j];

    }

}

int multiplication(int m1[][3], int m2[][3], int prod[][3]){

    for (int i=0; i<3; i++){

        for (int j=0; j<3; j++)

            prod[i][j]=m1[i][j]\*m2[i][j];

    }

}

int transpose(int m[][3], int trans[][3]){

    for (int i=0; i<3; i++){

        for (int j=0; j<3; j++){

            trans[j][i]=m[i][j];

        }

    }

}

int displayMatrix(int m[][3]){

    cout<<endl<<"============================="<<endl<<endl;

    for (int i=0; i<3; i++){

        for (int j=0; j<3; j++)

        cout<<setw(5)<<m[i][j];

        cout<<endl;

    }

    cout<<endl<<"============================="<<endl<<endl;

}

int main(){

    cout<<"\t \* MATRIX OPERATIONS (3x3) \*" <<endl<<endl;

    cout<<"========================"<<endl;

    cout<<"\t SELECT : "<<endl<<endl;

    cout<<" 1. MATRIX ADDITION "<<endl;

    cout<<" 2. MATRIX SUBTRACTION "<<endl;

    cout<<" 3. MATRIX MULTIPLICATION "<<endl;

    cout<<" 4. MATRIX TRANSPOSE "<<endl;

    cout<<"========================"<<endl;

    int choice, size;

    int matrix1[3][3];

    int matrix2[3][3];

    int difference[3][3];

    int sum[3][3];

    int prod[3][3];

    int trans[3][3];

    char ch='y';

    while (ch=='y'){

        cout<<"What operation do you want to perform (1,2,3 or 4)?? ";

        cin>>choice;

        cout<<endl<<"\t \* ENTER ELEMENTS FOR MATRIX 1 \*"<<endl;

        inputMatrix(matrix1);

        cout<<endl<<"\t \* ENTER ELEMENTS FOR MATRIX 2 \*"<<endl;

        inputMatrix(matrix2);

    switch(choice){

        case 1: addition(matrix1, matrix2, sum);

                cout<<endl<<setw(20)<<"\* SUM OF MATRIX \*";

                displayMatrix(sum);

                break;

        case 2: subtraction(matrix1, matrix2, difference);

                cout<<endl<<setw(20)<<" \* DIFFERENCE OF MATRIX \*";

                displayMatrix(difference);

                break;

        case 3: multiplication(matrix1, matrix2, prod);

                cout<<endl<<setw(20)<<" \* PRODUCT OF MATRIX \*";

                displayMatrix(prod);

                break;

        case 4: transpose(matrix1, trans);

                cout<<endl<<setw(20)<<" \* TRANSPOSE OF MATRIX 1 \*";

                displayMatrix(trans);

                transpose(matrix2, trans);

                cout<<endl<<setw(20)<<" \* TRANSPOSE OF MATRIX 2 \*";

                displayMatrix(trans);

                break;

        default : cout<<"INVALID CHOICE! "<<endl;

    }

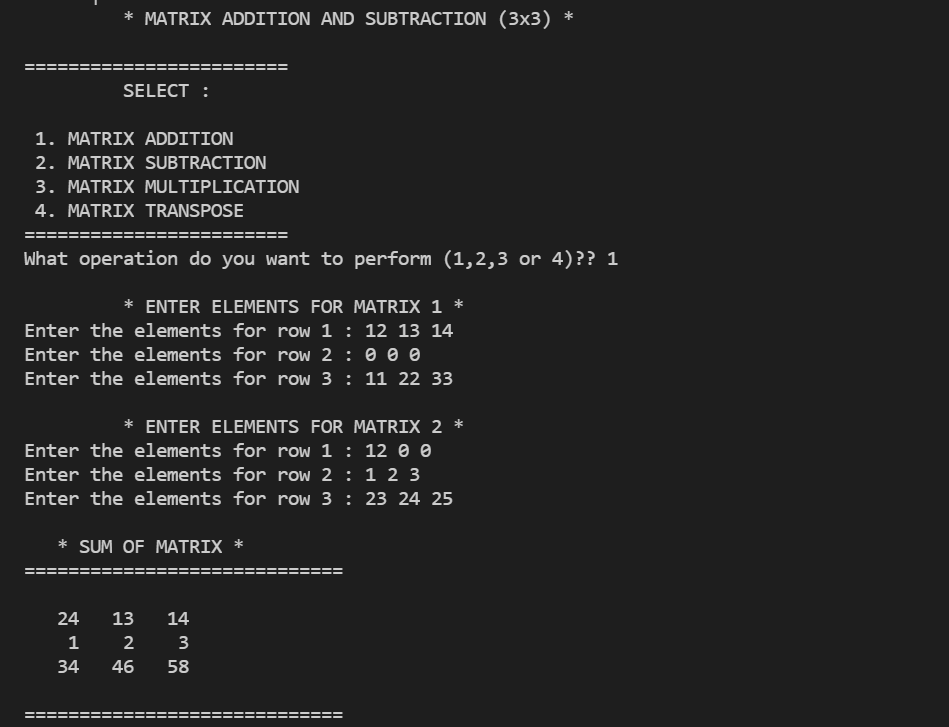
    cout<<"Do you want to continue(y/n)? ";

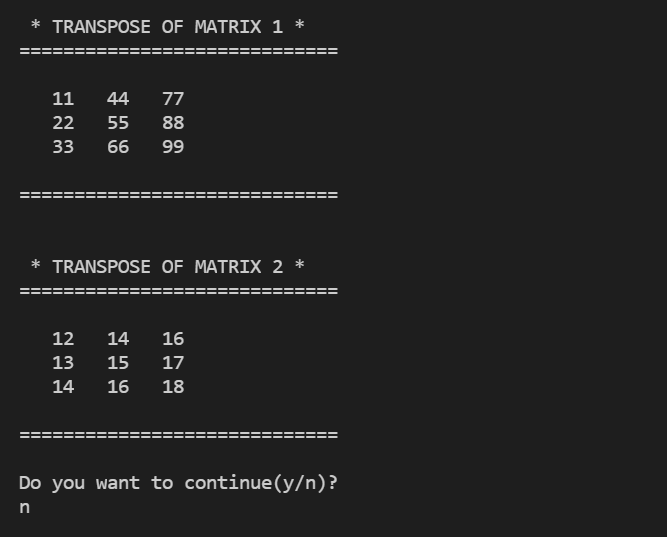
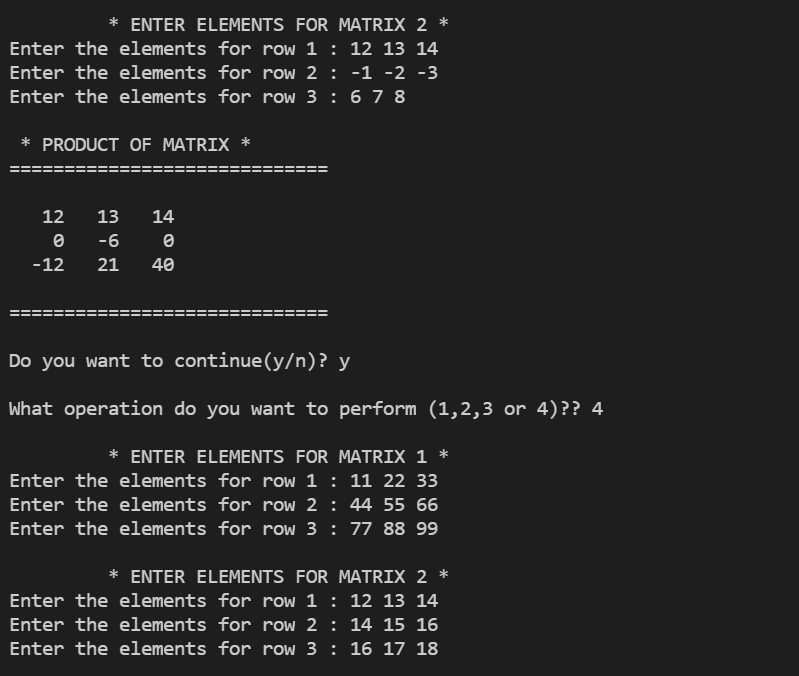
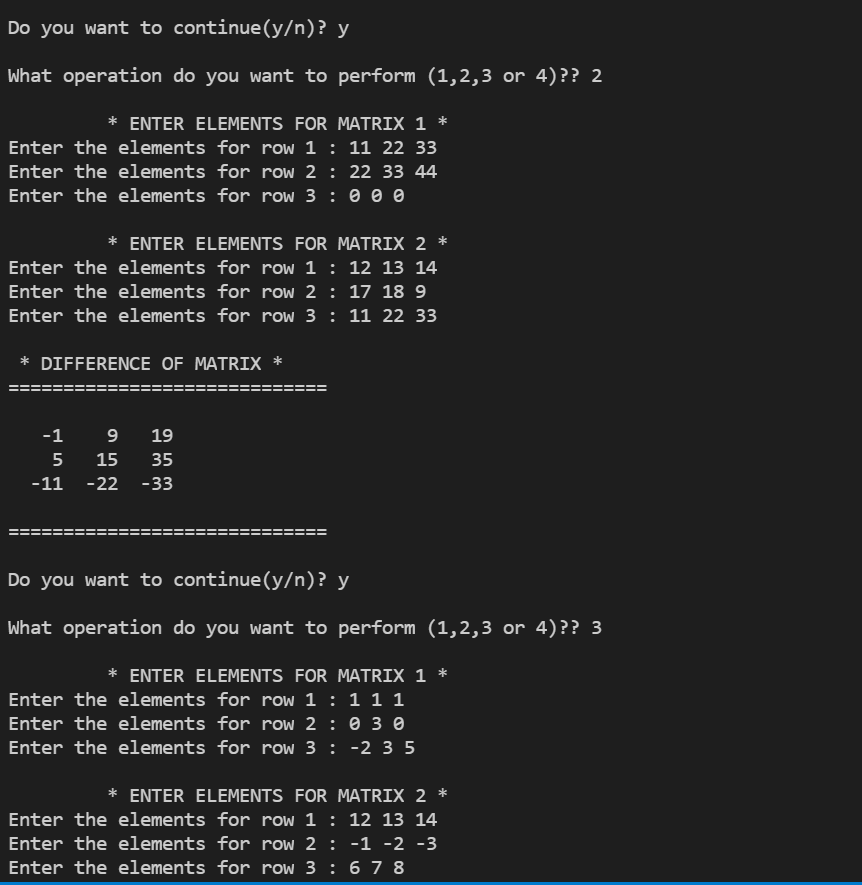
    cin>>ch;

    }

    return 0;

}





**9. Define a class Person having name as a data member. Inherit two classes Student and Employee from Person. Student has additional attributes as course, marks and year and Employee has department and salary. Write display() method in all the three classes to display the corresponding attributes. Provide the necessary methods to show runtime polymorphism.**

#include <iostream>

#include <cstring>

using namespace std;

class Individual

{

public:

    string name;

    void get();

    void display();

};

void Individual::get()

{

    cout << endl

         << "Enter your name: ";

    getline(cin, name);

}

void Individual::display()

{

    cout << "Your name is: ";

    cout << name;

}

class Student : public Individual

{

public:

    int cls;

    int rollno;

public:

    void get();

    void display();

};

void Student::get()

{

    cout << endl

         << "Enter your class: ";

    cin >> cls;

    cout << "Enter your Roll.No.: ";

    cin >> rollno;

}

void Student::display()

{

    cout << endl

         << "Your class is: " << cls << endl;

    cout << "Your roll. No. is: " << rollno << endl;

}

class Employee : public Individual

{

public:

    int salary;

    int id;

public:

    void get();

    void display();

};

void Employee::get()

{

    cout << endl

         << "Enter your salary: ";

    cin >> salary;

    cout << "Enter your employee ID: ";

    cin >> id;

}

void Employee::display()

{

    cout << "Your salary: " << salary << endl;

    cout << "your Employee ID: " << id << endl;

}

int main()

{

    cout << endl

         << "\t \*\* CLASSES AND INHERITANCE \*\*" << endl;

    Individual i1;

    Employee e1;

    Student s1;

    i1.get();

    e1.get();

    e1.display();

    cout << endl;

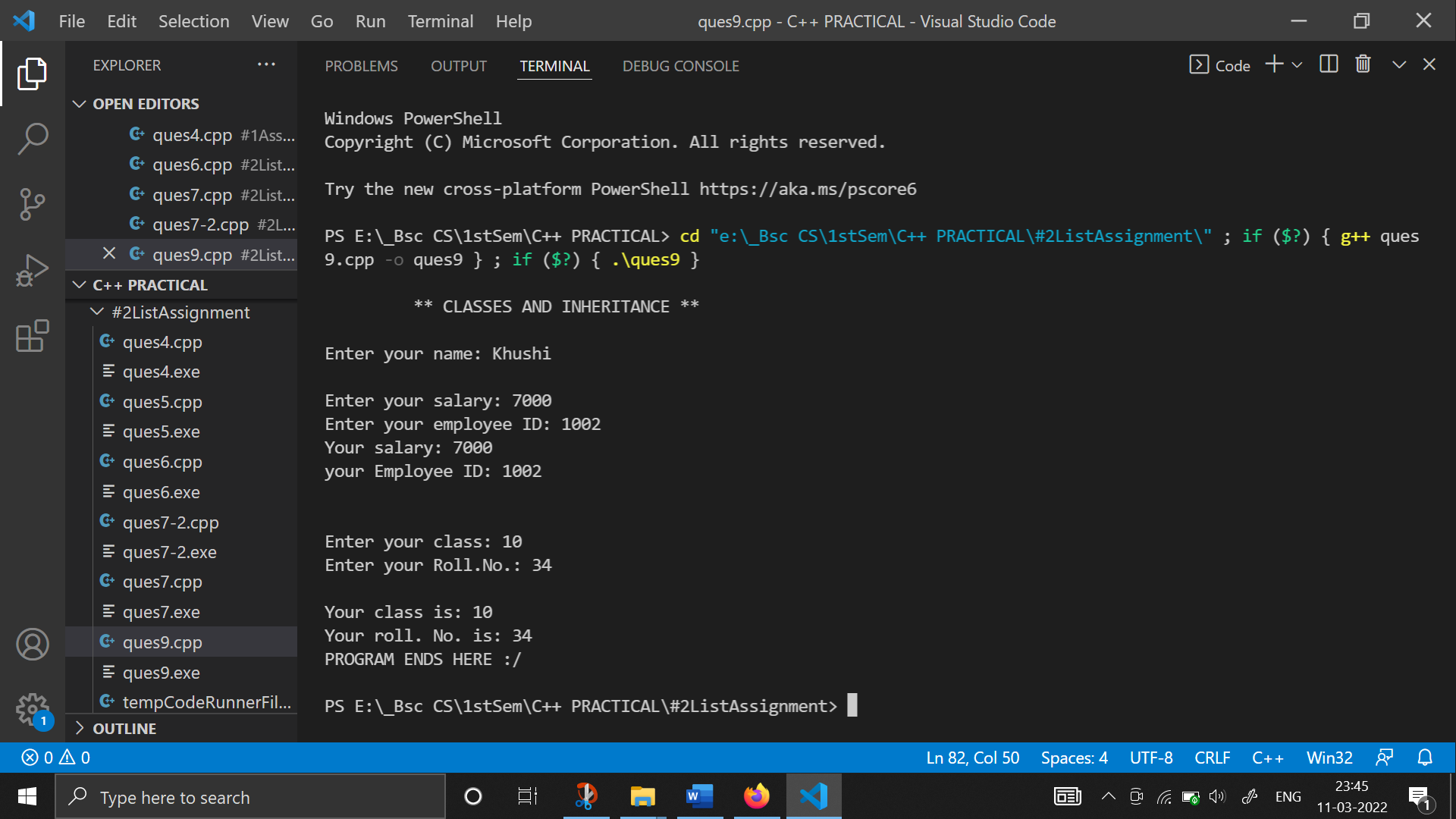
    s1.get();

    s1.display();

    cout << "PROGRAM ENDS HERE :/ " << endl

         << endl;

}



**10. Create a class Triangle. Include overloaded functions for calculating area. Overload assignment operator and equality operator.**

#include <iostream>

#include <math.h>

using namespace std;

class Triangle

{

private:

    float area;

public:

    Triangle()

    {

        area = 0;

    }

    void triangleArea(float a);

    void triangleArea(float a, float b);

    void triangleArea(float a, float b, float c);

    Triangle operator=(Triangle o2);

    int operator==(Triangle o2);

    void display()

    {

        cout << "The area of triangle is: " << area << endl;

    }

};

void Triangle::triangleArea(float a)

{

    area = (sqrt(3) / 4) \* a \* a;

    cout << "Area of equilateral triangle is: " << area << endl;

}

void Triangle::triangleArea(float a, float b)

{

    area = 0.5 \* a \* b;

    cout << "Area of isosceles triangle is: " << area << endl;

}

void Triangle::triangleArea(float a, float b, float c)

{

    float s = (a + b + c) / 2;

    float m = (s - a) \* (s - b) \* (s - c);

    area = sqrt(m);

    cout << "Area of a scalene triangle is: " << area << endl;

}

Triangle Triangle::operator=(Triangle o2)

{

    area = o2.area;

    return o2;

}

int Triangle::operator==(Triangle o2)

{

    if (area == o2.area)

        return 1;

    else

        return 0;

}

int main()

{

    cout<<endl<<"\t \*\* FINDING AREA FOR CLASS TRIANGLE : USING OVERLOADED FUNCTIONS \*\* "<<endl<<endl;

    Triangle o1, o2, o3, o4;

    o1.triangleArea(6);

    o2.triangleArea(8.7, 5.2);

    o3.triangleArea(9, 7, 2);

    o4 = o1;

    o4.display();

    o1.display();

    if (o1 == o4)

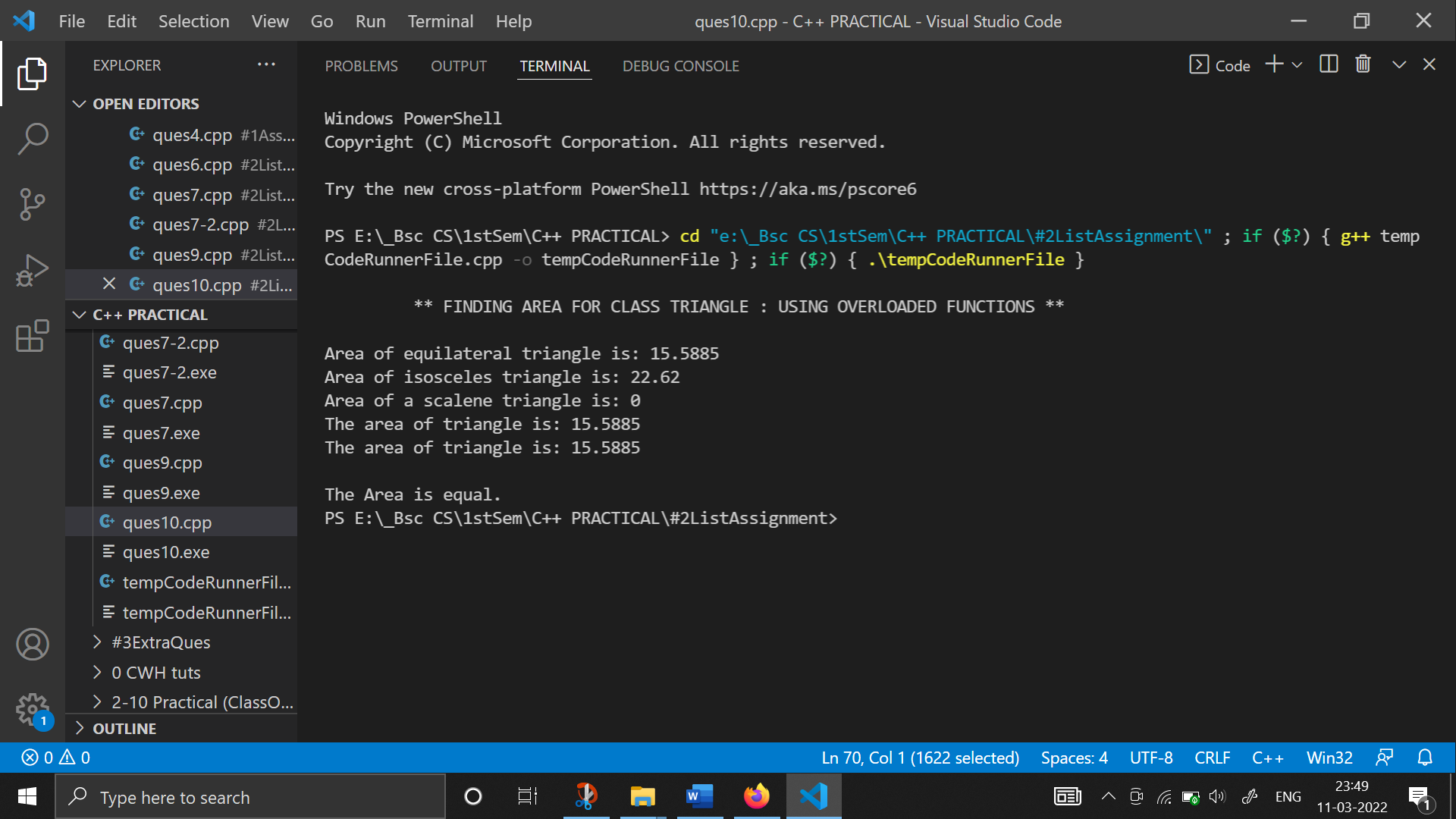
        cout << "\nThe Area is equal. " << endl;

    else

        cout << "\nThe Area is not equal. " << endl;

    return 0;

}



**11. Write a program to read two numbers p and q. If q is 0 then throw an exception else display the result of p/q.**

#include <iostream>

using namespace std;

int main()

{

    double p, q, quo;

    cout << endl

         << "\t \*\* EXCEPTION HANDLING \*\*" << endl;

    cout << endl

         << "Enter the numerator and denominator: ";

    cin >> p >> q;

    try

    {

        if (q == 0)

            throw "Attempted to divide by zero.";

        quo = p / q;

        cout << "The quotient is: " << quo << endl;

    }

    catch (const char \*p)

    {

        cout << p;

    }

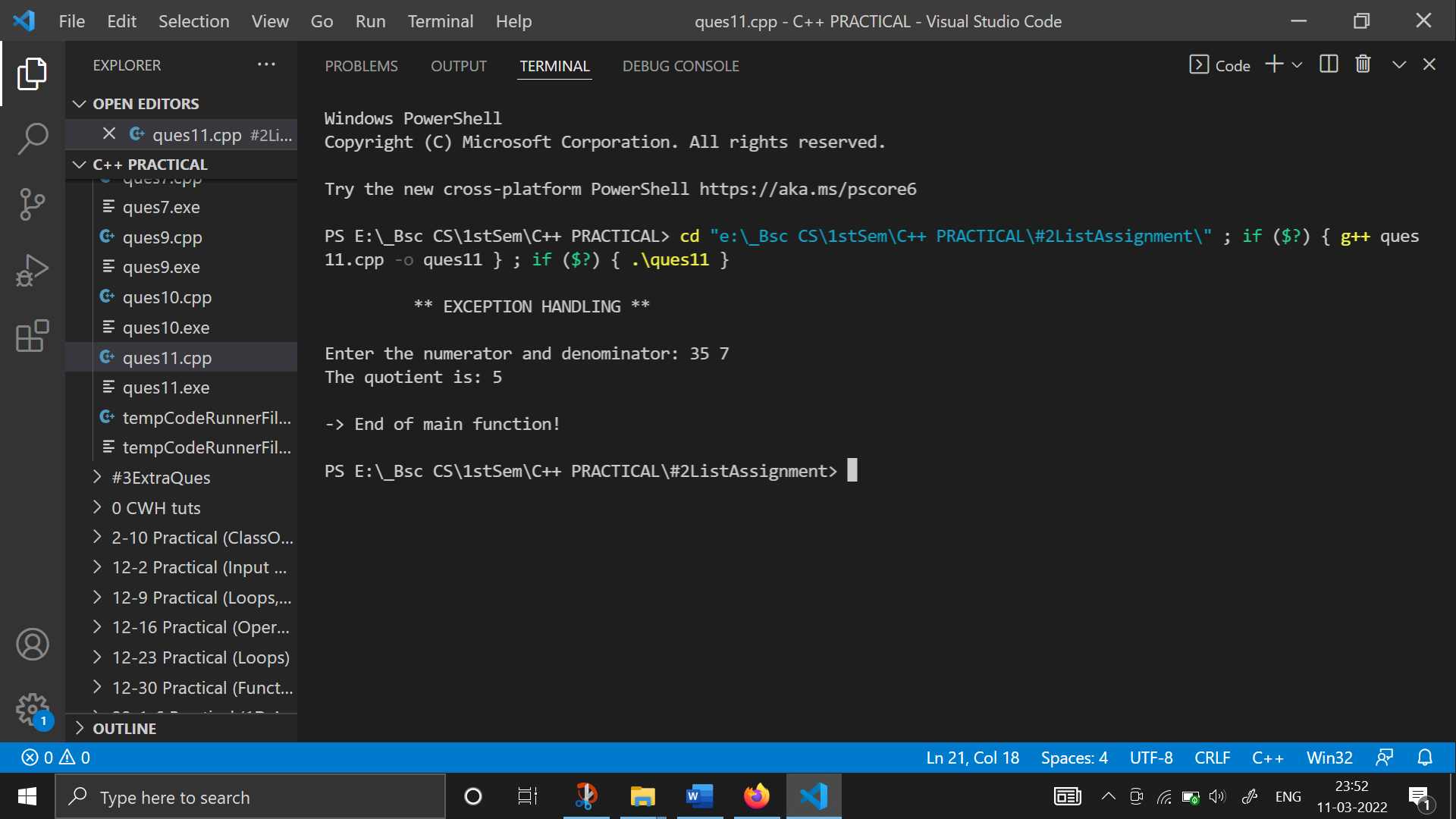
    cout << endl

         << "-> End of main function! " << endl

         << endl;

    return 0;

}



**12. Rewrite Matrix class of Q8 with exception handling. Exceptions should be thrown by the functions if matrices passed to them are incompatible and handled by main() function.**

#include <iostream>

#include <iomanip>

using namespace std;

class matrix

{

    int a[5][5];

    int row, col;

public:

    void input();

    void display();

    matrix operator\*(matrix o1);

    matrix operator+(matrix o1);

    matrix operator-(matrix o1);

    matrix transpose();

    matrix(int row = 5, int col = 5)

    {

        (\*this).row = row;

        (\*this).col = col;

        for (int i = 0; i < row; i++)

            for (int j = 0; j < col; j++)

                a[i][j] = 0;

    }

};

void matrix::input()

{

    cout << endl

         << "\t -> ENTER MATRIX ( nxn )" << endl;

    for (int i = 0; i < row; i++)

    {

        cout << " Enter the elements of row " << (i + 1) << " : ";

        for (int j = 0; j < col; j++)

            cin >> a[i][j];

    }

}

matrix matrix::operator\*(matrix o1)

{

    cout << "\*\* Multiplication of matrix 1st and 2nd \*\*\n";

    matrix o3;

    if (col == o1.row)

    {

        o3.row = row;

        o3.col = o1.col;

        for (int i = 0; i < row; i++)

        {

            for (int j = 0; j < o1.col; j++)

            {

                o3.a[i][j] = 0;

                for (int k = 0; k < col; k++)

                    o3.a[i][j] = o3.a[i][j] + a[i][k] \* o1.a[k][j];

            }

        }

    }

    return o3;

}

matrix matrix::transpose()

{

    matrix temp;

    temp.row = col;

    temp.col = row;

    for (int i = 0; i < temp.row; i++)

        for (int j = 0; j < temp.col; j++)

            temp.a[i][j] = a[j][i];

    return temp;

}

matrix matrix::operator+(matrix o1)

{

    cout << "\t \*\* Addition of 1st and 3rd matrix\*\*\n";

    matrix o6;

    o6.row = row;

    o6.col = o1.col;

    if (row != o1.row || col != o1.col)

        throw "\n Incompatible matrix\n";

    for (int i = 0; i < row; i++)

        for (int j = 0; j < o1.col; j++)

            o6.a[i][j] = a[i][j] + o1.a[i][j];

    return o6;

}

matrix matrix::operator-(matrix o1)

{

    cout << "\t\*\* Subtraction of 1st and 3rd matrix \*\* ";

    matrix o6;

    o6.row = row;

    o6.col = o1.col;

    if (row != o1.row || col != o1.col)

        throw "\nIncompatible matrix\n";

    for (int i = 0; i < row; i++)

        for (int j = 0; j < o1.col; j++)

            o6.a[i][j] = a[i][j] - o1.a[i][j];

    return o6;

}

void matrix::display()

{

    cout << endl

         << "\*\*\* THE RESULTANT MATRIX \*\*\*" << endl;

    for (int i = 0; i < row; i++)

    {

        for (int j = 0; j < col; j++)

            cout << setw(3) << a[i][j];

        cout << endl;

    }

}

int main()

{

    int c, x;

    char ch = 'y';

    matrix o1(3, 4), o2(4, 3), o3(3, 3), o4(4, 3), o5(3, 4), o6(3, 4);

    cout << "\n \*\* MATRIX CLASS (EXCEPTION HANDLING) \*\*\n";

    while (ch == 'y')

    {

        cout << " 1. Input . \n";

        cout << " 2. multiplication of two matrices . \n";

        cout << " 3. transpose of a matrix .\n";

        cout << " 4. Addition of two matrices . \n";

        cout << " 5. Subtraction of two matrices . \n\n";

        cout << " Enter your choice (from 1 to 5): ";

        cin >> x;

        switch (x)

        {

        case 1:

            o1.input();

            o1.display();

            o2.input();

            o2.display();

            o5.input();

            o5.display();

            break;

        case 2:

            try

            {

                o3 = o1 \* o2;

                o3.display();

            }

            catch (const char \*p)

            {

                cout << p;

            }

            break;

        case 3:

            try

            {

                o4 = o1.transpose();

                cout << endl

                     << " The TRANSPOSE of matrix is " << endl;

                o4.display();

            }

            catch (const char \*p)

            {

                cout << p;

            }

            break;

        case 4:

            try

            {

                o6 = o1 + o5;

                o6.display();

            }

            catch (const char \*p)

            {

                cout << p;

            }

            break;

        case 5:

            try

            {

                o6 = o1 - o5;

                o6.display();

            }

            catch (const char \*p)

            {

                cout << p;

            }

            break;

        default:

            cout << " Doesn't find your choice,enter the correct choice ";

            break;

        }

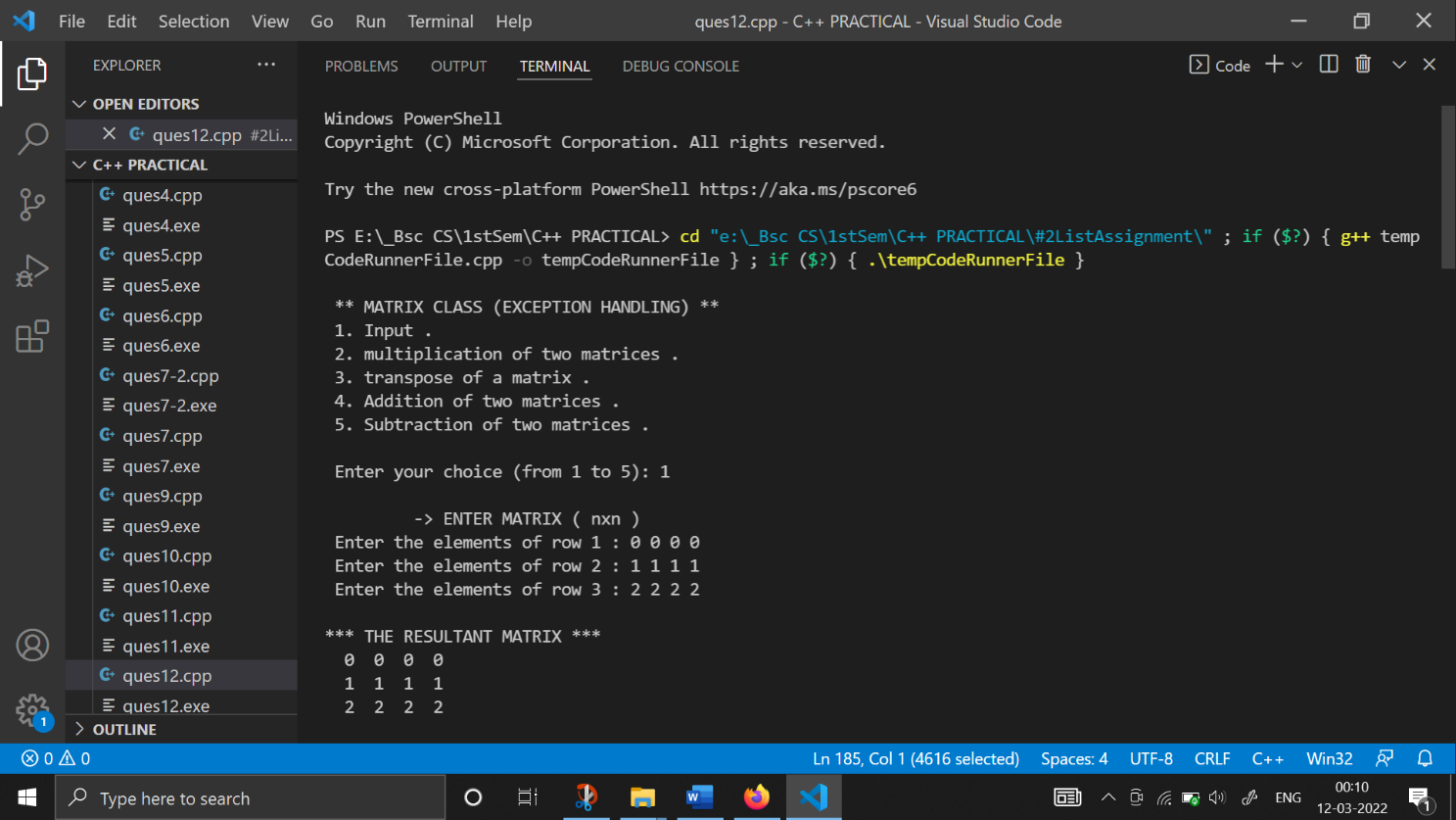
        cout << "DO YOU WANT TO CONTINUE (y/n) ?? ";

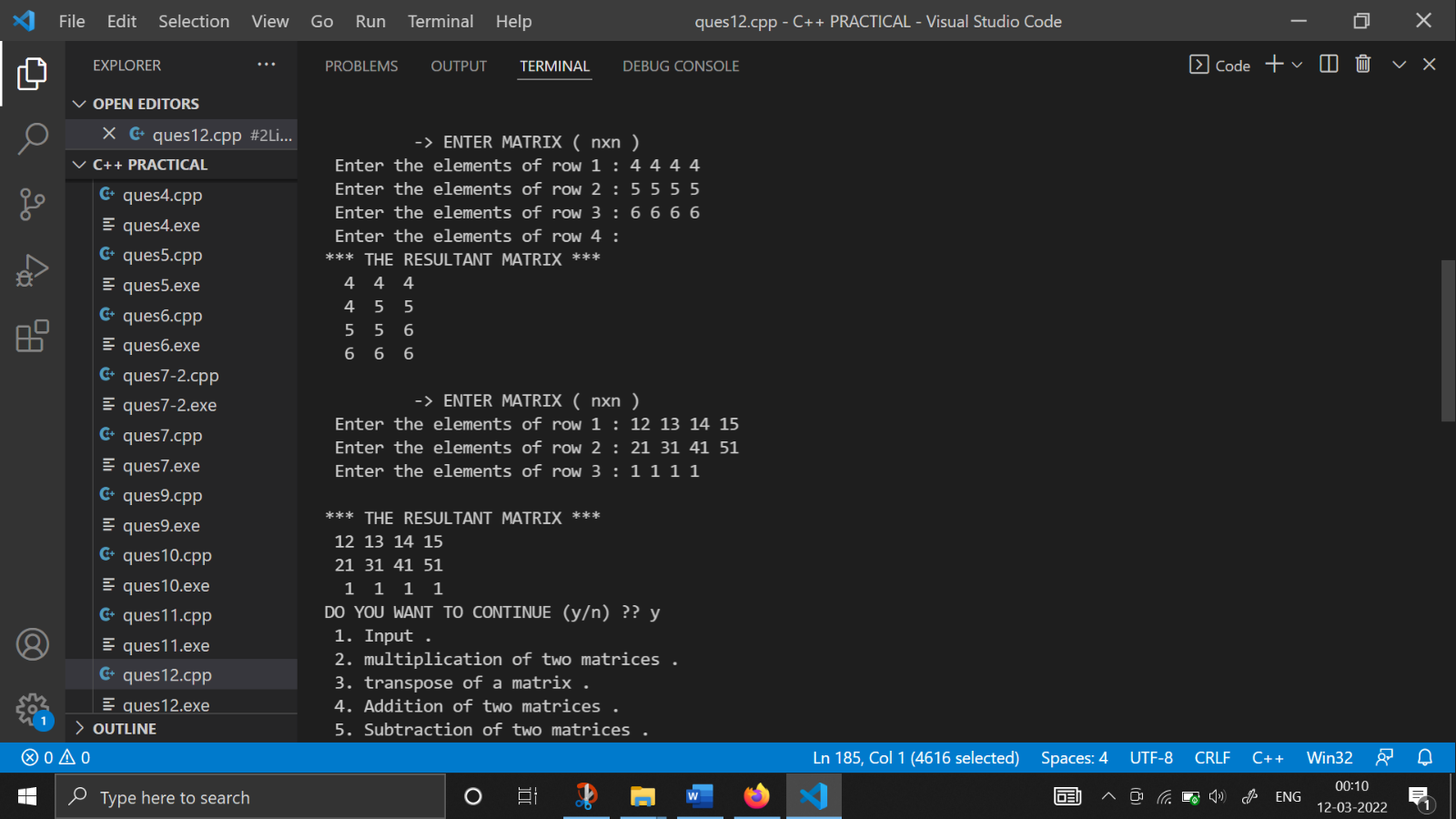
        cin >> ch;

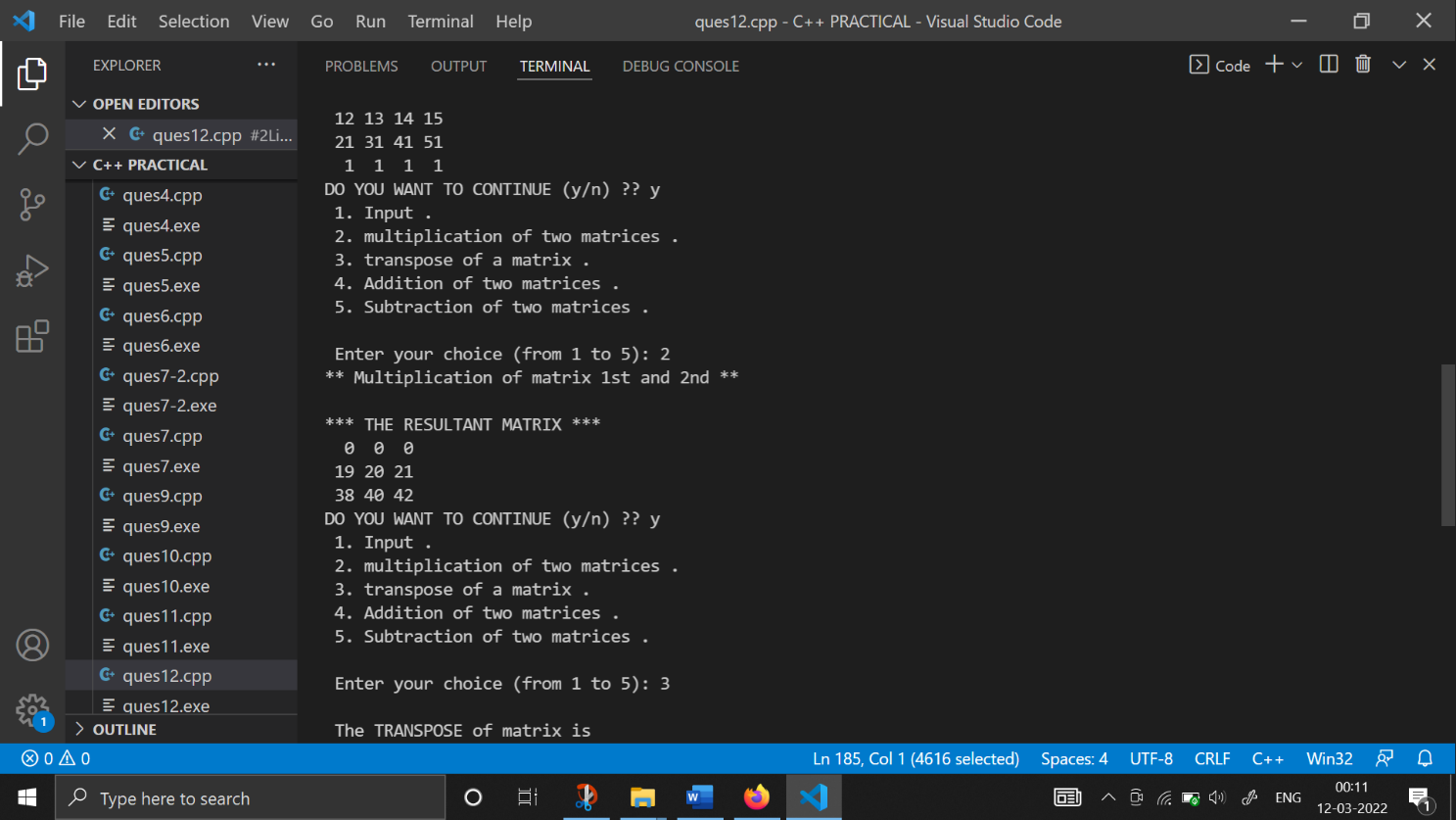
    }

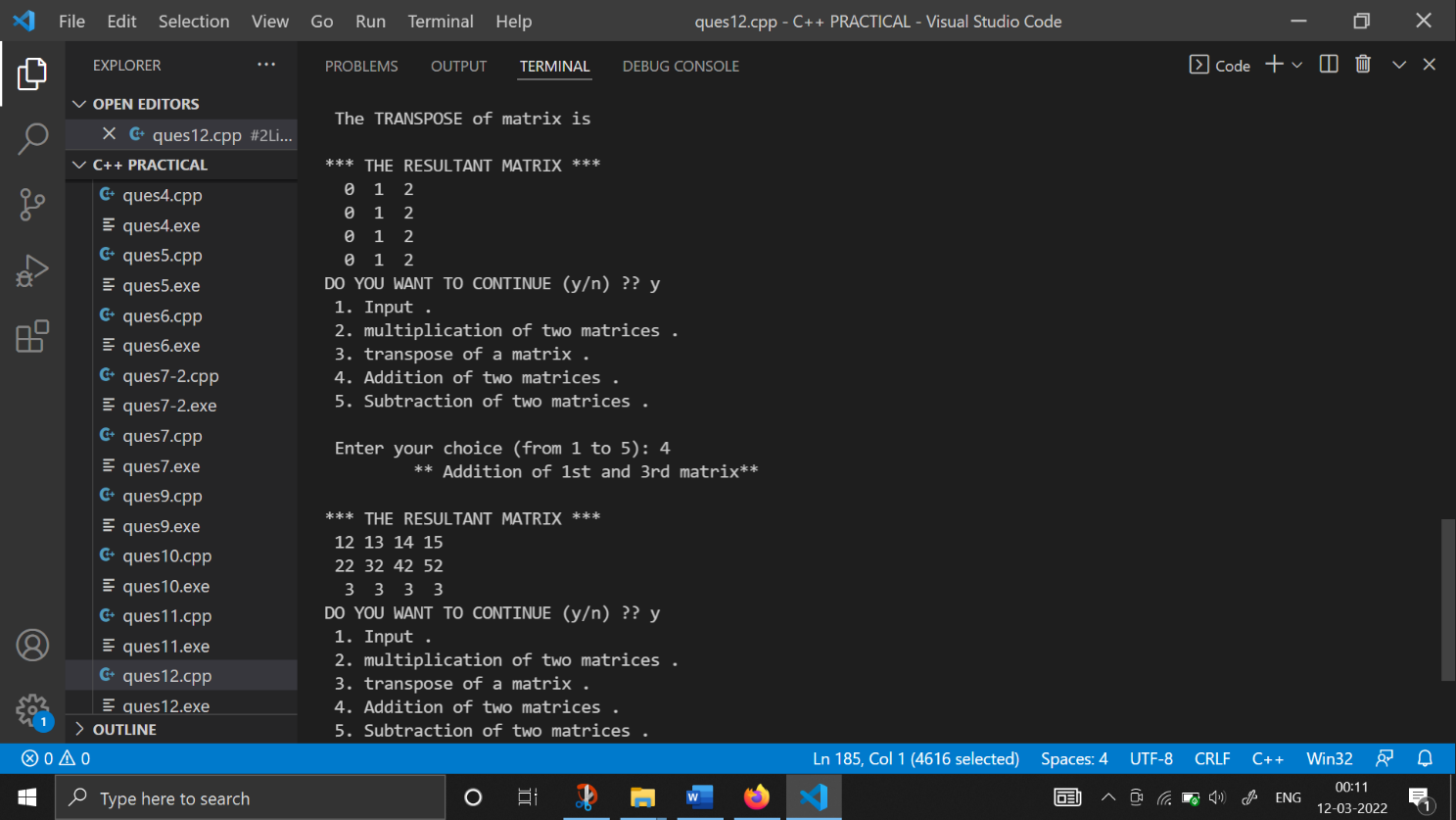
    return 0;

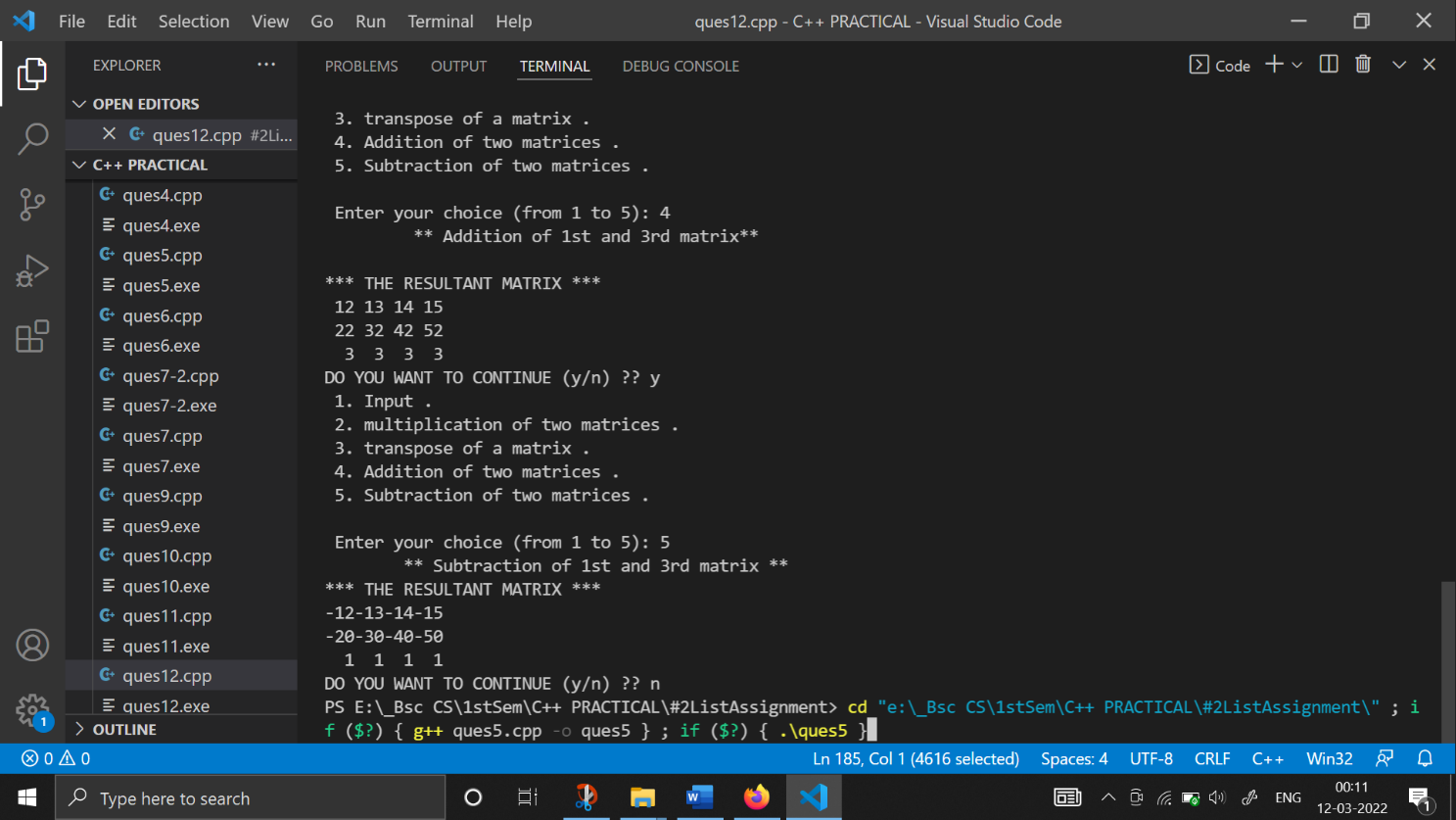
}











**13. Create a class Student containing fields for Roll No., Name, Class, Year and Total Marks. Write a program to store 5 objects of Student class in a file. Retrieve these records from file and display them.**

#include <iostream>

#include <fstream>

using namespace std;

class student

{

    int roll;

    char name[30];

    int Class;

    int year;

    float marks;

public:

    student() {}

    void getData();

    void displayData();

};

void student ::getData()

{

    cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

    cout<<"Enter Roll No. :: ";

    cin >> roll;

    cin.ignore();

    cout << "\nEnter Name :: ";

    cin.getline(name, 30);

    cout << "\nEnter class :: ";

    cin >> Class;

    cout << "\nEnter year :: ";

    cin >> year;

    cout << "\nEnter Marks :: ";

    cin >> marks;

}

void student ::displayData()

{

    cout << "\nRoll No. :: " << roll << endl;

    cout << "\nName :: " << name << endl;

    cout << "\nclass :: " << Class << endl;

    cout << "\nyear :: " << year << endl;

    cout << "\nMarks :: " << marks << endl;

}

int main()

{

    student s[5];

    fstream file;

    int i;

    file.open("C:\\Users\\acer\\Documents\\khushi.txt", ios ::out);

    cout << "\nWriting Student information to the file :- " << endl;

    cout << "\nEnter 5 students Details to the File :- " << endl;

    for (i = 0; i < 5; i++)

    {

        s[i].getData();

        // write the object to a file

        file.write((char \*)&s[i], sizeof(s[i]));

    }

    file.close();

    file.open("C:\\Users\\acer\\Documents\\khushi.txt", ios ::in);

    cout << "\nReading Student information to the file :- " << endl;

    for (i = 0; i < 5; i++)

    {

        // read an object from a file

        file.read((char \*)&s[i], sizeof(s[i]));

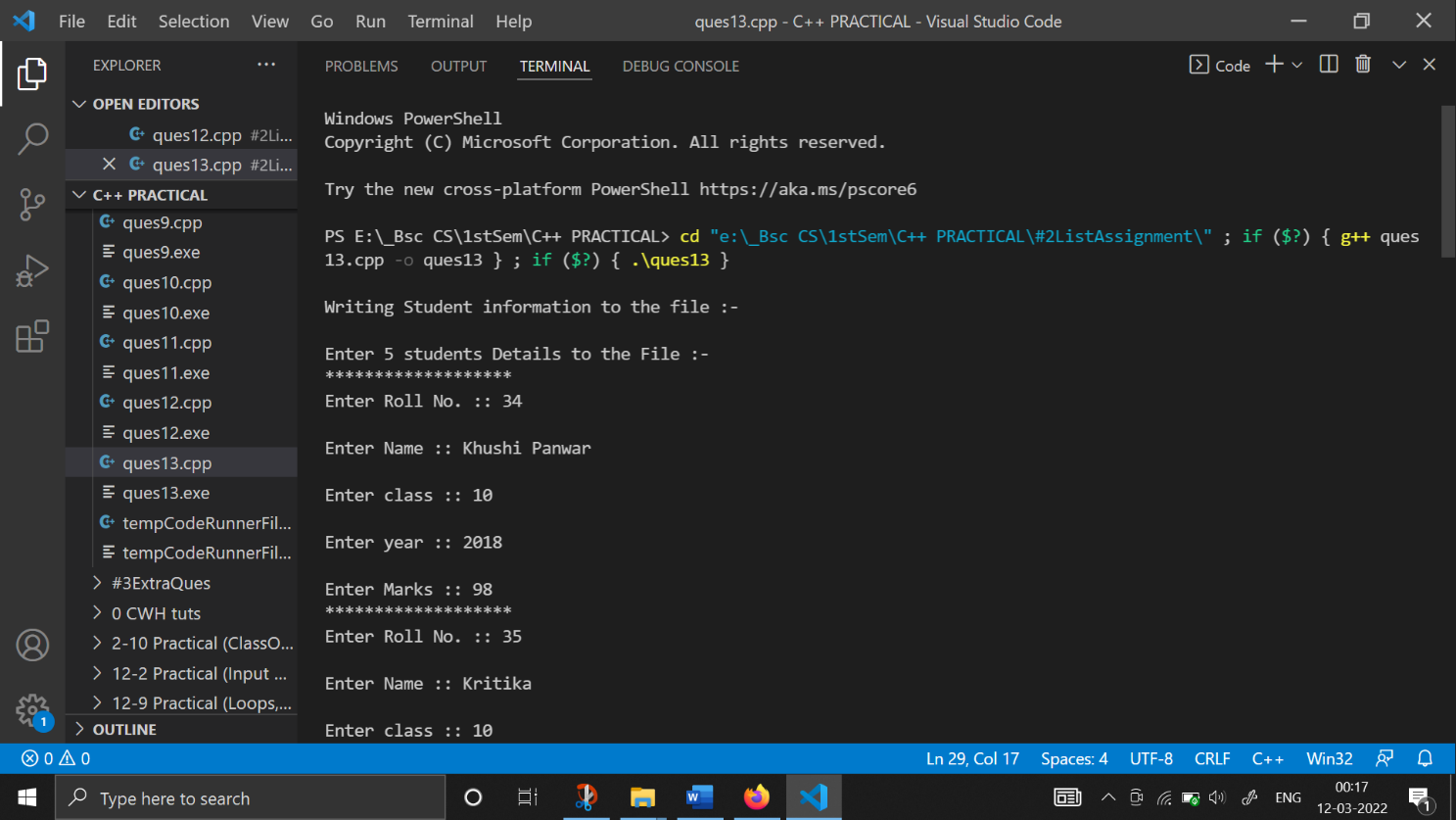
        s[i].displayData();

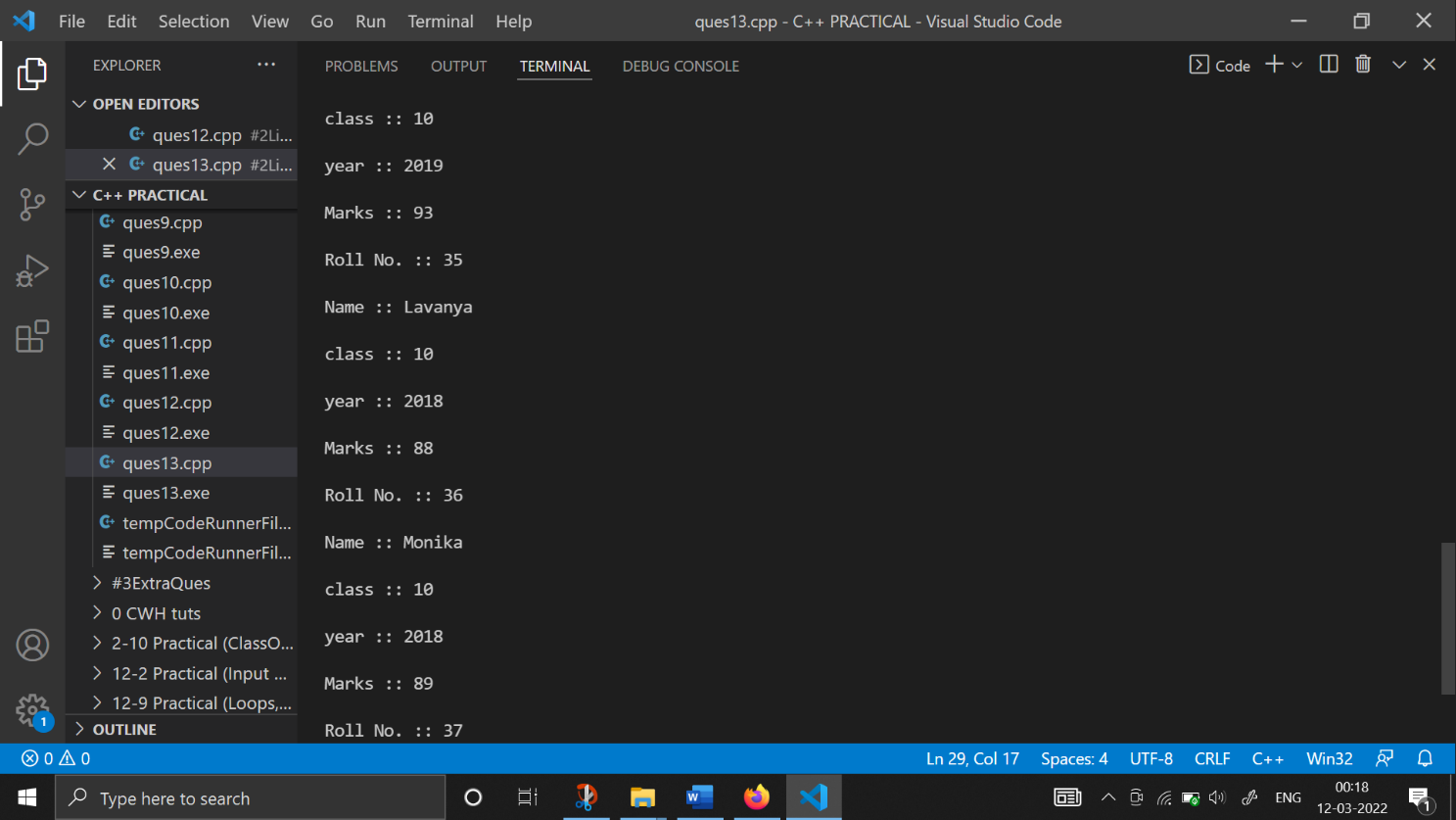
    }

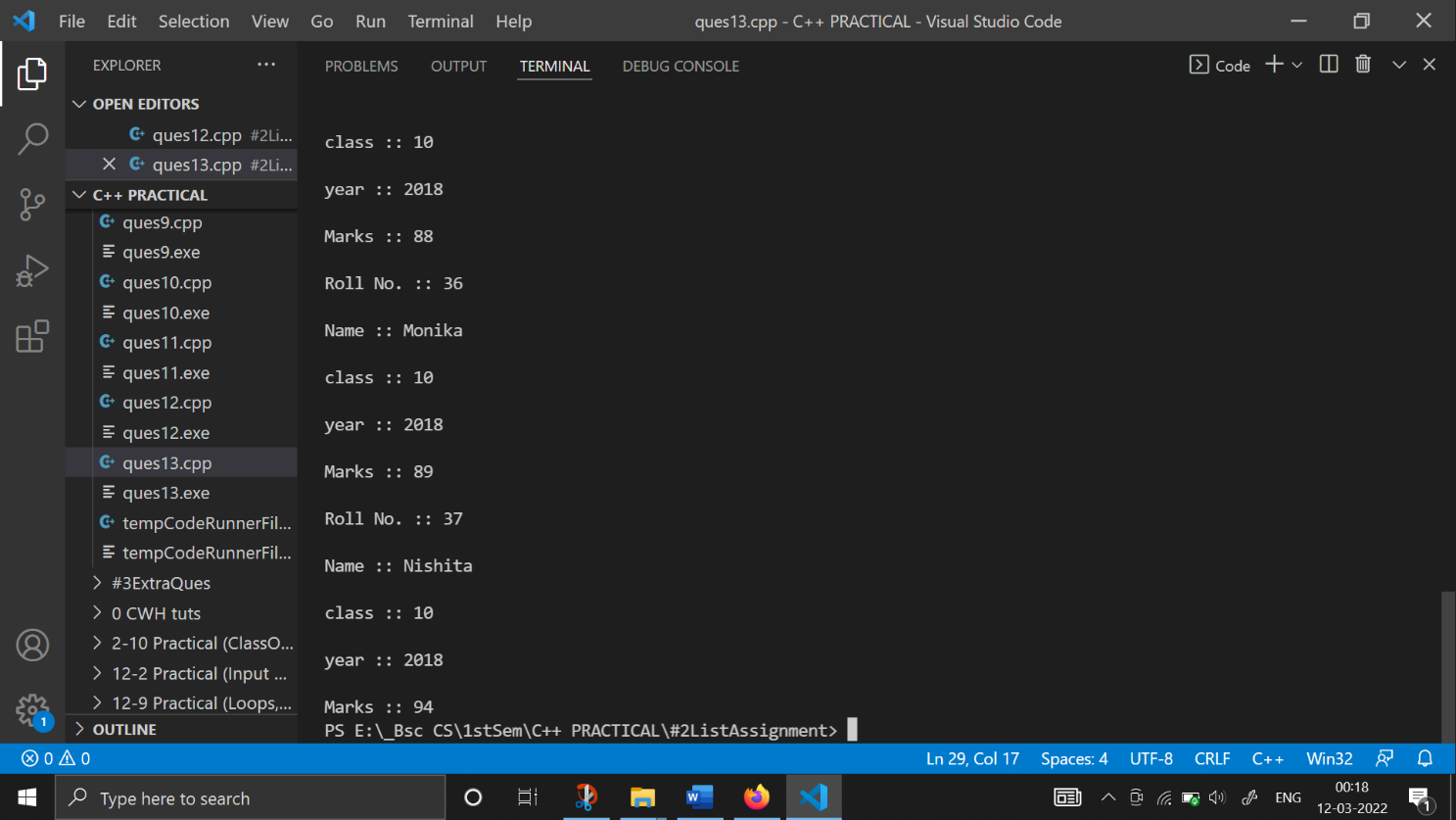
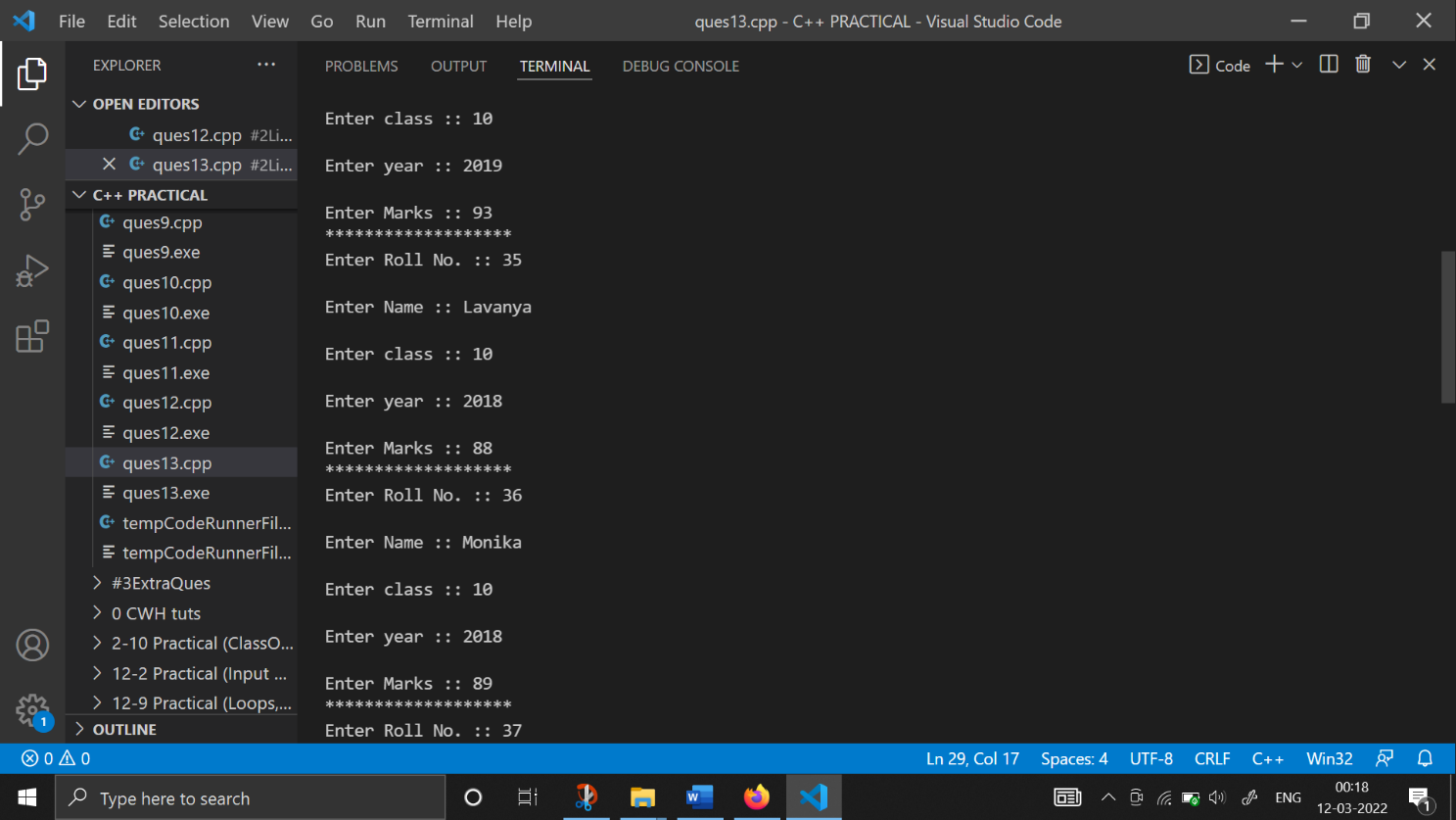
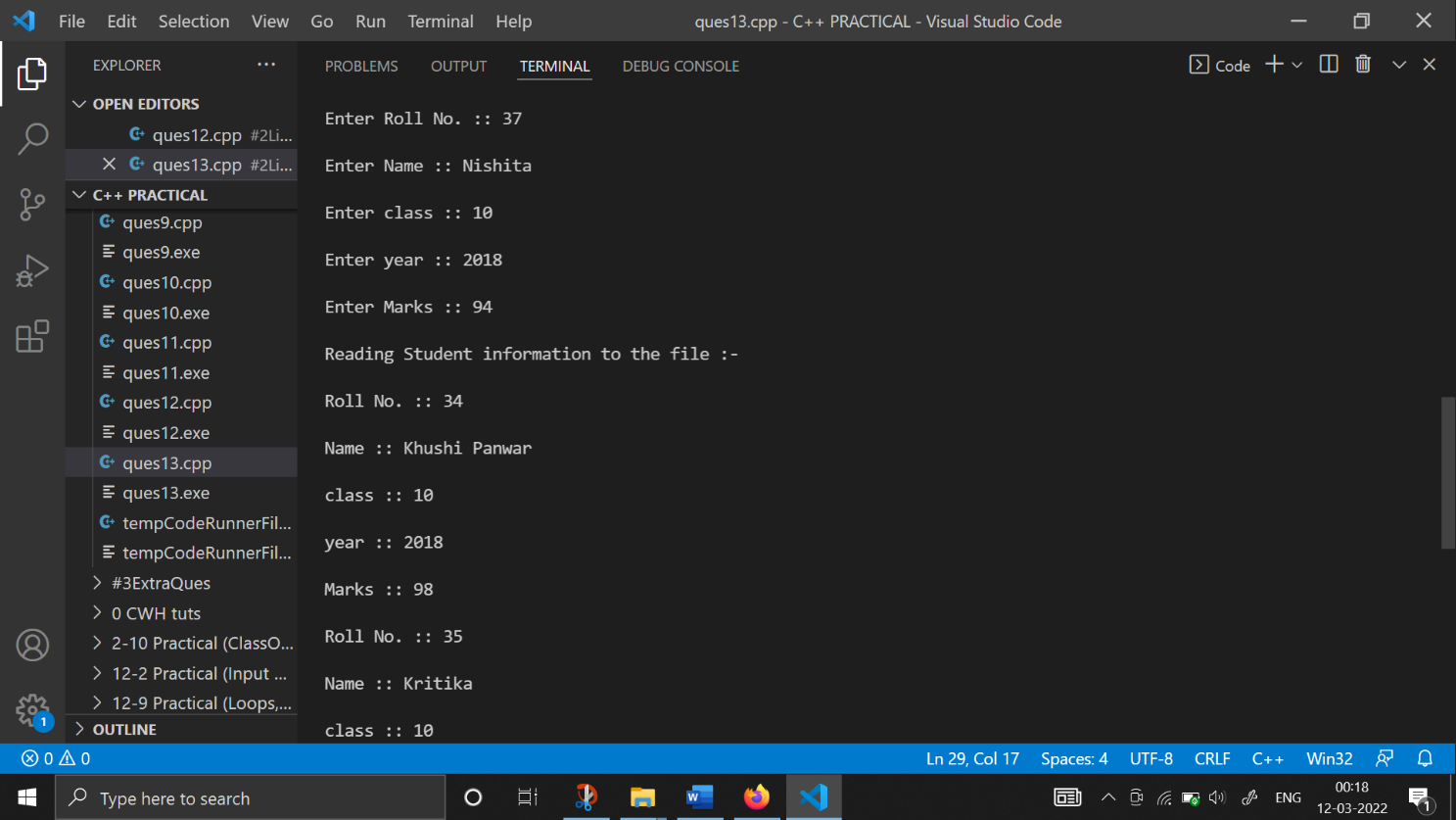
    file.close();

    return 0;

}







**14. Copy the contents of one text file to another file, after removing all whitespaces**

#include <iostream>

#include <fstream>

using namespace std;

int main()

{

    cout <<endl<<"\t \*\* This Program Copies the content of two text files! \*\*" << endl

         << endl;

    ifstream f1;

    ofstream f2;

    char ch;

    f1.open("khushi1.txt");

    if (!f1)

    {

        cout << "error in opening read file";

        exit(99);

    }

    f2.open("khushi2.txt ");

    if (!f2)

    {

        cout << "error in opening a file";

        exit(100);

    }

    cout << "-> Write text\n";

    while (f1.get(ch))

    {

        if (ch != ' ' && ch != '\t' && ch != '\n')

            f2.put(ch);

    }

    f1.close();

    f2.close();

    cout << endl

         << "\tend!" << endl

         << endl;

    return 0;

}

