**NAME – KHUSHI PANWAR**

**ROLL NO – 2021334**

**C++ Assignment Questions**

1. **Define the class Time and declare the following data members**

**hours , minutes and seconds of type integer**

**Define the member functions mentioned as follows**

**a) Parametric and parameterless constructor to initialize the data members of the class.**

**b) Overload>> operator to read the values for the data members of the time class object.**

**c) Overload ++ operator to increment the time class object.**

**d) Overload << operator to display the time class object.**

**e) Overload == to check for the equality of two time class object. The function should return 1 if both time objects are equal otherwise 0.**

#include <iostream>

using namespace std;

class Time{

    public:

    int hours;

    int minutes;

    int seconds;

    Time(int hours, int minutes, int seconds){        ;

    }

    Time(){

        cout<<endl<<"\t \* TIME DETAILS \* "<<endl;

        cout<<"Enter hours : ";

        cin>>hours;

        cout<<"Enter minutes: ";

        cin>>minutes;

        cout<<"Enter seconds: ";

        cin>>seconds;

    }

    friend ostream& operator <<(ostream &output, Time& t){

        cout<<endl<<"\t-> \*USING << OPERATOR TO DISPLAY TIME\* "<<endl;

        output<<t.hours<<"h "<<t.minutes<<"m "<<t.seconds<<"s "<<endl;

        return output;

    }

    friend istream& operator >>(istream &input, Time &t){

        cout<<endl<<"\t-> \*USING >> OPERATOR TO INPUT TIME\* "<<endl;

        cout<<"Enter hours: ";

        input>>t.hours;

        cout<<"Enter minutes: ";

        input>>t.minutes;

        cout<<"Enter seconds: ";

        input>>t.seconds;

        return input;

    }

    Time operator ++(int t2){

        Time t3;

        cout<<endl<<"\t-> \*INCREMENTING TIME AS PER REQUIREMENT\* "<<endl;

        cout<<"Enter the increment digit : ";

        int num = 0;

        cin>>num;

        t3.hours += num;

        t3.minutes += num;

        t3.seconds += num;

        cout<<"Here's the modified time "<<endl;

        cout<<t3.hours<<"h "<<t3.minutes<<"m "<<t3.seconds<<"s "<<endl;

        return t3;

    }

    friend Time operator ==(Time &t, Time& t2){

        cout<<endl<<"\t-> \*USING == OPERATOR TO COMPARE TIME VALUES\* "<<endl;

        cout<<"\*\*Enter the details for Time 1 -> ";

        Time t4;

        cout<<"\*\*Enter the details for Time 2 -> ";

        Time t5;

        int flag1 = 1;

        int flag2 = 1;

        int flag3 = 1;

        if(t4.hours != t5.hours) flag1 = 0;

        if(t4.minutes != t5.minutes) flag2 = 0;

        if(t4.seconds != t5.seconds) flag3 = 0;

        if(flag1 != 0 && flag2 != 0 && flag3 != 0){

            cout<<endl<<"Both Times are equal!"<<endl;

        }

        else cout<<endl<<"Times aren't matching"<<endl;

        return t4;

    }

};

int main(){

    Time t1;

    cin>>t1;

    cout<<t1;

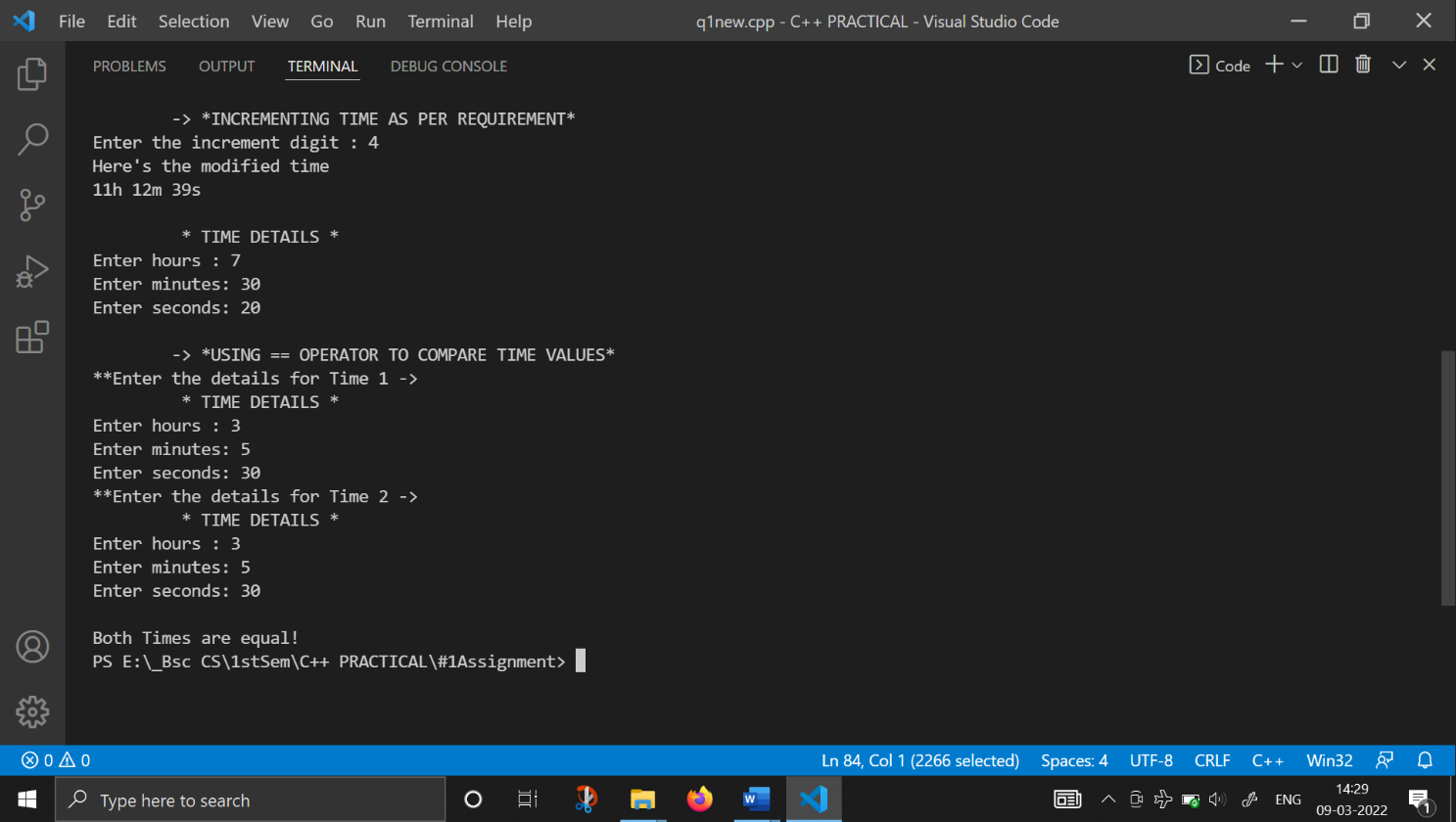
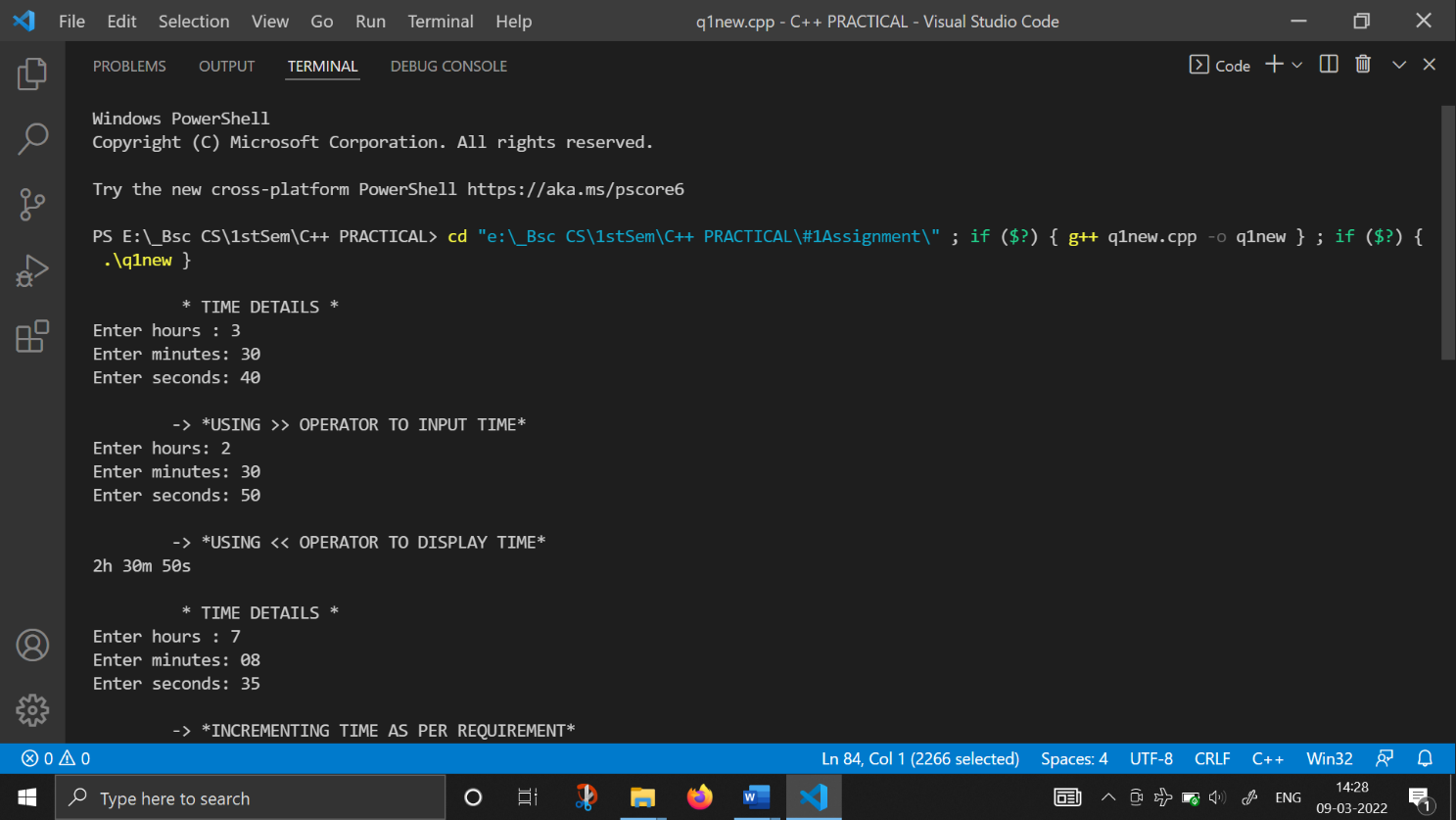
    t1++;

    Time t2;

    t1 == t2;

    return 0;

}



1. **Write a program in C++ that defines a 2D array A of characters. The array has a size m\*n where m is the number of rows and n is the number of columns. m and n should be declared as constants.**

**Define the functions with the following prototypes in the program.**

**• void read(char A[][n], int m1): This function will read an array of m1 strings and stores them in the array A .**

**• void countvowels(char A[][n],int m1): This function will display the count of vowels in the array A .**

**• int find(char A[][n],int m1, char b[n]):This function will search a string b in the array A . It will return 1 if the string b is found in the array A else it returns 0.**

**• void display(char A[][n], int m1 ):This function will display the array of strings A.**

#include<iostream>

#include<string.h>

using namespace std;

void read(char A[][10],int m1)

{       cout<<"\n \t-> Enter the string with only 10 characters : "<<endl;

    for(int i=0;i<m1;i++)

    {   cout<<" Enter string "<<i+1<<": ";

        for(int j=0;j<10;j++)

            cin>>A[i][j];

    }

}

void countvowels(char A[][10], int m1)

{   int countvowel=0;

    for(int i=0;i<m1;i++)

    {

        for(int j=0;j<10;j++)

        {

            if(A[i][j]=='A'||A[i][j]=='E'||A[i][j]=='I'||A[i][j]=='O'||A[i][j]=='U'||

               A[i][j]=='a'||A[i][j]=='e'||A[i][j]=='i'||A[i][j]=='o'||A[i][j]=='u')

            {

                countvowel++;

            }

        }

    }

    cout<<"\n Vowel count in Array A= "<<countvowel<<endl;;

}

int find(char A[][10],int m1,char b[10])

{

    for(int i=0;i<m1;i++)

    {

        for(int j=0;j<10;j++)

        {

        if(b[j]==A[i][j])

            return 1;

        }

    }

    return 0;

}

void display(char A[][10],int m1)

{

    for(int i=0;i<m1;i++)

    {   cout<<"\n String "<<i+1<<"= ";

        for(int j=0;j<10;j++)

           {

               cout<<A[i][j];

           }

            cout<<endl;

    }

}

int main()

{

    char A[10][10],b[10]; int m,R;

    cout<<"\n How many elements do you want to enter?(less than 10) : ";

    cin>>m;

    read(A,m);

    cout<<"\n Enter the string you want to search(less than 10) : ";

    for(int j=0;j<10;j++)

    cin>>b[j];

    R=find(A,m,b);

    if(R=1)

    {

        cout<<"\n String is found in 2D Array\n";

    }

    else cout<<"\n String is not found in the 2D Array\n";

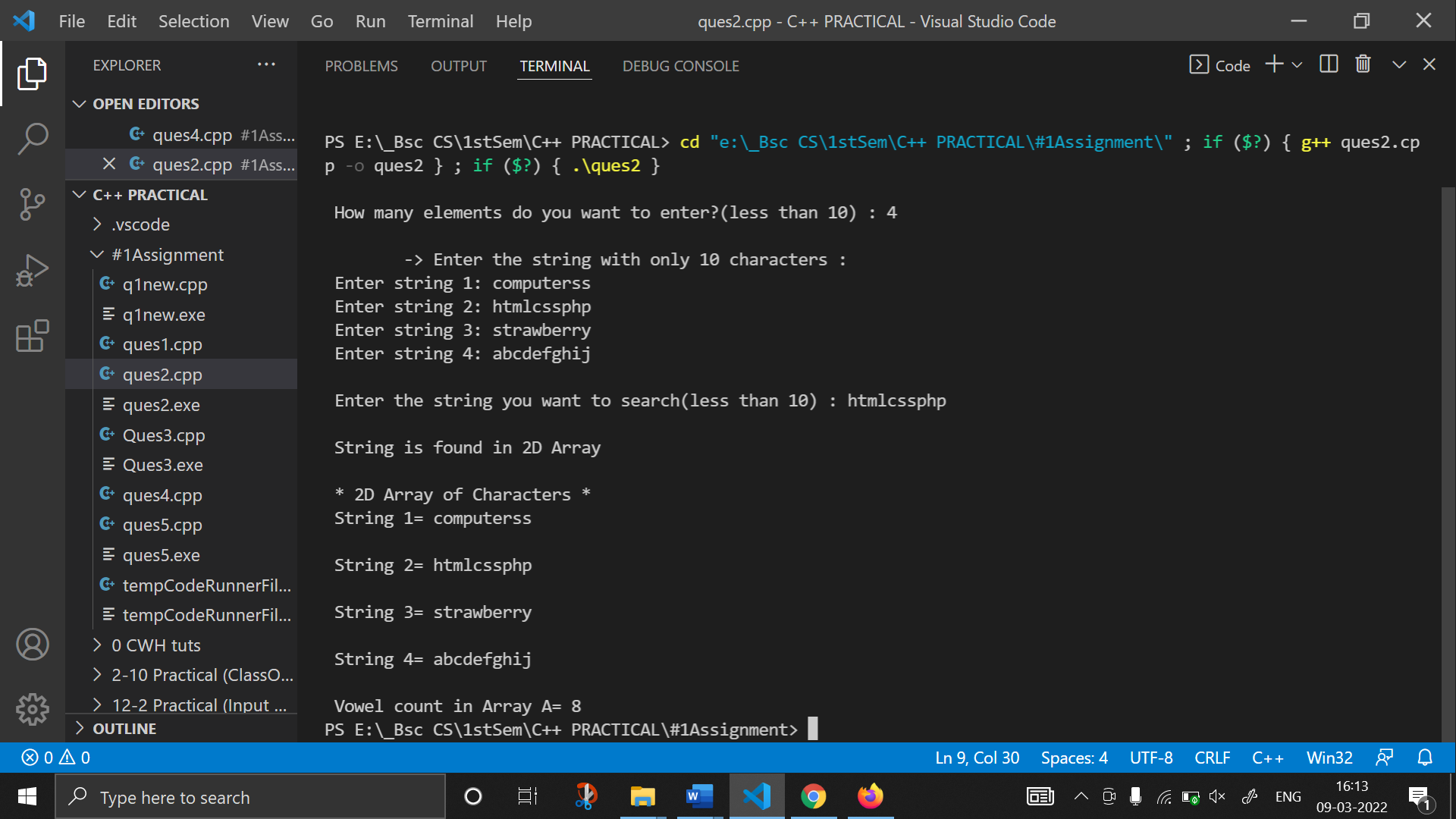
    cout<<"\n \* 2D Array of Characters \*";

    display(A,m);

    countvowels(A,m);

    return 0;

}



1. **Define a class Employee with data members – Name, Organization, Qualification and Salary. In this class define member functions as given below:**

**• parameterless constructor to initialize the data members - Name, Organization, Qualification and Salary.**

**• A function disp() for displaying the Name, Organization, Qualification and Salary.**

**• A pure virtual function void print().**

**Derive the class Faculty from the Employee class using public inheritance. In the Faculty class declare data members: Course and Workload(number of working hours of a faculty in**

**a week) of the appropriate data type and define a parametric constructor for initializing these data members. Override the print() function in Faculty class to display the Course and**

**Workload .**

**Derive a class Staff from the Employee class using public inheritance. This derived class declares data members -Designation and Job Description of the appropriate data type.**

**Define the parametric constructor of this class to initialize the values of Designation and Job Description. Override the print function in this class to display Designation and Job Description.**

**Define the main() function and declare one object each of Faculty class and Staff class.**

**Use run time polymorphism and display the details of Faculty and Staff class objects.**

 #include <iostream>

#include <string>

using namespace std;

class Employee{

    public:

    string name;

    string organisation;

    string qualification;

    int salary;

    Employee(){

        cout<<endl<<"\t -> \* EMPLOYEE DETAILS \*"<<endl<<endl;

        cout<<"Enter Employee name: "<<endl;

        cin>> name;

        cout<<"Enter organisation: "<<endl;

        cin>>organisation;

        cout<<"Enter qualification: "<<endl;

        cin>>qualification;

    }

    void disp(){

        cout<<"The Name of the employee is: "<<name<<endl;

        cout<<"The Employee's organisation: "<<organisation<<endl;

        cout<<"The Qualification of the employee is: "<<qualification<<endl;

    }

    virtual void print(){

        cout<<"hello";

    }

};

class Faculty:public Employee{

    public:

    string course;

    int workload;

    Faculty(string course1, int workload1){

        course = course1;

        workload = workload1;

    }

    void print(){

        cout<<"Course: "<<course<<endl;

        cout<<"Workload: "<<workload<<endl;

    }

};

class Staff: public Employee{

    public:

    string designation;

    string job\_description;

    Staff(string designation, string job\_description){

        cout<<" > Enter designation: ";

        cin>>designation;

        cout<<" > Enter job\_description: ";

        cin>>job\_description;

    }

    void print(){

        cout<<"Designation: "<<endl;

        cin>>designation;

    }

};

int main(){

    int workload;

    string course;

    string designation;

    string job\_description;

    cout<<endl<<"\t \* EMPLOYEE DETAILS \*"<<endl<<endl;

    cout<<"Enter course: ";

    cin>>course;

    cout<<"Enter workload: ";

    cin>> workload;

     cout<<"Enter Designation: ";

    cin>>designation;

    cout<<"Enter job description: ";

    cin>>job\_description;

    Faculty f(course, workload);

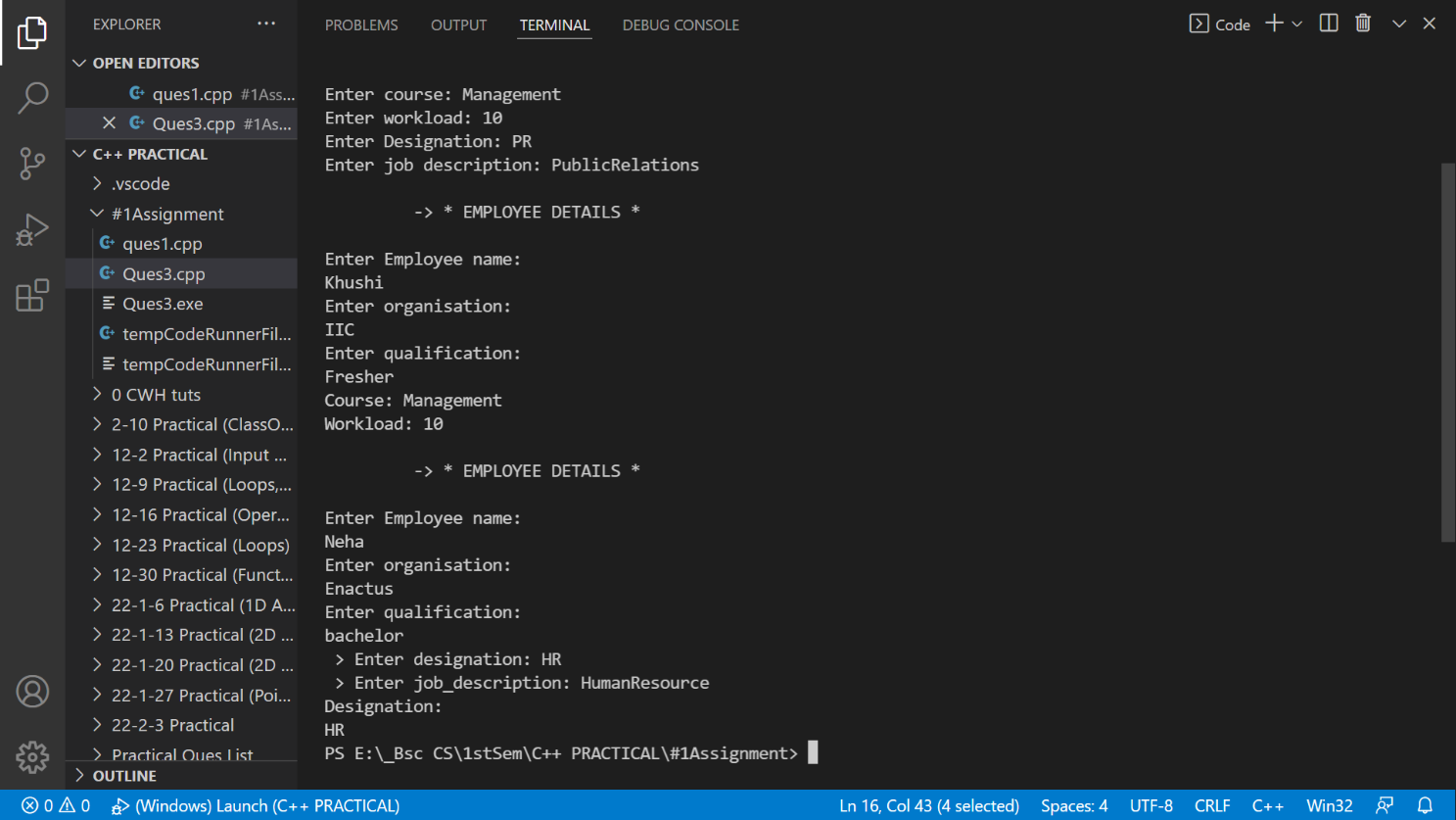
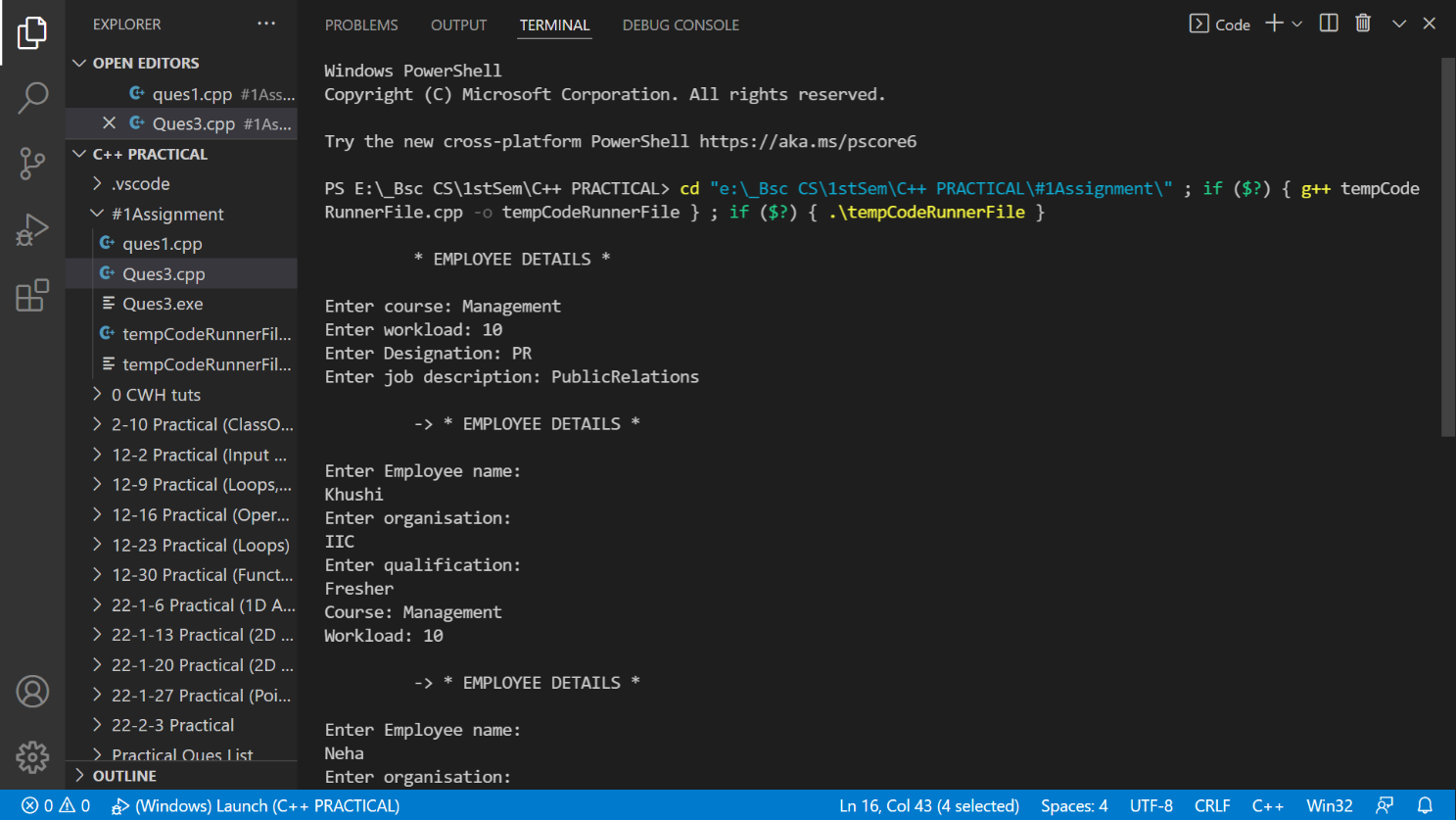
    f.print();

    Staff s(designation, job\_description);

    s.print();

    return 0;

}



**4. Write C++ statements for the following tasks:**

**a. Write the prototype of a function funP that accepts two arguments- a constant one dimensional array of integers and a float; and returns a pointer to a double.**

double \* funP(int array[], float a);

**b. Accept three strings through command line arguments. If the number of integers entered is less than or greater than three, the program exits after displaying the message "Wrong number of inputs!"**

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

{

    if(argc>3|| argc<3){

        cout<<"\nWrong number of input";

        return 0;

    }

**c. Interchange values of two integer variables using pointers.**

int\*p,\*q,\*temp;

temp=p;

p=q;

q=temp;

**d. An anonymous object of a derived class Derv is to be pointed to by a reference of its base class Base.**

int main()

{

Base \* base;

base=&Derv;

return 0;

}

**e. Write the prototype of a friend function funObjs(...) having references to two objects of a class MyClassOne as parameters and an object of a class MyClassTwo as return type.**

friend MyClassTwo funObjs( myclassone & obj1,myclassone & obj2 );

1. **Write a C++ program that prompts a user for three integers- the first denoting a month (1 to 12), the second denoting a day (1 to 31) and the third denoting a year. The output is displayed as "month day, year" string where month represents the name of the month.**

**For example, if inputs are 12, 15 and 2020 respectively, the output is December 15, 2020.**

**A parameterized function convertDate(...) is defined to accept above three integers as arguments and return the appropriate string.**

**If the user enters any number other than a valid month number (integers from 1 to 12) as a first input, then the program throws an error (exception) of the type string. Write an exception handler that handles the exception by displaying a message "Not a valid month" and exits.**

**If the user enters any number other than a valid day number (integers from 1 to either 28, 30, or 31, depending on the month) as second input, then the program throws an error**

**(exception) of the type integer. Write an exception handler that handles the exception by displaying a message "Not a valid day" and exits.**

**If the user enters month value 2, day value 29 and a non-leap year as third input, then the program throws an error (exception) of the type double. Write an exception handler that handles the exception by displaying a message "Not a leap year" and exits.**

#include <iostream>

using namespace std;

int convertDate(int month, int day, int year)

{

    cout<<endl<<"-> The converted date is as follows : ";

    switch (month)

    {

    case 1:

        cout << "JANUARY ";

        break;

    case 2:

        cout << "FEBRUARY ";

        break;

    case 3:

        cout << "MARCH ";

        break;

    case 4:

        cout << "APRIL ";

        break;

    case 5:

        cout << "MAY ";

        break;

    case 6:

        cout << "JUNE ";

        break;

    case 7:

        cout << "JULY ";

        break;

    case 8:

        cout << "AUGUST ";

        break;

    case 9:

        cout << "SEPTEMBER";

        break;

    case 10:

        cout << "OCTOBER ";

        break;

    case 11:

        cout << "NOVEMBER ";

        break;

    case 12:

        cout << "DECEMBER ";

        break;

    }

    cout << year << " , ";

    cout <<' '<<day<< ' '<<endl;

}

int main()

{

    int month, day, year;

    cout<<"\t\* DATE-MONTH-YEAR PROGRAM \*"<<endl<<endl;

    try

    {

        cout << "Enter the month (in digits) : ";

        cin >> month;

        char m=month;

        char y=year;

        char d=day;

        if (m < 1 || m > 12)

        {

            throw("Error");

        }

        cout << "Enter day : ";

        cin >> day;

        if (((m == 1 || m == 3 || m == 5 || m == 7 || m == 8 || m == 10 || m == 12) && (d > 31)) || ((m == 4 || m == 6 || m == 9 || m == 11) && (d > 30)) || ((m == 2) && (d > 29)))

        {

            throw("Error");

        }

        cout << "Enter the year : ";

        cin >> year;

        if (m == 2 & d == 29 && ((y % 4 == 0) && (y % 100 != 0) || (y % 400 == 0)))

        {

            throw("Error:Not a leap year");

        }

        convertDate(month, year, day);

    }

    catch (const char msg)

    {

        cout << msg << endl;

        exit(0);

    }

}

