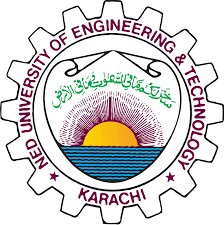
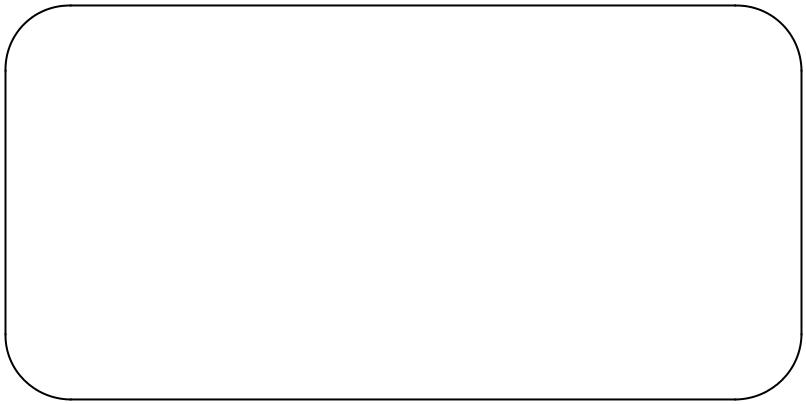
Lab Manual

SE-201

****

Object Oriented Concepts & Programming

Spring 2020

Name : \_NIMRAH ALTAF ADAM\_\_\_\_\_\_\_

Year : \_\_\_ 2019\_\_\_\_\_\_\_\_\_\_\_\_\_

Batch : \_\_\_\_\_\_\_\_FESE\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Roll No : \_\_\_\_\_\_\_\_\_77\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

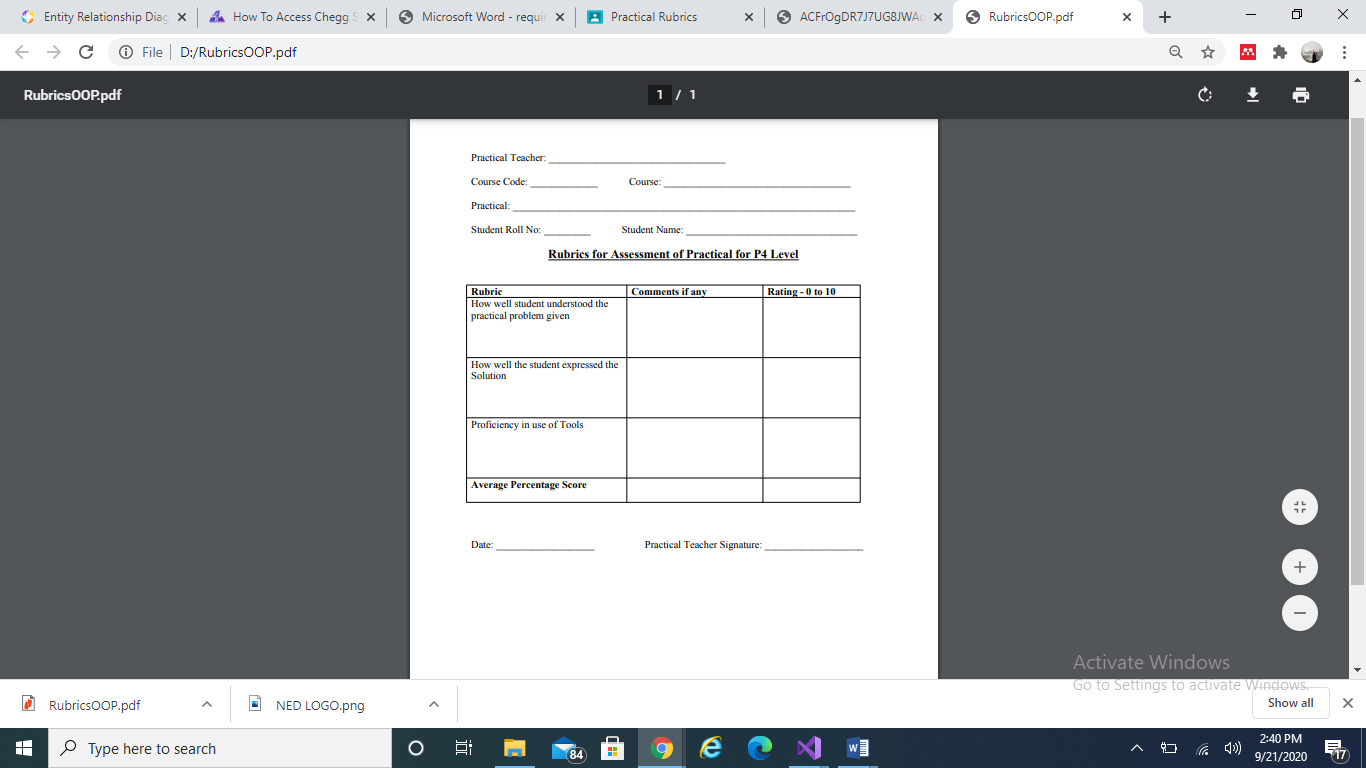
Department: \_SOFTWARE ENGINEERING \_\_

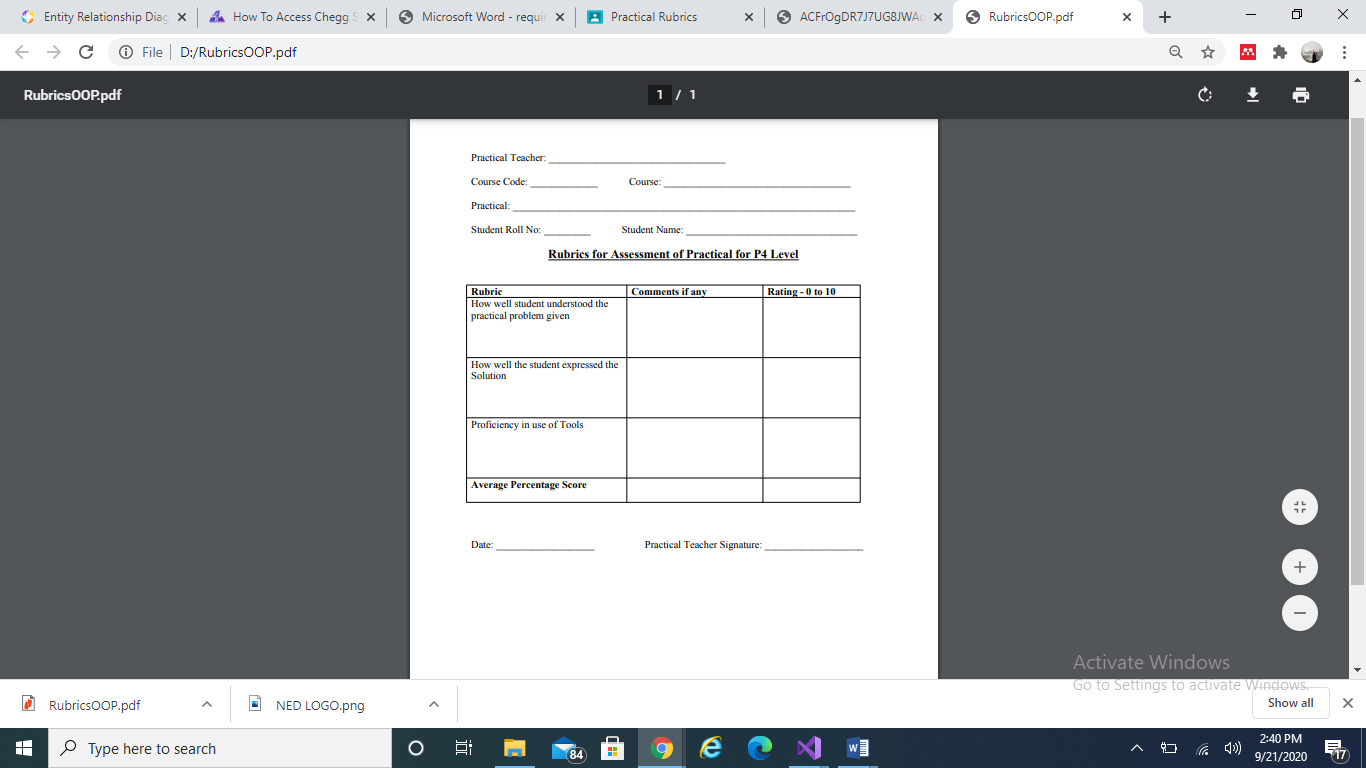
**Department of Software Engineering**

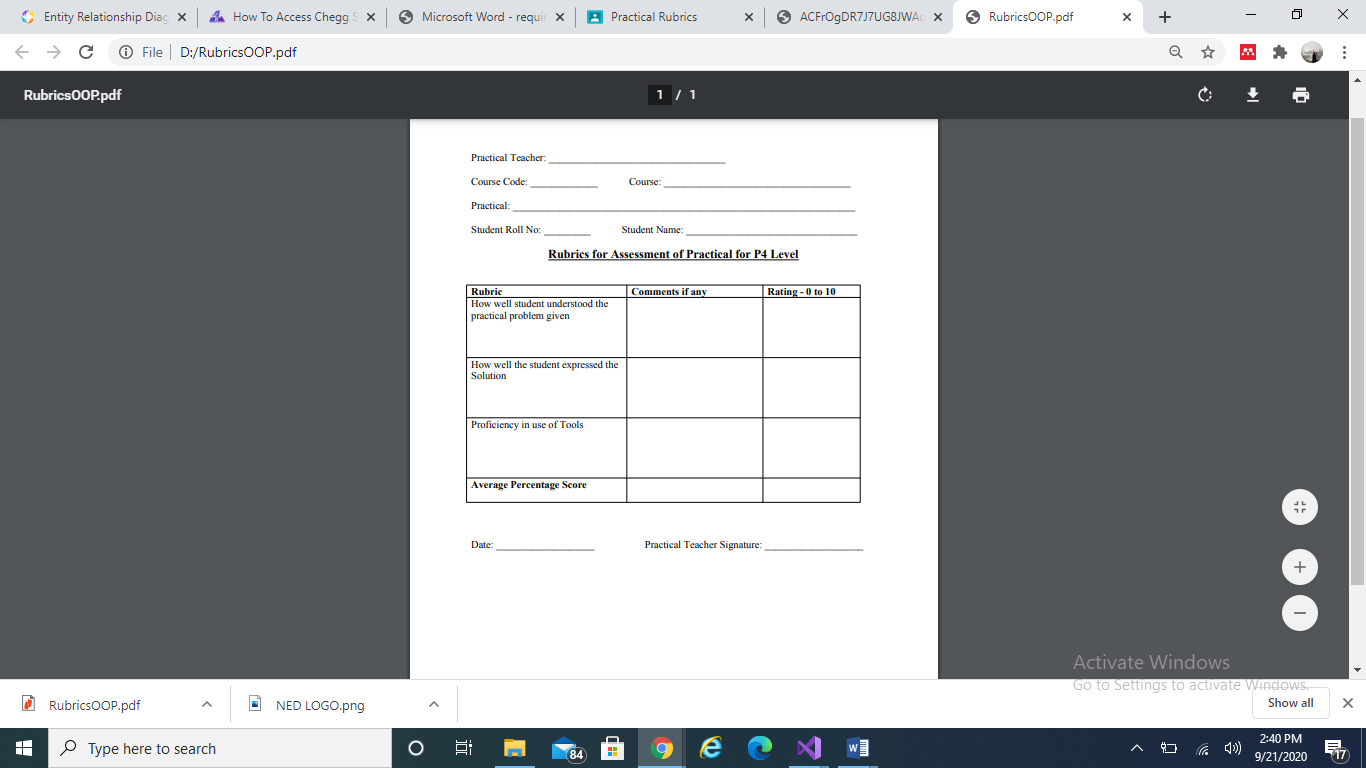
**NED University of Engineering & Technology,**

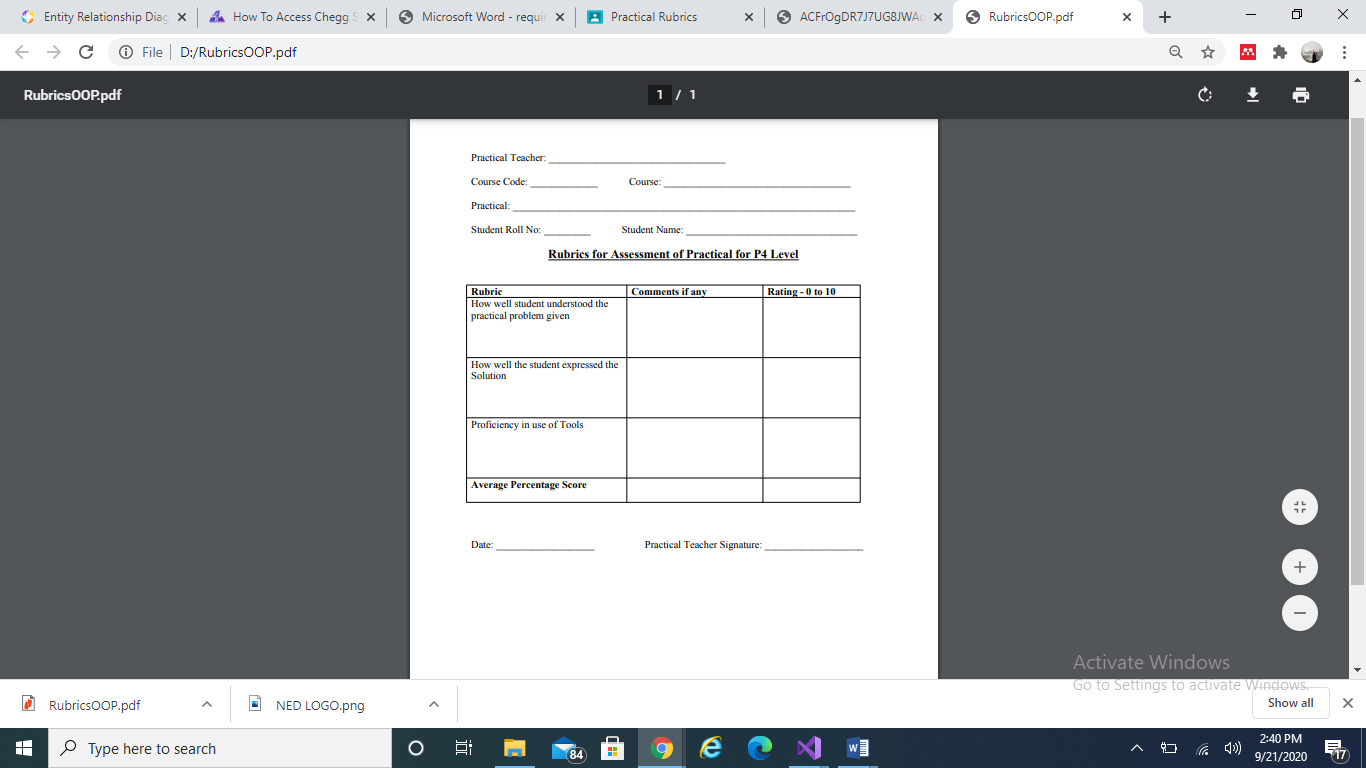
**Karachi – 75270, Pakistan**

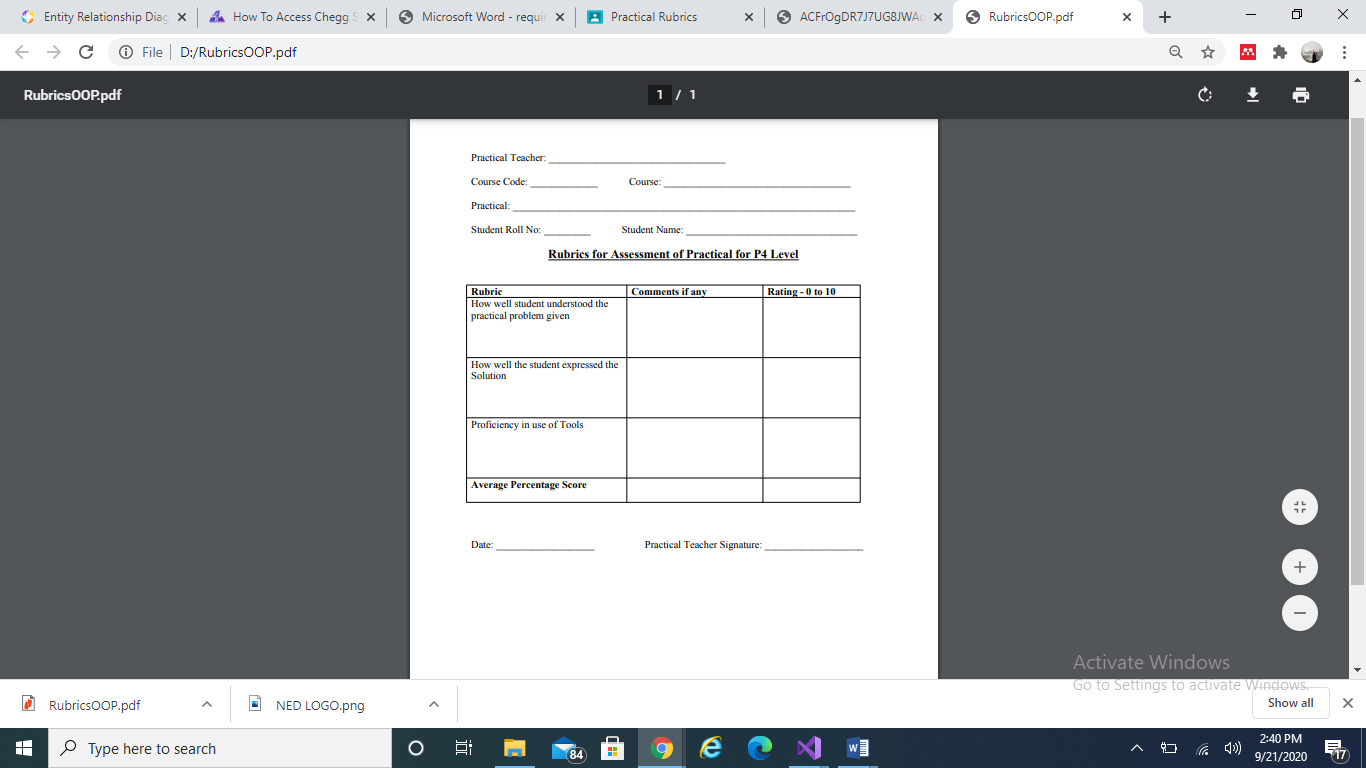
|  |  |  |  |
| --- | --- | --- | --- |
| **S.No.** | **Date** | **Topic** | **Signature/Remarks** |
| 1 | 23/06/2020 | Using basic C++ types and user-defined objects, C++ Input and Output and Expressions |  |
| 2 | 30/06/2020 | Basic C++ types , arrays and multi-dimensional arrays and structures |  |
| 3 | 7/07/2020 | Concept of Pointer and use of Pointers.  Reference and Dereference operator |  |
| 4 | 14/07/2020 | Classes and objects  Constructors and Destructors  . (No argument and with Argument Constructors) |  |
| 5 | 21/07/2020 | Problem solving with classes(Class description, class descriptions into class declarations,working with objects)  Inheritance |  |
| 6 | 28/07/2020 | Inheritance and Method Overiding |  |
| 7 | 18/08/2020 | Implementation of oop in C# |  |
| 8 | 25/08/2020 | Controls ,Graphics and,Mouse Events in C# |  |
| 9 | 1/09/2020 | Database Application using C# |  |
| 10 | 5/09/2020 | Database Binding using C# |  |











**LABTASK 1**

Q1) Write a program that read two integers from the keyboard and print their sum and average.

PROGRAM:

#include<iostream>

#include<conio.h>

#include<string>

using namespace std;

int main()

{

int num1, num2, sum;

double avg;

cout << "Enter the first number : ";

cin >> num1;

cout << "Enter the second number : ";

cin >> num2;

sum = num1 + num2;

avg = sum / 2;

cout << "Sum is " << sum << endl;

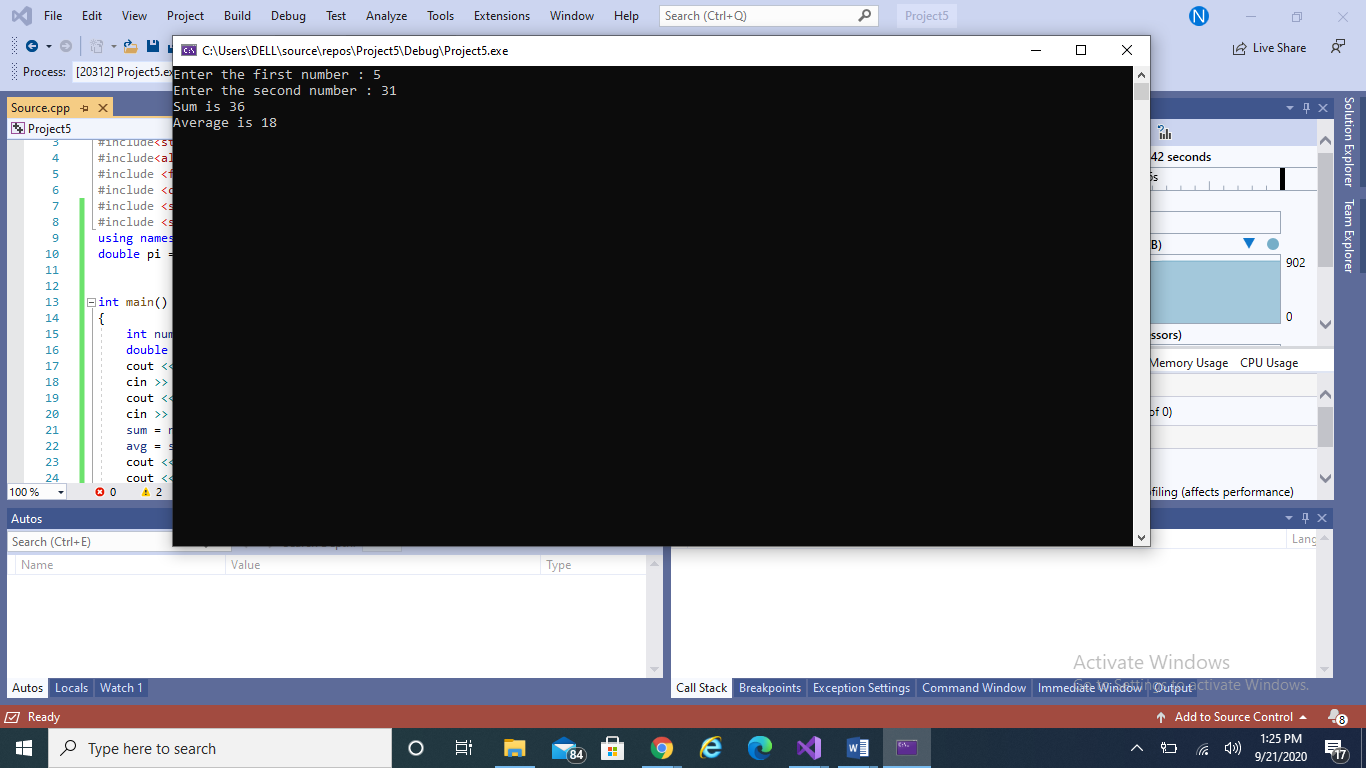
cout << "Average is " << avg<< endl;

\_getch();

return(0);

}

OUTPUT:



Q2) Write a program that prompts for time in seconds and output that time in hours, minutes, and seconds. Here student will learn the usage of divide and modulus arithmetic operators in integers.

int main()

{

int time, hour, min, sec, remaining\_time;

cout << "Enter the time in seconds: ";

cin >> time;

hour = time / 3600;

cout << "\n-------------------\nHours in time is: \n-------------------\n" << hour << endl;

remaining\_time = time - (hour \* 3600);

min = remaining\_time / 60;

cout << "\n-------------------\nMinutes in time is:\n-------------------\n " << min << endl;

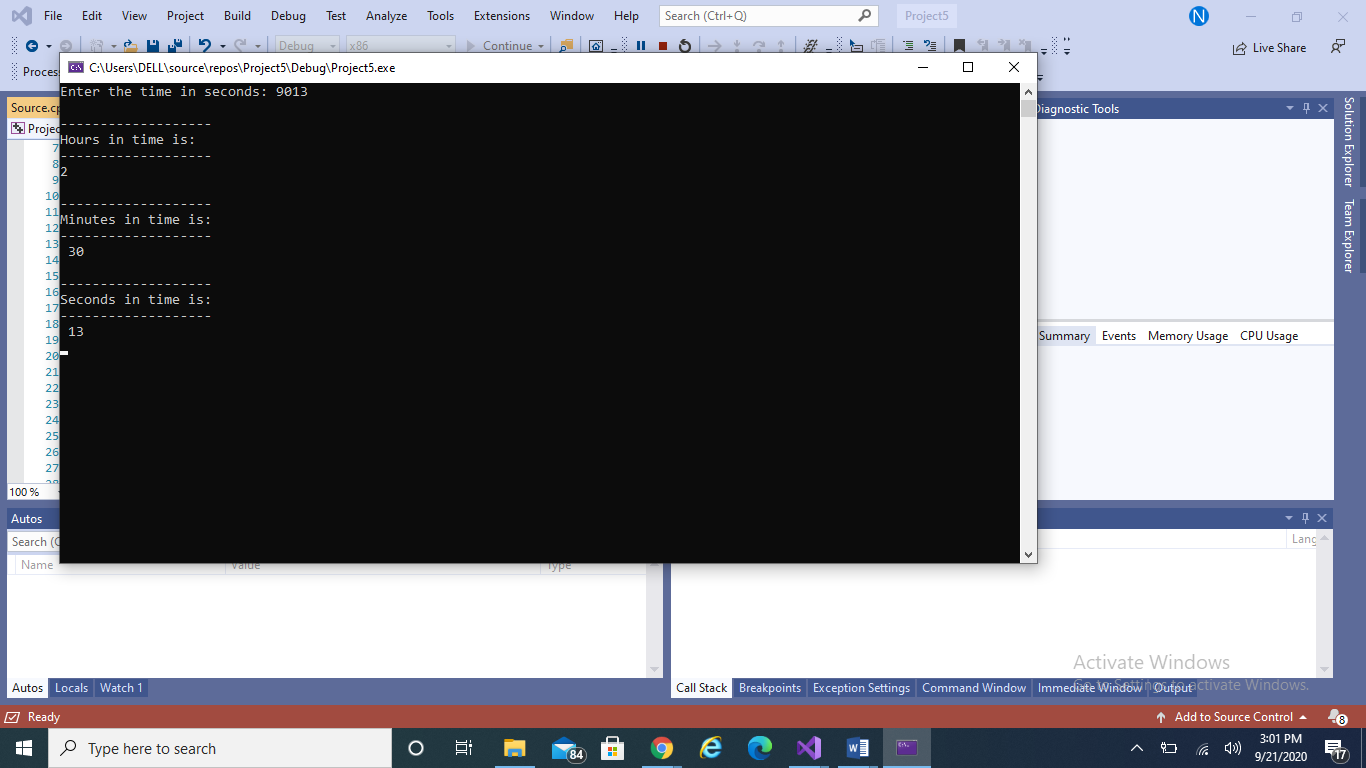
sec = remaining\_time - (min \* 60);

cout << "\n-------------------\nSeconds in time is:\n-------------------\n " << sec << endl;

\_getch();

return(0);

)



Q3) Write a program that prompts for a person’s height in inches. Convert this height measurement into feet by using the conversion factor of foot2Inch= 12 inch. Now, the value obtained can easily, be translated into feet and inches which are then output by the program.

PROGRAM:

int main()

{

float h\_inches, h\_feets;

cout << "Enter the height in inches : ";

cin >> h\_inches;

h\_feets = h\_inches / 12;

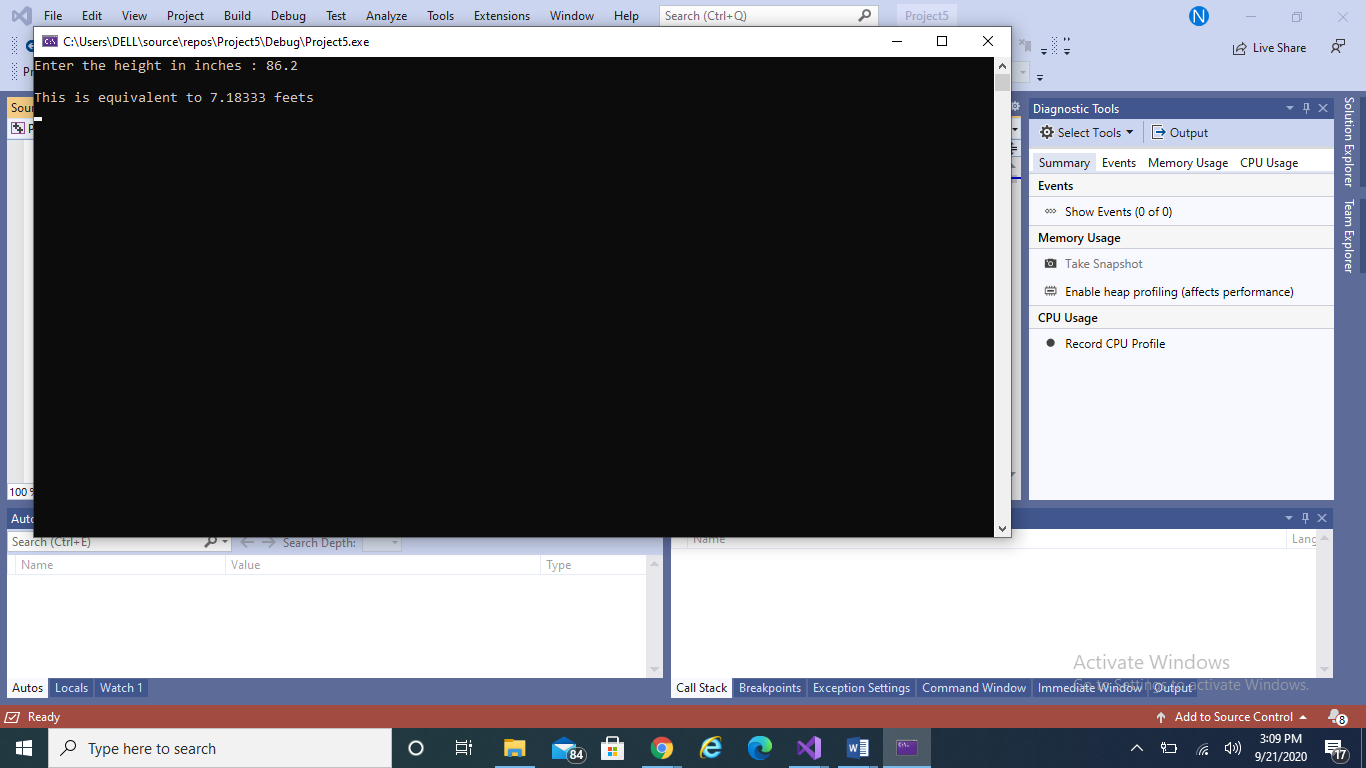
cout << endl << "This is equivalent to " << h\_feets << " feets" << endl;

\_getch();

return(0);

}

OUTPUT:



Q4) Write a program that prompts for amount in rupees and show how many 1000’s,

500’s, 100’s, 50’s, 10’s, 5’s, 2’s and 1’s in it.

PROGRAM:

int main()

{

int amount, thousand, fivehund, hund, fiftys, tens, fives, twos, ones, remaining\_am;

cout << "Enter amount in rupees : ";

cin >> amount;

thousand = amount / 1000;

cout << "\n------------------ - \n1000's in the given amount is\n-------------------\n : " << thousand << endl;

remaining\_am = amount - (thousand \* 1000);

fivehund = remaining\_am / 500;

cout << "\n------------------ - \n500's in the given amount is :\n-------------------\n " << fivehund << endl;

remaining\_am = remaining\_am - (fivehund \* 500);

cout << remaining\_am;

hund = remaining\_am / 100;

cout << "\n------------------ - \n100's in the given amount is :\n-------------------\n " << hund << endl;

remaining\_am = remaining\_am - (hund \* 100);

fiftys = remaining\_am / 50;

cout << "\n------------------ - \n50's in the given amount is :\n-------------------\n " << fiftys << endl;

remaining\_am = remaining\_am - (fiftys \* 50);

tens = remaining\_am / 10;

cout << "\n-------------------\n10's in the given amount is :\n-------------------\n " << tens << endl;

remaining\_am = remaining\_am - (tens \* 10);

fives = remaining\_am / 5;

cout << "\n-------------------\n5's in the given amount is :\n-------------------\n " << fives << endl;

remaining\_am = remaining\_am - (fives \* 5);

twos = remaining\_am / 2;

cout << "\n-------------------\n2's in the given amount is :\n-------------------\n " << twos << endl;

remaining\_am = remaining\_am - (twos \* 2);

ones = remaining\_am / 1;

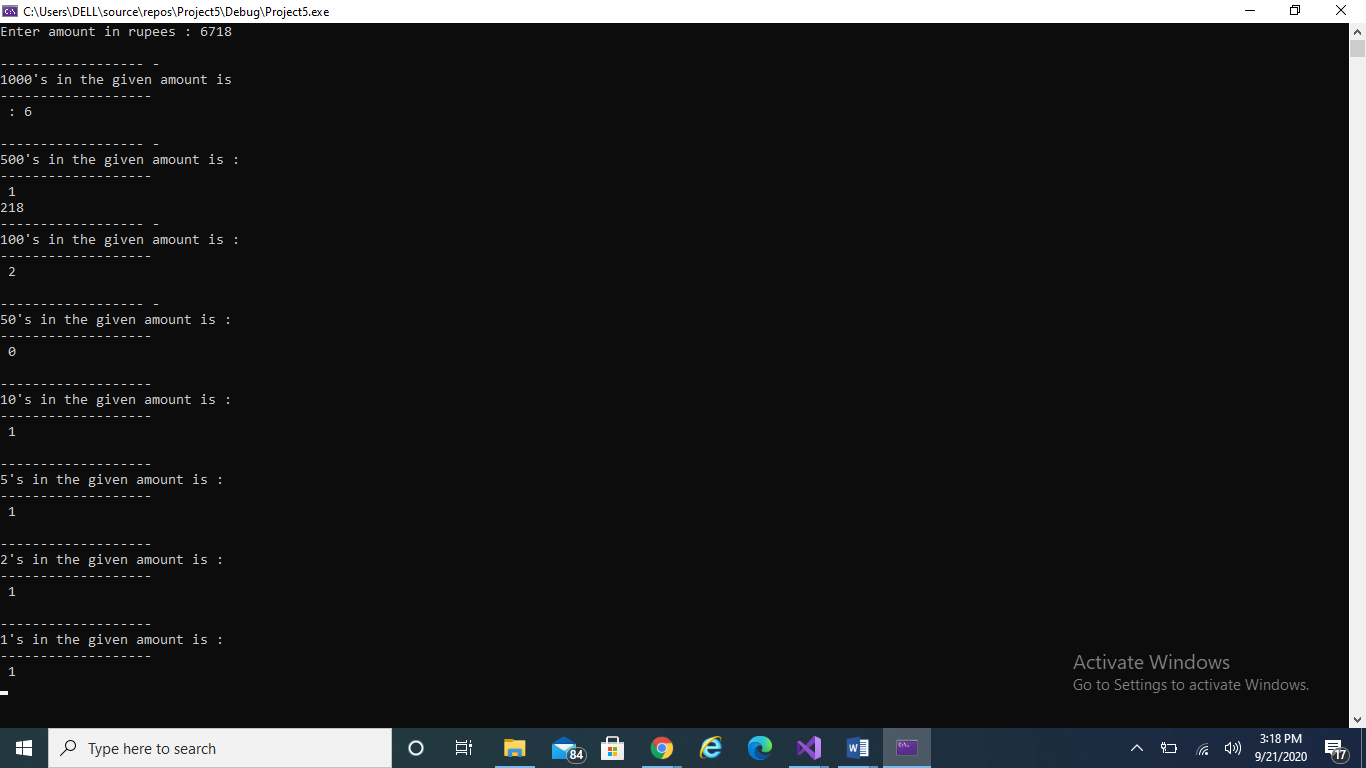
cout << "\n-------------------\n1's in the given amount is :\n-------------------\n " << ones << endl;

\_getch();

return(0);

}

OUTPUT:



**Q4)** Write a program that calculates the temperature in Fahrenheit. For that it prompts for temperature in Celsius degrees. Formula to calculate Fahrenheit temperature is Fahrenheit=Celsius (9/5+32). Once if the task done do the vice versa i.e. Celsius=5/9(Fahrenheit -32)

PROGRAM:

int main()

{

double temp\_cel, temp\_far;

cout << "Enter the Celsius temperature: ";

cin >> temp\_cel;

temp\_far = (temp\_cel \* 9 / 5) + 32;

cout << "Farenheit Temperature: " << temp\_far << endl;

temp\_cel = 0;

temp\_far = 0;

cout << "Enter the Farenheit temperature: " ;

cin >> temp\_far;

temp\_cel = (temp\_far - 32) \* 5/9;

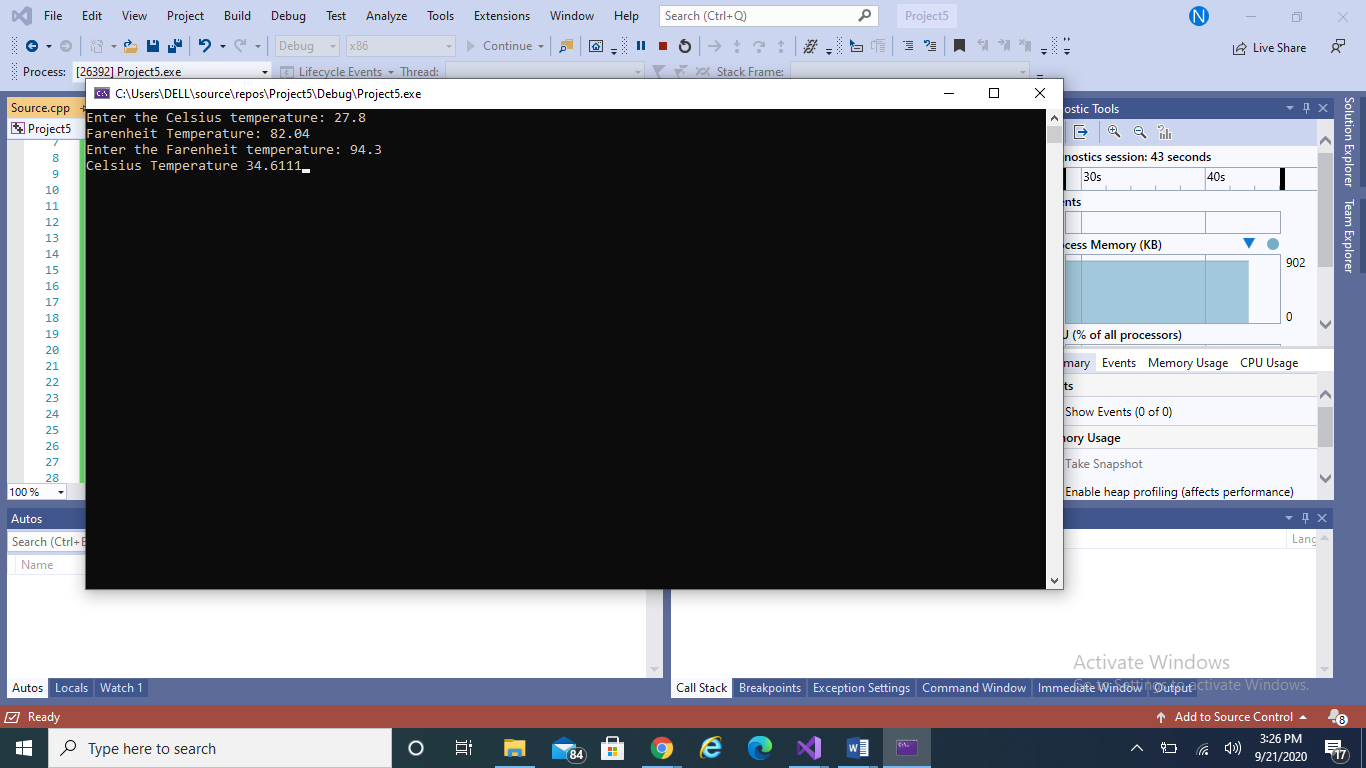
cout << "Celsius Temperature " << temp\_cel;

\_getch();

return(0);

}

OUTPUT:



**Q5)** Write a program that inputs a two digit integer value, and output its reverse order.

PROGRAM:

int main()

{

int num, first\_digit, second\_digit;

cout << "Enter a 2 digit integer value: ";

cin >> num;

first\_digit = num % 10;

second\_digit = num / 10;

cout << "Reverse of this value is: " << first\_digit;

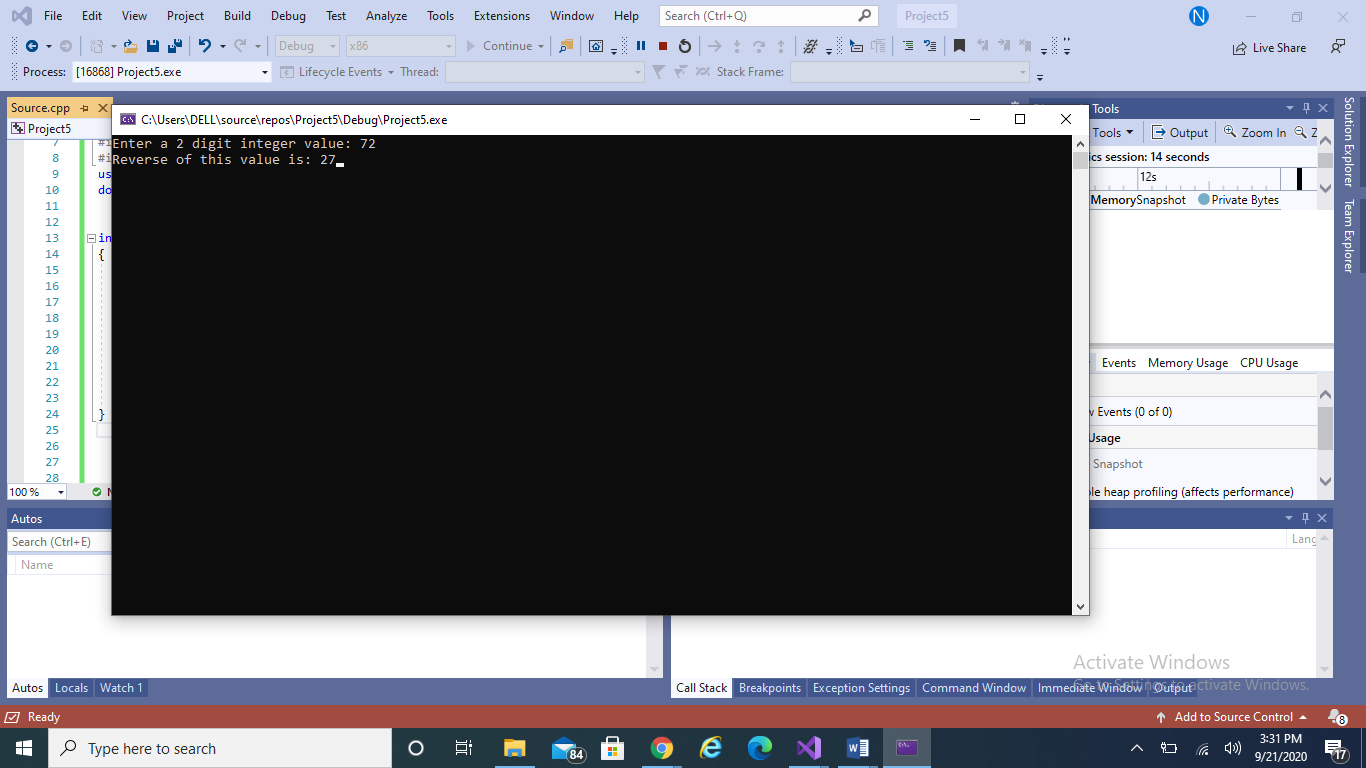
cout << second\_digit;

\_getch();

return(0);

}

OUTPUT:



**Q6)** Write a program that reads the two digit number as two characters chTen and chUnit and convert that two digit number into an integer value. In order to compute the corresponding integer value, each character must be converted to the digit in the range 0 to 9. this is done by subtracting 48(‘0’) from the ASCII value of the character. ValueTen=chTen-‘0’; // ’8’-‘0’ is 8

ValueUnit=chUnit-‘0’; // ’2’-‘0’ is 2. To create integer value for M, the positional value of each digit must be used. In this case multiply ValueTen by 10. M=ValueTen\*10+ValueUnit;//m=8\*10+2=82

PROGRAM:

int main()

{

char chNum[2], chTen, chUnit;

int ValueTen, ValueUnit, M;

cout << "Enter a two digit number: ";

cin >> chNum;

chTen = chNum[0];

chUnit = chNum[1];

ValueTen = chTen - 48;

ValueUnit = chUnit - 48;

M = ValueTen \* 10 + ValueUnit;

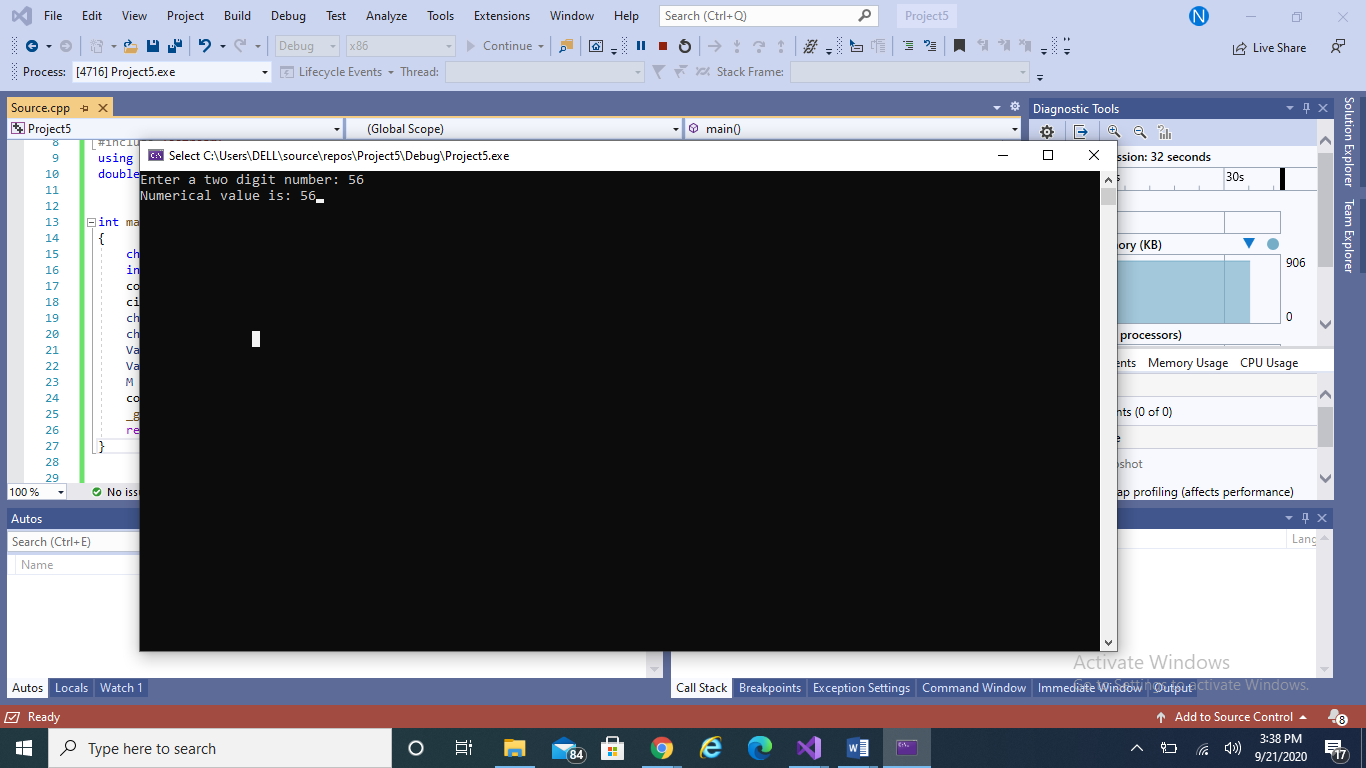
cout << "Numerical value is: " << M;

\_getch();

return(0);

}

OUTPUT:



**LABTASK 2**

**Q1)** Write a program to compute the length of the line segment connecting two points. For input the two numbers representing each point are entered in the form (x,y). The parentheses and comma are read as character data and then discarded.

PROGRAM:

# include<iostream>

# include<conio.h>

using namespace std;

int main()

{

char x[5];

char y[5];

double b1, b2, d, sqr;

cout << "ENTER FIRST CORDINATES IN FORM OF (X1,Y1)" << endl;

for (int i = 0; i < 5; i++)

{

cin >> x[i];

}

cout << "ENTER SECOND CORDINATES IN FORM OF (X2,Y2)" << endl;

for (int j = 0; j < 5; j++)

{

cin >> y[j];

}

b1 = (x[1] - y[1]) \* (x[1] - y[1]);

b2 = (x[3] - y[3]) \* (x[3] - y[3]);

d = b1 + b2;

sqr = sqrt(d);

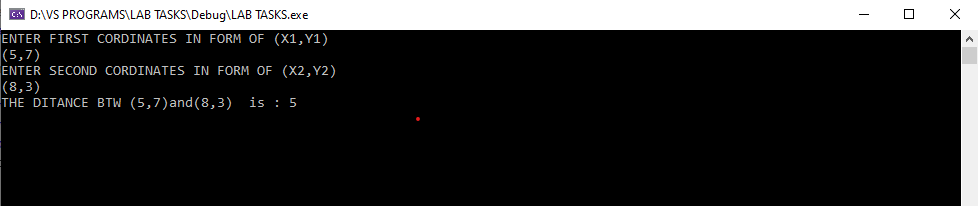
cout << "THE DITANCE BTW " << "(" << x[1] << "," << x[3] << ")" << "and" << "(" << y[1] << "," << y[3] << ") is : " << sqr << endl;

\_getch;

return 0;

}

OUTPUT:

****

**Q2)** Take two matrix M1 and M2 as an input and perform the addition and

multiplication of these two matrices.

**A#** PROGRAM:

int main()

{

int r, c, a[100][100], b[100][100], sum[100][100], i, j;

cout << "Enter number of rows (between 1 and 100): ";

cin >> r;

cout << "Enter number of columns (between 1 and 100): ";

cin >> c;

cout << endl << "Enter elements of 1st matrix: " << endl;

// Storing elements of first matrix entered by user.

for (i = 0; i < r; ++i)

for (j = 0; j < c; ++j)

{

cout << "Enter element a" << i + 1 << j + 1 << " : ";

cin >> a[i][j];

}

// Storing elements of second matrix entered by user.

cout << endl << "Enter elements of 2nd matrix: " << endl;

for (i = 0; i < r; ++i)

for (j = 0; j < c; ++j)

{

cout << "Enter element b" << i + 1 << j + 1 << " : ";

cin >> b[i][j];

}

// Adding Two matrices

for (i = 0; i < r; ++i)

for (j = 0; j < c; ++j)

sum[i][j] = a[i][j] + b[i][j];

// Displaying the resultant sum matrix.

cout << endl << "\n-----------------\nSum of two matrix is: \n-----------------\n " << endl;

for (i = 0; i < r; ++i)

for (j = 0; j < c; ++j)

{

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

if (j == c - 1)

cout << endl;

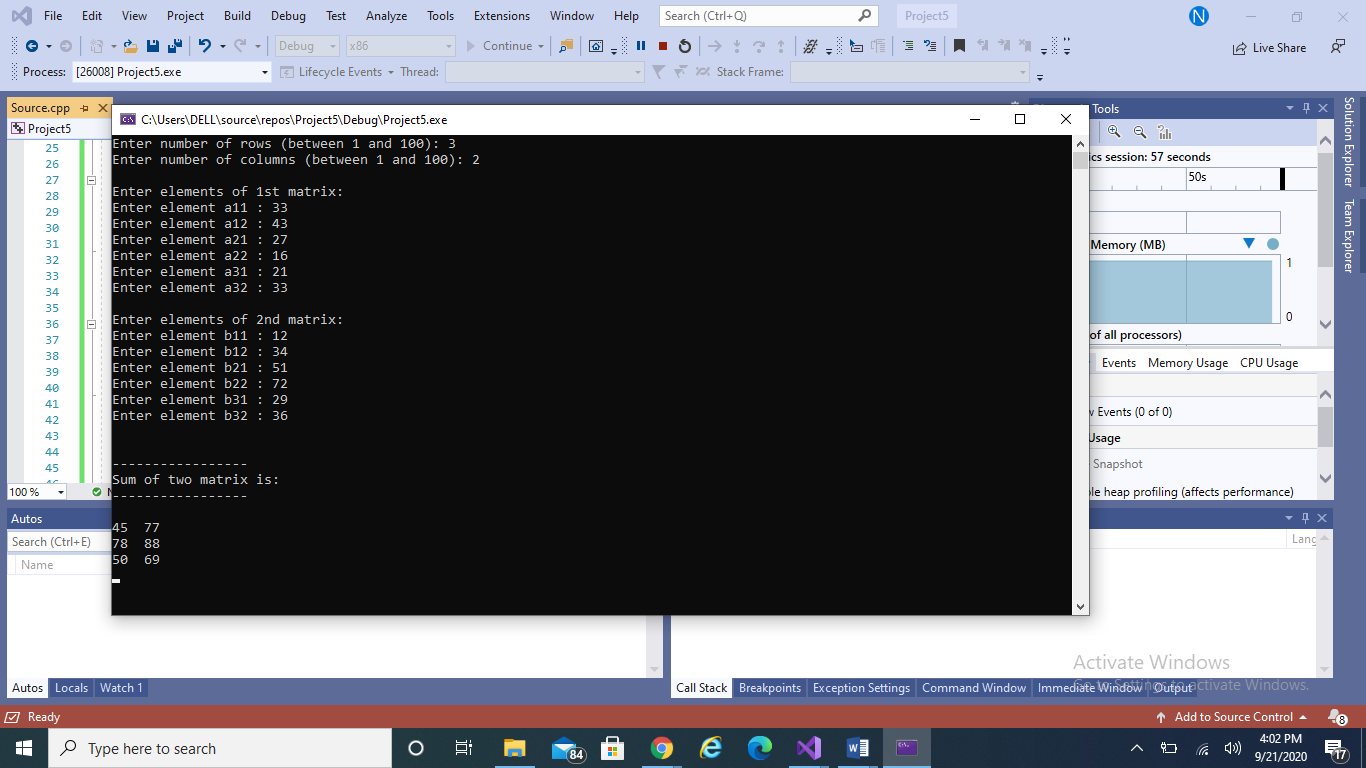
}

\_getch();

return(0);

}

OUTPUT:



**B#** PROGRAM:

int main()

{

int a[10][10], b[10][10], mult[10][10], r1, c1, r2, c2, i, j, k;

cout << "Enter rows and columns for first matrix: ";

cin >> r1 >> c1;

cout << "Enter rows and columns for second matrix: ";

cin >> r2 >> c2;

// If column of first matrix in not equal to row of second matrix,

// ask the user to enter the size of matrix again.

while (c1 != r2)

{

cout << "Error! column of first matrix not equal to row of second.";

cout << "Enter rows and columns for first matrix: ";

cin >> r1 >> c1;

cout << "Enter rows and columns for second matrix: ";

cin >> r2 >> c2;

}

// Storing elements of first matrix.

cout << endl << "Enter elements of matrix 1:" << endl;

for (i = 0; i < r1; ++i)

for (j = 0; j < c1; ++j)

{

cout << "Enter element a" << i + 1 << j + 1 << " : ";

cin >> a[i][j];

}

// Storing elements of second matrix.

cout << endl << "Enter elements of matrix 2:" << endl;

for (i = 0; i < r2; ++i)

for (j = 0; j < c2; ++j)

{

cout << "Enter element b" << i + 1 << j + 1 << " : ";

cin >> b[i][j];

}

// Initializing elements of matrix mult to 0.

for (i = 0; i < r1; ++i)

for (j = 0; j < c2; ++j)

{

mult[i][j] = 0;

}

// Multiplying matrix a and b and storing in array mult.

for (i = 0; i < r1; ++i)

for (j = 0; j < c2; ++j)

for (k = 0; k < c1; ++k)

{

mult[i][j] += a[i][k] \* b[k][j];

}

// Displaying the multiplication of two matrix.

cout << endl << " \n----------------\nOutput Matrix:\n----------------\n " << endl;

for (i = 0; i < r1; ++i)

for (j = 0; j < c2; ++j)

{

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

if (j == c2 - 1)

cout << endl;

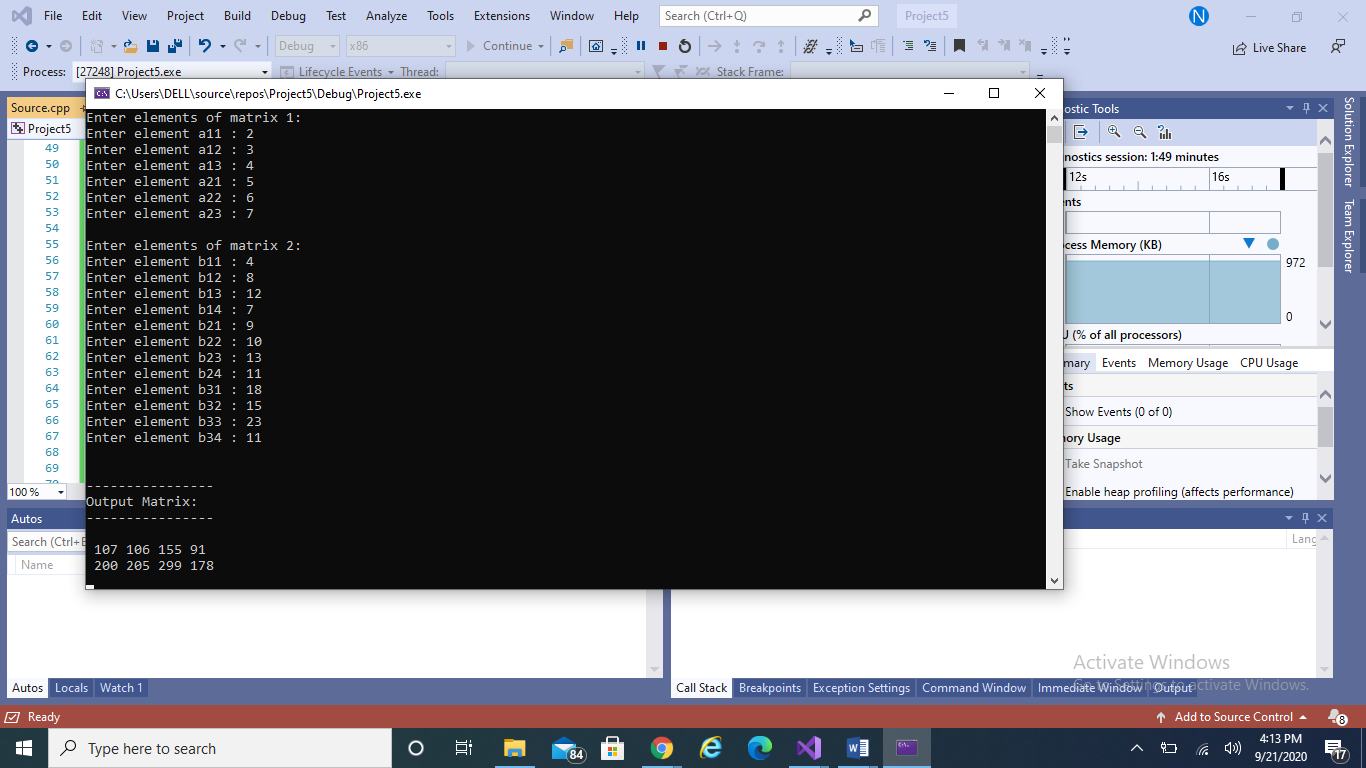
}

\_getch();

return(0);

}

OUTPUT:



**Q3) & Q4)** Write a program that define structure to maintain student records, structure student

should be consisting of the following attributes.1. Student first name (max 20 characters)

2. Student last name (max 20 characters) 3. Student scores (float/double) e.g 85.4. Pass the structure define in Q.3 to some function to move to display.

PROGRAM:

struct student

{

string fname;

string lname;

double score;

};

void showdata(student s1)

{

cout << " THE STUDENT RECORD IS " << endl << endl;

cout << "FIRST NAME : " << s1.fname << endl;

cout << "LAST NAME  : " << s1.lname << endl;

cout << "SCORE OUT OF 50 : " << s1.score << endl;

};

int main()

{

student s1;

cout << "NAME (max. 20 characters)    :  " << endl;

cin >> s1.fname;

cout << "LAST NAME(max. 20 characters):  " << endl;

cin >> s1.lname;

cout << "SCORE    :  " << endl;

cin >> s1.score;

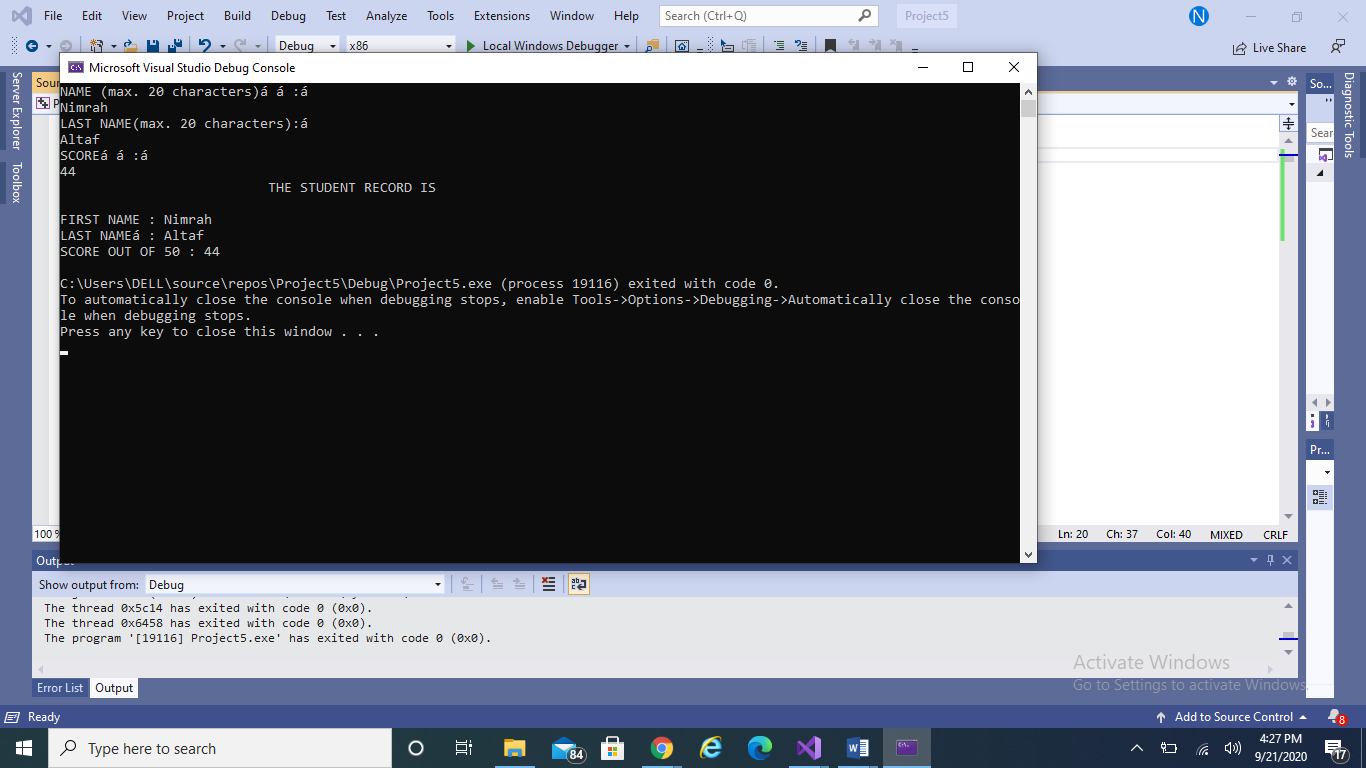
showdata(s1);

\_getch;

return 0;

}

OUTPUT:



**Q5)** Create nested structure . Firstly define Address structure and then call address Structure in Employee Structure and program will give some raise in salary ,if it is less than 50000 Address (house no, city, pin code) Employee (empid,name,salary,address).

PROGRAM:

struct Address

{

string HouseNo;

string City;

int PinCode;

};

struct Employee

{

string EmpID;

string name;

int salary;

Address add;

};

int main()

{

Employee e1;

e1.EmpID = "IT\_DEP-0023";

e1.name = "Rubab Amir";

e1.salary = 45000;

e1.add.City = "Karachi";

e1.add.HouseNo = "34-A Block 2, Gulshan";

e1.add.PinCode = 7500;

if (e1.salary < 50000)

{

cout << "As " << e1.name << " of employee id " << e1.EmpID << " salary is less than 50 000 hence " << e1.name << " will receive a pay raise.";

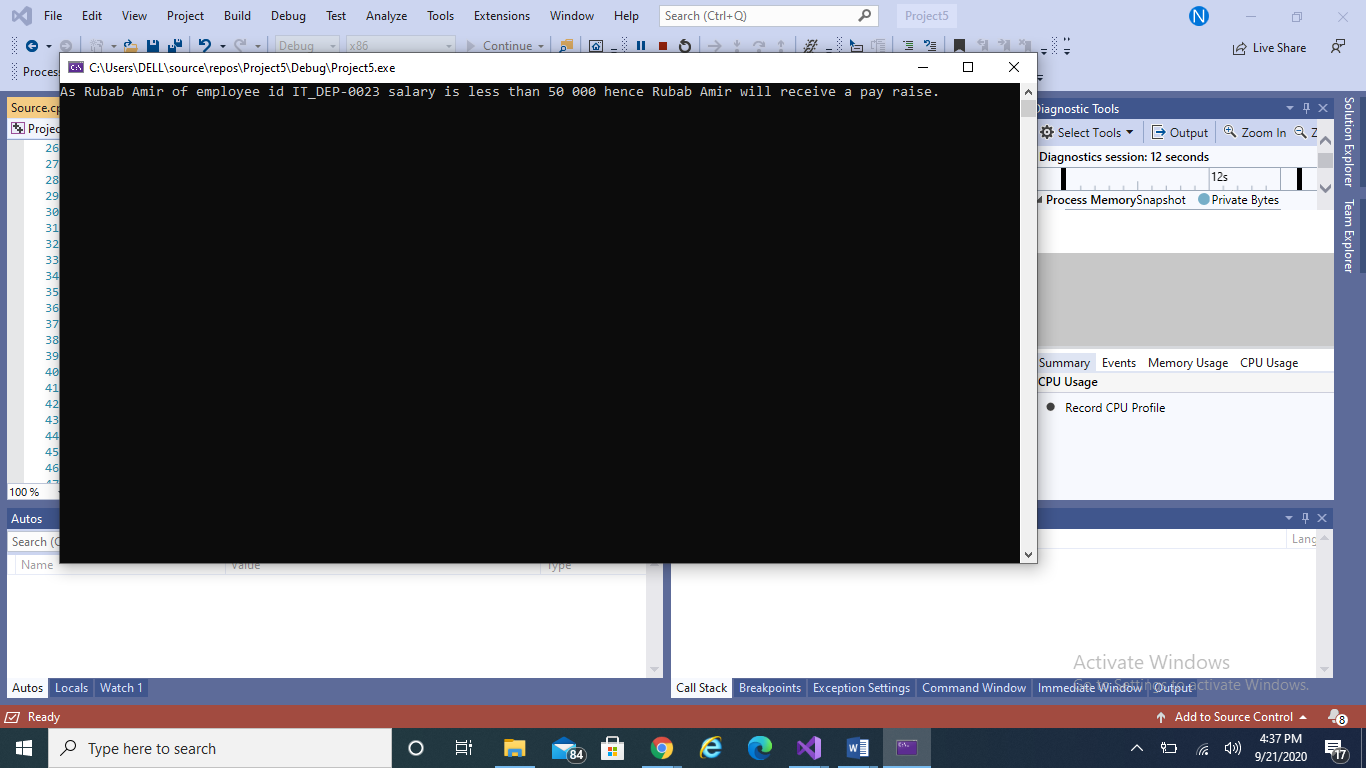
}

\_getch();

return(0);

}

OUTPUT:



**Q6)** Write a C++ Program to create a small Calculator by using Pointers.

PROGRAM:

int sum(int\* p1, int\* p2) // we use pointers to define the parameter of a function if we want to pass the address of variables to it.

{

int result;

result = \*p1 + \*p2;

return result;

}

int diff(int\* p1, int\* p2)

{

int result;

result = \*p1 - \*p2;

return result;

}

int multi(int\* p1, int\* p2)

{

int result;

result = \*p1 \* \*p2;

return result;

}

int div(int\* p1, int\* p2)

{

int result;

result = \*p1 / \*p2;

return result;

}

int main()

{

int num1, num2;

int\* pt1, \* pt2;

char op;

cout << "Enter value for num1" << endl;

cin >> num1;

cout << "Enter value for num2" << endl;

cin >> num2;

pt1 = &num1;

pt2 = &num2;

cout << "Enter an operator";

cin >> op;

switch (op)

{

case '+':

{

cout << num1 << " + " << num2 << " = "<< sum(&num1, &num2) << endl;

}

break;

case '-':

{

cout << num1 << " - " << num2 << " = " << diff(&num1, &num2) << endl;

}

break;

case '\*':

{

cout << num1 << " \* " << num2 << " = " << multi(&num1, &num2) << endl;

}

break;

case '/':

{

cout << num1 << " / " << num2 << " = " << div(&num1, &num2) << endl;

}

break;

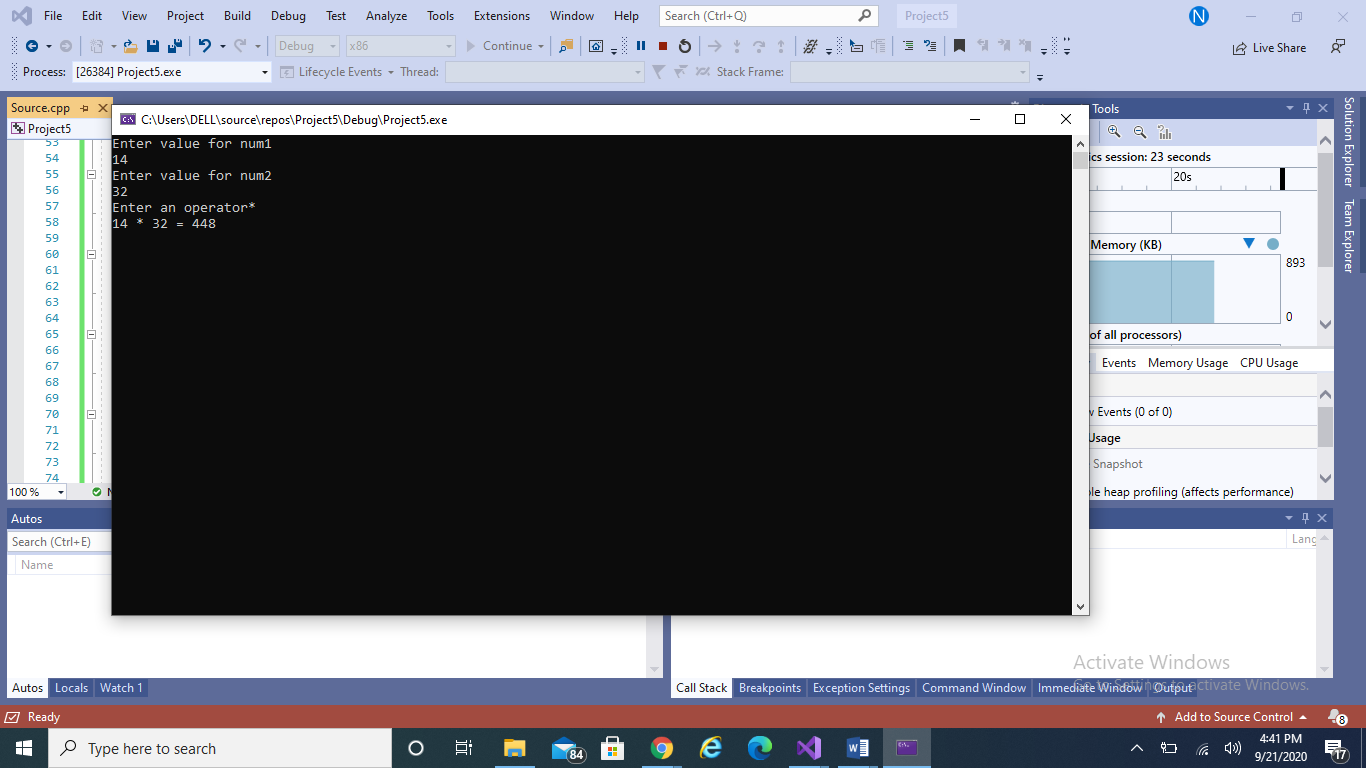
}

\_getch();

return 0;

}

OUTPUT:



**LABTASK 3**

**Q1)** Write a program in C++ that can calculate the factorial of a number by passing the address of that number to a function using pointers.

PROGRAM:

int fact(int\* n)

{

int fact = 1;

if (\*n <= 1)

{

return 1;

}

else

{

for (int count = 1; count < \*n; count++)

{

fact += fact \* count;

}

return fact;

}

//return \*n \* fact(\*n - 1);

}

int main()

{

int num;

int\* p1;

cout << "Please enter a number to calculate its factorial" << endl;

cin >> num;

p1 = &num;

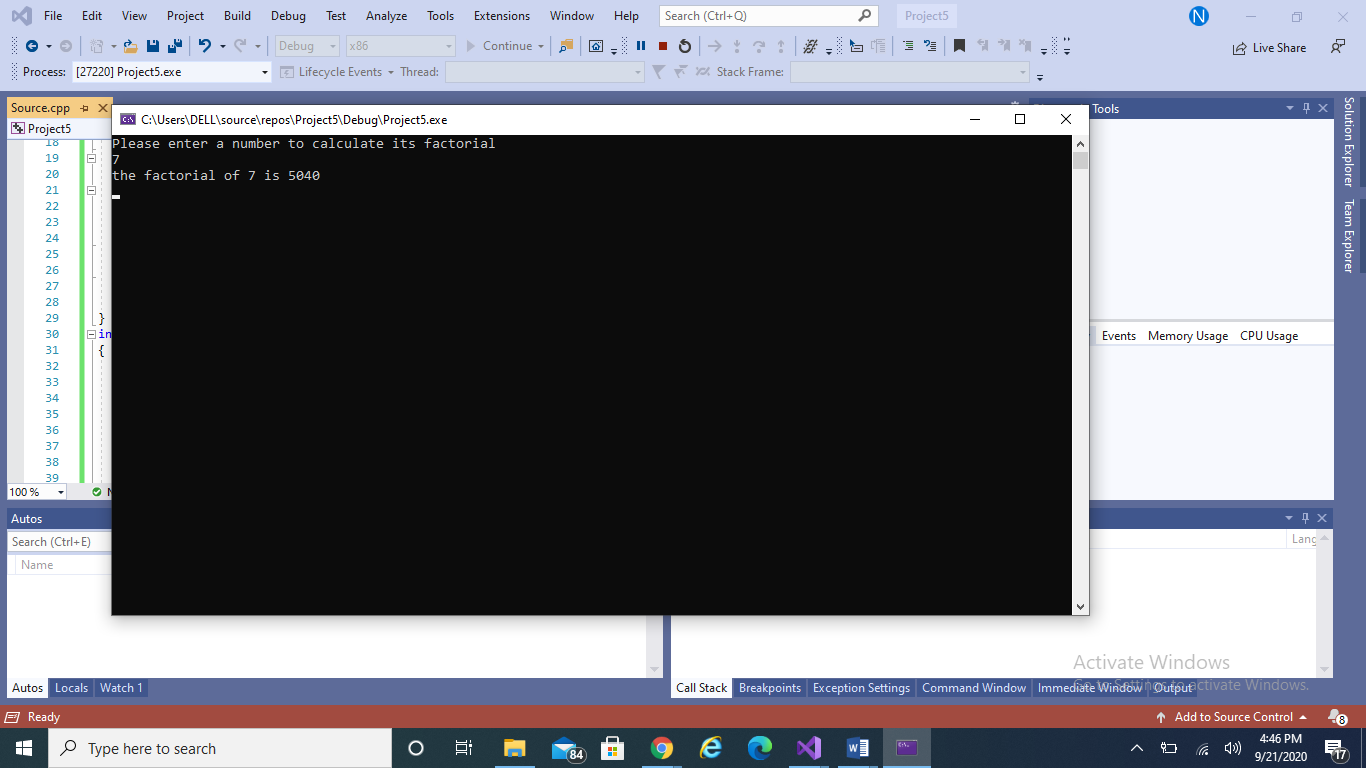
cout << "the factorial of " << num << " is " << fact(p1) << endl;

\_getch();

return 0;

}

OUTPUT:



**Q2)** Write down a C++ program that will declare and initialize two arrays and would generate the sum of these two arrays by using pointers.

PROGRAM:

void sum1(int\* p1, int\* p2)

{

int c[5]; // initialize a new array to get the sum of two arrays

int\* sum; // initialize a pointer to store the add of new array

sum = c; // address stored in the pointer

for (int i = 0; i < 5; i++)

{

\*sum = \*p1 + \*p2; // the content of the add stored at p1 is added to the content in the add stored at p2

// the sum is then saved at the add stored in sum.

cout << "a[" << i << "]= " << \*p1 << " b[" << i << "]= " << \*p2 << " and their sum is " << \*sum << endl;

p1++;

p2++;

}

}

int main()

{

int num1[5] = { 3,6,9,12,15 };

int num2[5] = { 6,12,18,24,30 };

int\* pt1, \* pt2;

pt1 = num1;

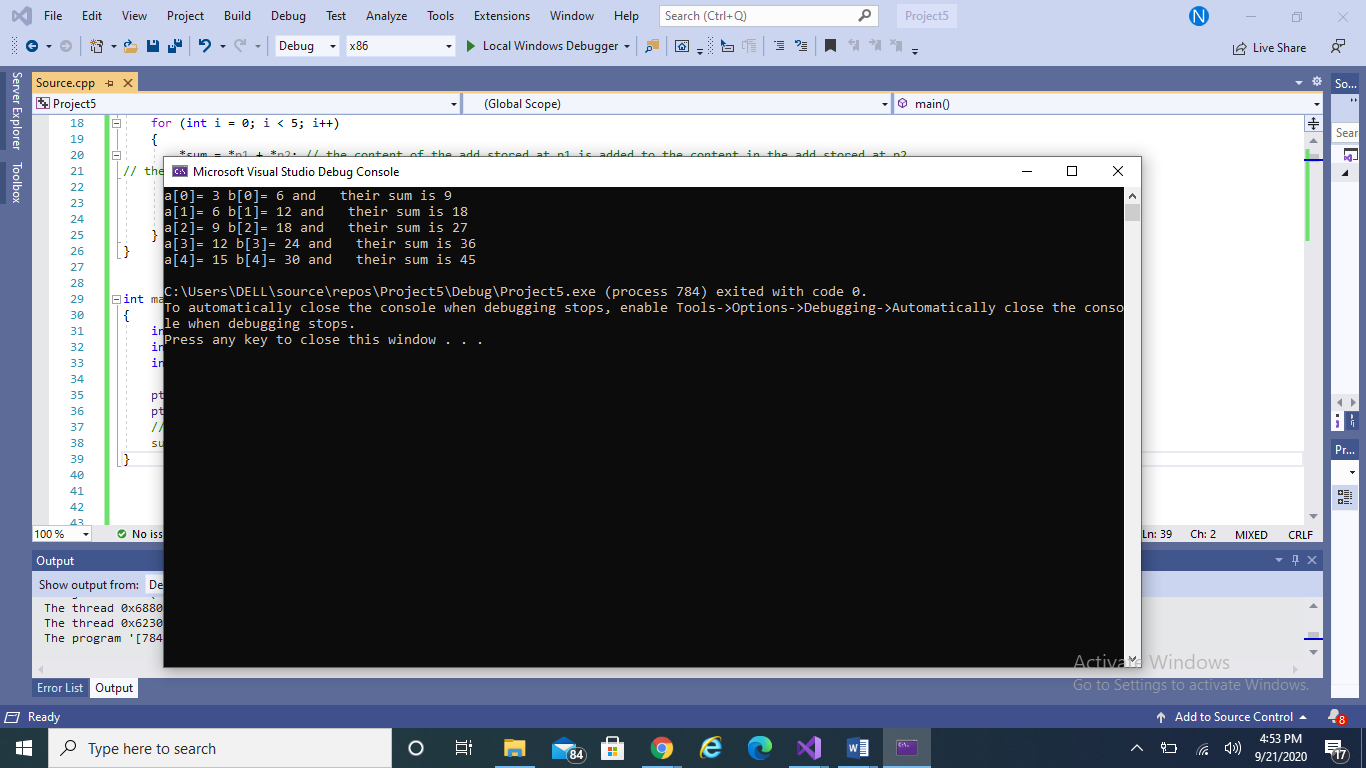
pt2 = num2;

// sum of 2 arrays

sum1(num1, num2);

}

OUTPUT:



**Q3)** Write down a C++ program , that will Calculate the area of a Circle by using Constant Data member PI=3.14, and by using Constant Pointer.

PROGRAM:

int main()

{

double rad, area;

const double pi = 3.14;

const double\* ptr = &pi;

cout << "Enter the radius to calculate the area of a circle";

cin >> rad;

area = \*ptr \* rad \* rad;

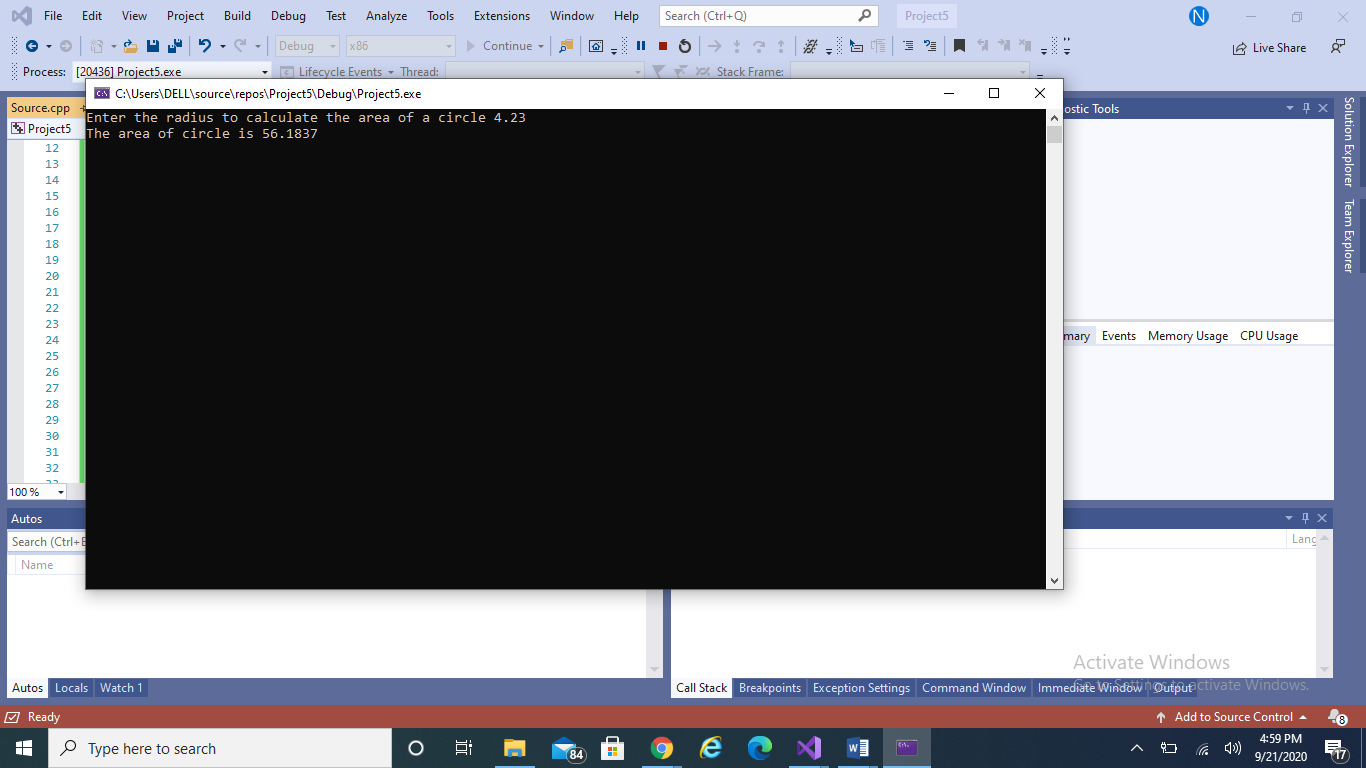
cout << "The area of circle is " << area;

\_getch();

return 0;

}

OUTPUT:



**Q4)** Write down a C++ program that would generate some table 2\*1=2,2\*10=20 by using pointers.

PROGRAM:

int main()

{

int num, len;

int\* ptr1,\*ptr2;

cout << "Enter a number to calculate its table";

cin >> num;

cout << "Enter the the length of the table";

cin >> len;

ptr1 = &num;

ptr2 = &len;

for (int i = 1; i <= \*ptr2; i++)

{

cout << \*ptr1 << " \* " << i << " = " << \*ptr1 \* i<< endl;

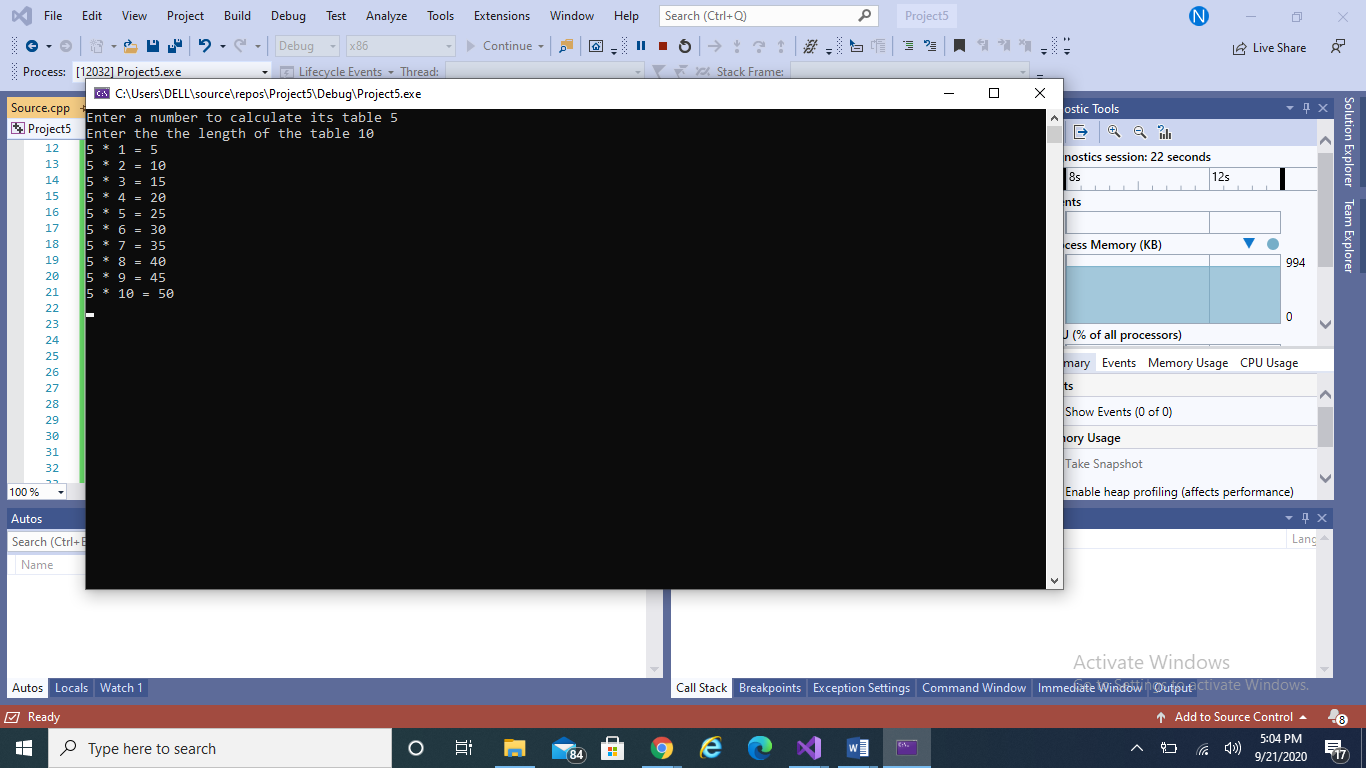
}

\_getch();

return 0;

}

OUTPUT:



**LABTASK 4**

**Q1)** Design and then implement to represent the class flight. A flight has a flight number, source, destination and a number of available seats.

PROGRAM:

class flight

{

int f\_num, avail\_seats;

string source, dest;

private:

public:

// PART (a)

flight(int fn, string s, string d, int ns) : f\_num(fn), source(s), dest(d), avail\_seats(ns)

{

shortAndCapital(s); // Calling the method of part (f)

shortAndCapital(d);

};

// PART (b)

flight(int fn, int ns) : f\_num(fn), avail\_seats(ns)

{

source = "";

dest = "";

};

// PART (c)

flight(int fn) : f\_num(fn)

{

source = "";

dest = "";

avail\_seats = 0;

};

// PART (d)

int getFlightNum()

{

return f\_num;

}

string getSource()

{

return source;

}

string getDestination()

{

return dest;

}

int getNumOfSeats()

{

return avail\_seats;

}

// PART (e)

void setSource(string s)

{

source = s;

}

void setDestination(string d)

{

dest = d;

}

void setNumOfSeats(int ns)

{

avail\_seats = ns;

}

// PART (f)

void reserve(int numOfSeats)

{

if (numOfSeats < avail\_seats)

{

avail\_seats = avail\_seats - numOfSeats;

cout << "Seats reserved!" << endl;

}

else

{

cout << "Seats not available for reservation" << endl;

}

}

// PART (g)

void cancel(int numOfSeats)

{

avail\_seats += numOfSeats;

cout << "The Reservation has been Cancelled!" << endl;

}

// PART (h)

void toString()

{

cout << "Flight No: " << f\_num << endl;

cout << "From: " << shortAndCapital(source) << endl;

cout << "To: " << shortAndCapital(dest) << endl;

}

// PART (i)

void equals(int fn1, int fn2)

{

if (fn1 == fn2)

cout << "The two flights have the same flight number "<< endl;

}

// PART (j)

private:

string toupper(string a)

{

for (int i = 0; i < a.length(); i++)

{

if (a[i] >= 97 && a[i] <= 122)

a[i] -= 32;

else

continue;

}

return a;

}

string shortAndCapital(string name)

{

if (name.length() <= 3)

{

name = toupper(name);

return name;

}

else

{

name = name.substr(0, 3);

name = toupper(name);

return name;

}

}

};

int main()

{

int f\_num, avail\_seats = 336, noOfSeats;

string source, dest;

cout << "Enter Data of Flight 1: " << endl;

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

cin >> f\_num;

cout << "Enter source: " << endl;

cin >> source;

cout << "Enter dest: " << endl;

cin >> dest;

flight f1(f\_num, source, dest, avail\_seats);

f1.toString();

cout << "Enter Number of seats to reserve: " << endl;

cin >> noOfSeats;

f1.reserve(noOfSeats);

cout << "The number of seats that are now available in flight " << f1.getFlightNum() << " are " << f1.getNumOfSeats() << endl;

cout << endl << endl;

cout << "Enter Data of Flight 2: " << endl;

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

cin >> f\_num;

cout << "Enter source: " << endl;

cin >> source;

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

cin >> dest;

flight f2(f\_num, source, dest, avail\_seats);

f2.toString();

cout << "Enter Number of seats to reserve: " << endl;

cin >> noOfSeats;

f2.reserve(noOfSeats);

cout << "Enter Number of seats to cancel: " << endl;

cin >> noOfSeats;

f2.cancel(noOfSeats);

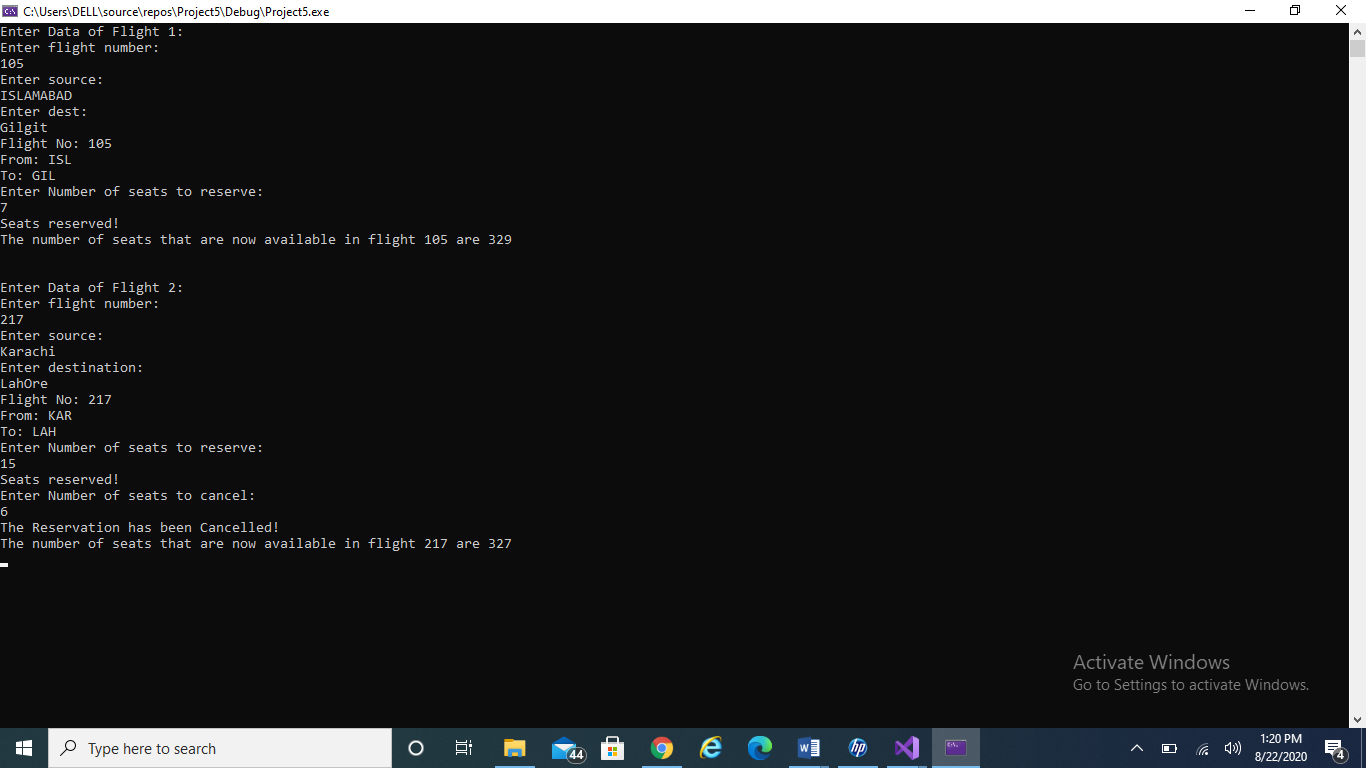
cout << "The number of seats that are now available in flight " << f2.getFlightNum() << " are " << f2.getNumOfSeats() << endl;

\_getch();

return 0;

}

OUTPUT:



**Q2)** Implement class car that has the following features brandNmae, priceNew, color, odometer.

PROGRAM:

class Car

{

private:

string brandName;

double priceNew;

string color;

double odometer;

public:

float getpriceNew()

{

return priceNew;

}

void setpriceNew(float pn)

{

priceNew = pn;

}

float getodometer()

{

return odometer;

}

void setodometer(float om)

{

odometer = om;

}

Car(string bn, string cl, double pn, double om) : brandName(bn), color(cl), priceNew(pn), odometer(om)

{};

// PART (A)

double usedprice;

double getPriceAfterUse(double pn, double om)

{

usedprice = pn \* (1 - (om / 600000));

return usedprice;

}

// PART (B)

void updateMileage(double traveledDistance)

{

odometer += traveledDistance;

}

// PART (C)

void outputDetails()

{

cout << "Brand Name: " << brandName << endl;

cout << "Color of the Car: " << color << endl;

cout << "New Price: " << priceNew << endl;

cout << "Odometer: " << odometer << endl;

cout << "Used Price: " << getPriceAfterUse(priceNew, odometer) << endl;

}

};

int main()

{

string br, cl;

double np, om;

cout << "Enter Brand Name: " << endl;

cin >> br;

cout << "Enter Color of the Car: " << endl;

cin >> cl;

cout << "Enter New Price: " << endl;

cin >> np;

cout << "Enter Odometer: " << endl;

cin >> om;

Car c1(br, cl, np, om);

c1.getPriceAfterUse(c1.getpriceNew(), c1.getodometer());

c1.outputDetails();

c1.updateMileage(10000.0);

c1.outputDetails();

c1.updateMileage(20000.0);

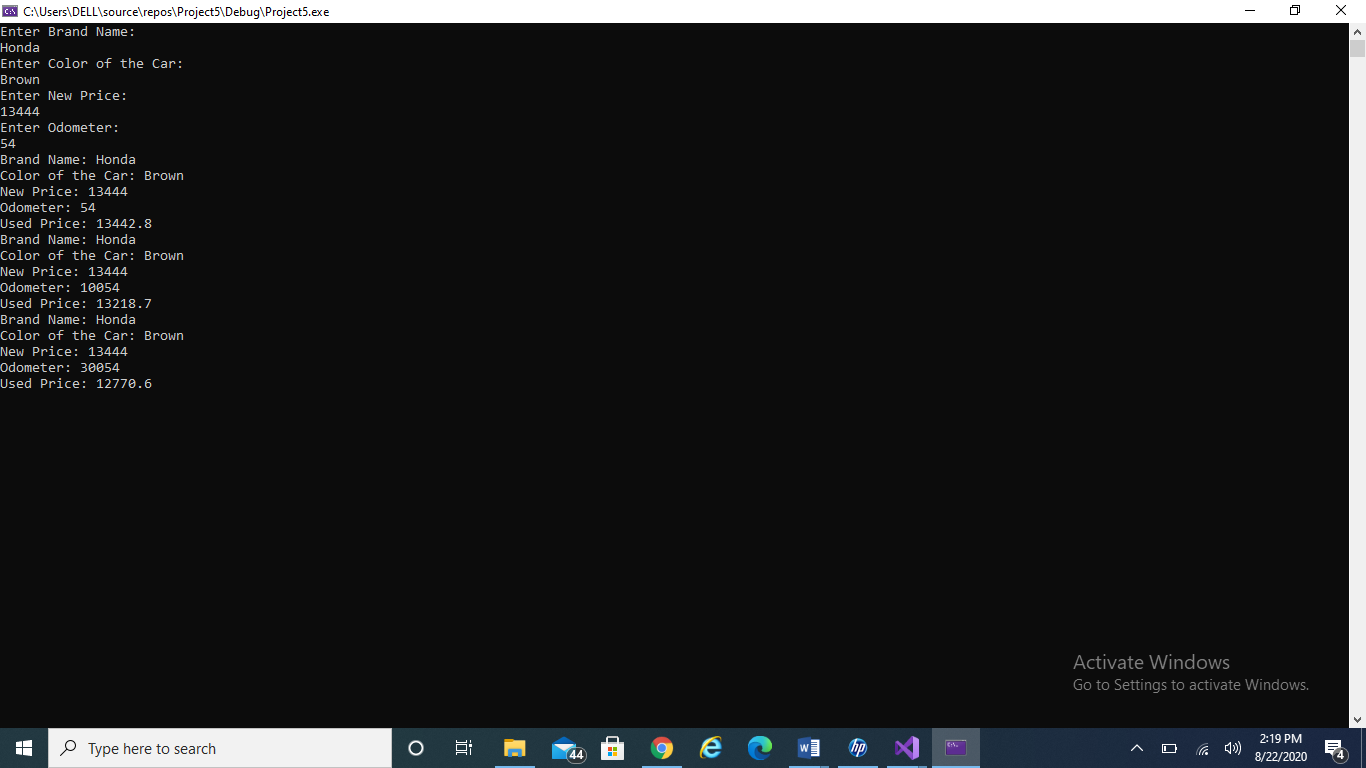
c1.outputDetails();

\_getch();

return(0);

}

OUTPUT:



**Q3)** Develop an application that computes the total cost of an order.

PROGRAM

class coffee

{

private:

int NumOfBags, count, remainingbags;

const double priceofbag = 5.50;

double cost;

public:

coffee(int Nb) : NumOfBags(Nb)

{};

double getCostOfOrder(int n)

{

return n \* priceofbag;

}

double largebox(int n)

{

return n \* 1.80;

}

double mediumbox(int n)

{

return n \* 1.00;

}

double smallbox(int n)

{

return n \* 0.60;

}

void totalcost(int NoOfBags)

{

count = NoOfBags / 20;

if (NoOfBags < 20)

{

count = 1;

cost = largebox(count);

cout << "\t" << count << " Large - $" << cost << endl;

}

else

{

cost = largebox(count);

cout << "\t" << count << " Large - $" << cost << endl;

}

remainingbags = NoOfBags - (count \* 20);

count = remainingbags / 10;

if (remainingbags < 10)

{

count = 1;

cost += mediumbox(count);

cout << "\t" << count << " Medium - $" << mediumbox(count) << endl;

}

else

{

cost += mediumbox(count);

cout << "\t" << count << " Medium - $" << mediumbox(count) << endl;

}

remainingbags = remainingbags - (count \* 10);

count = remainingbags / 5;

if (remainingbags < 5)

{

count = 1;

cost += smallbox(count);

cout << "\t" << count << " Small - $" << smallbox(count) << endl;

}

else

{

cost += smallbox(count);

cout << "\t" << count << " Small - $" << smallbox(count) << endl;

}

cout << "Yout total cost is: " << getCostOfOrder(NumOfBags) + cost << endl;

}

void displayinfo()

{

cout << "Number of Bags Ordered: " << NumOfBags << endl;

cout << "The Cost of Order: " << getCostOfOrder(NumOfBags) << endl;

cout << "Boxes Used:" << endl;

totalcost(NumOfBags);

}

};

int main()

{

int nb;

cout << "Enter number of Bags" << endl;

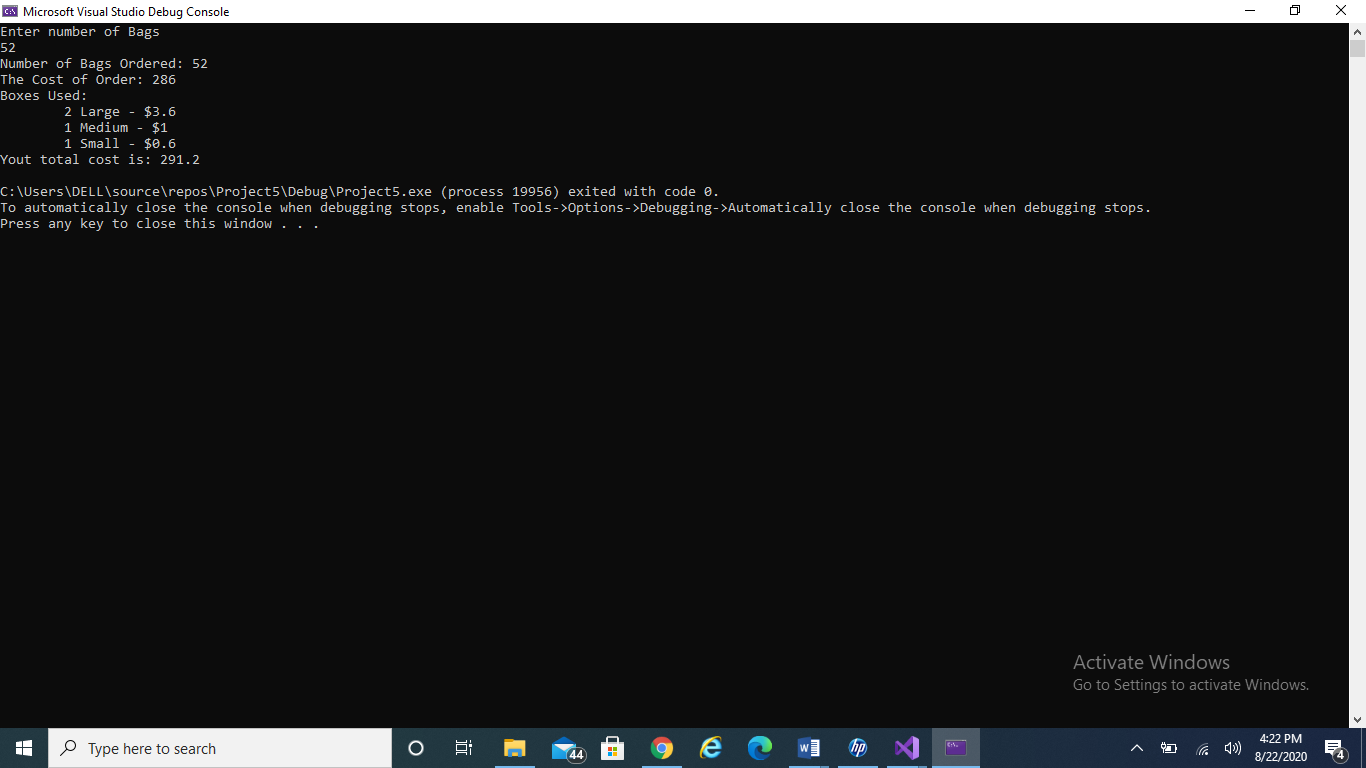
cin >> nb;

coffee c1(nb);

c1.displayinfo();

}

OUTPUT:



**Lab Task 5**

**Q1)** You are already familiar with the Rectangle and DormRoom object types and have also written class descriptions for the two classes. Write down C++ class declarations for both the class.

PROGRAM:

class dormRoom{

private:

int beds,desks,lights,chairs,fans,days;

int rentPerDay;

public:

dormRoom()

{

rentPerDay = 3500;

}

//getters

int getbeds()

{

return beds;

}

int getdesks()

{

return desks;

}

int getlights()

{

return lights;

}

int getchairs()

{

return chairs;

}

int getfans()

{

return fans;

}

int getdays()

{

return days;

}

//setters

void setbeds(int beds)

{

this->beds = beds;

}

void setdesks(int desks)

{

this->desks = desks;

}

void setlights(int lights)

{

this->lights = lights;

}

void setchairs(int chairs)

{

this->chairs = chairs;

}

void setfans(int fans)

{

this->fans = fans;

}

void setdays(int days)

{

this->days = days;

}

float setngetRent()

{

float rent = rentPerDay \* days;

return rent;

}

void output()

{

cout << "Beds : " << getbeds() << "\nDesks : " << getdesks() << "\nChairs : " << getchairs();

cout << "\nLights : " << getlights() << "\nFans : " << getfans() << "\nRent for " << getdays() << " days is :";

cout << setngetRent();

}

};

class rectangle

{

private:

float length, width;

public:

rectangle(float length, float width) {

this->length = length;

this->width = width;

}

//getter

float getl() {

return length;

}

float getw() {

return width;

}

//setter

void setl(float length) {

this->length = length;

}

void setw(float width) {

this->width = width;

}

inline float calcArea() {

return length \* width;

}

};

int main()

{

cout << "Class Dorm Room :\n-----------------\n";

dormRoom dr;

dr.setbeds(3);

dr.setdesks(2);

dr.setlights(4);

dr.setchairs(3);

dr.setfans(2);

dr.setdays(4);

dr.output();

cout << "\n-----------------\nClass Rectangle :\n-----------------\n";

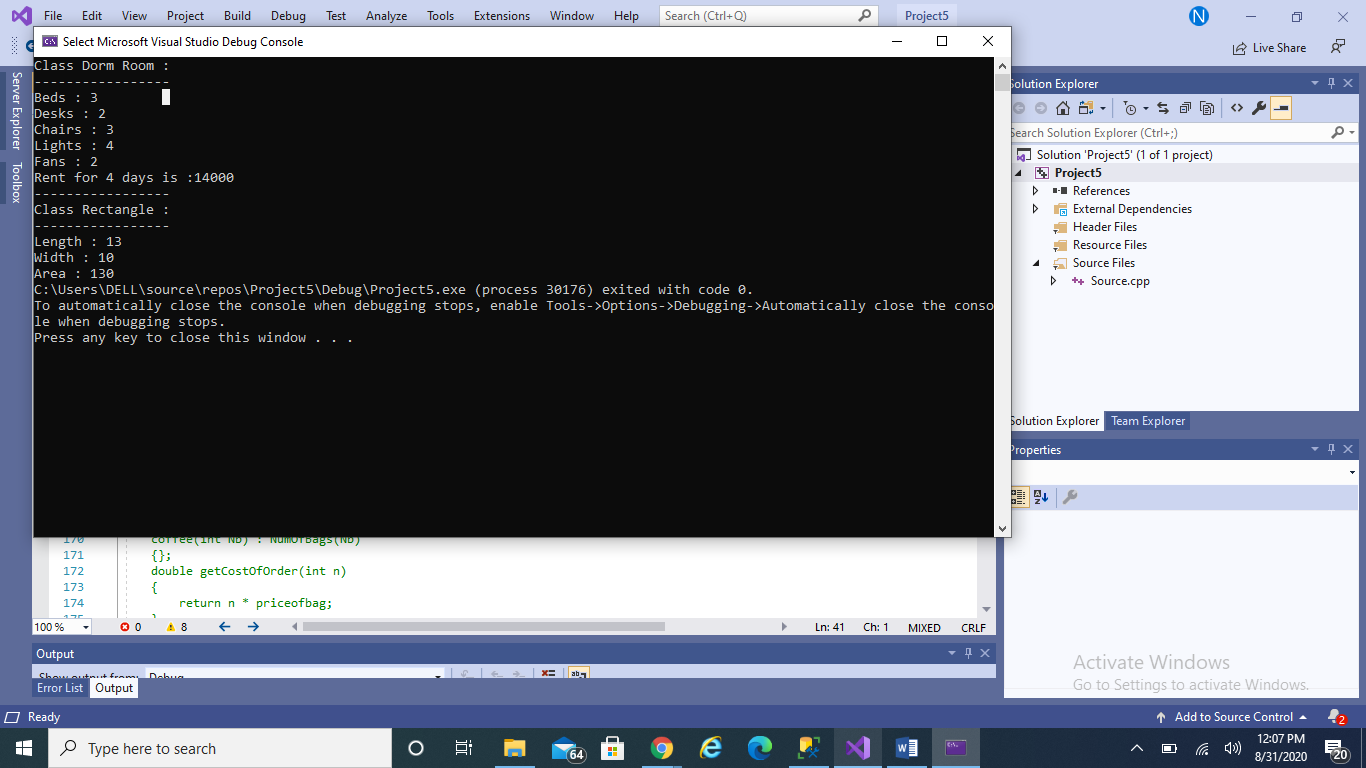
rectangle r(13, 10);

cout << "Length : " << r.getl();

cout << "\nWidth : " << r.getw();

cout << "\nArea : " << r.calcArea();

OUTPUT:



**Q2)** To produce the software for this machine write down the class description and declaration for the MealBill class.

PROGRAM:

class MealBill

{

private:

float MealCost, Tax, Tip, TotalCost, Payment, Change;

public:

float GetMealCost()

{

return MealCost;

}

void SetMealCost(float m)

{

MealCost = m;

}

float CalculateTax()

{

return (7.5 / 100) \* MealCost;

}

float CalculateTip()

{

return 0.15 \* MealCost;

}

float CalculateTotalAmount()

{

return MealCost + CalculateTax() + CalculateTip();

}

/\*Meal cost xx.xx

Tax xx.xx

Tip xx.xx

Total cost xx.xx\*/

void ShowBill()

{

cout << "\n-----------------\nBill\n-----------------\n" << endl;

cout << "Tax " << CalculateTax() << endl;

cout << "Tip " << CalculateTip() << endl;

cout << "Total cost " << CalculateTotalAmount() << endl;

}

/\*Receipt

Total cost xx.xx

Payment xx.xx

Change xx.xx

Thank you for dining at& lt; restaurant name& gt; .\*/

void ShowReceipt(float p, string r)

{

cout << "\n-----------------\nReceipt\n-----------------\n" << endl;

cout << "\Total cost " << CalculateTotalAmount() << endl;

cout << "Payment " << p << endl;

cout << "Change " << p - CalculateTotalAmount() << endl;

cout << "Thank you for dining at " << r << endl;

cout << CalculateTip() << endl;

}

};

int main()

{

MealBill MB1;

float meal;

float paymment;

string name;

cout << "Enter the cost of meal " << endl;

cin >> meal;

MB1.SetMealCost(meal);

MB1.ShowBill();

cout << endl<< endl<< "Enter name of the restraunt" << endl;

cin >> name;

cout << "Enter customer payment" << endl;

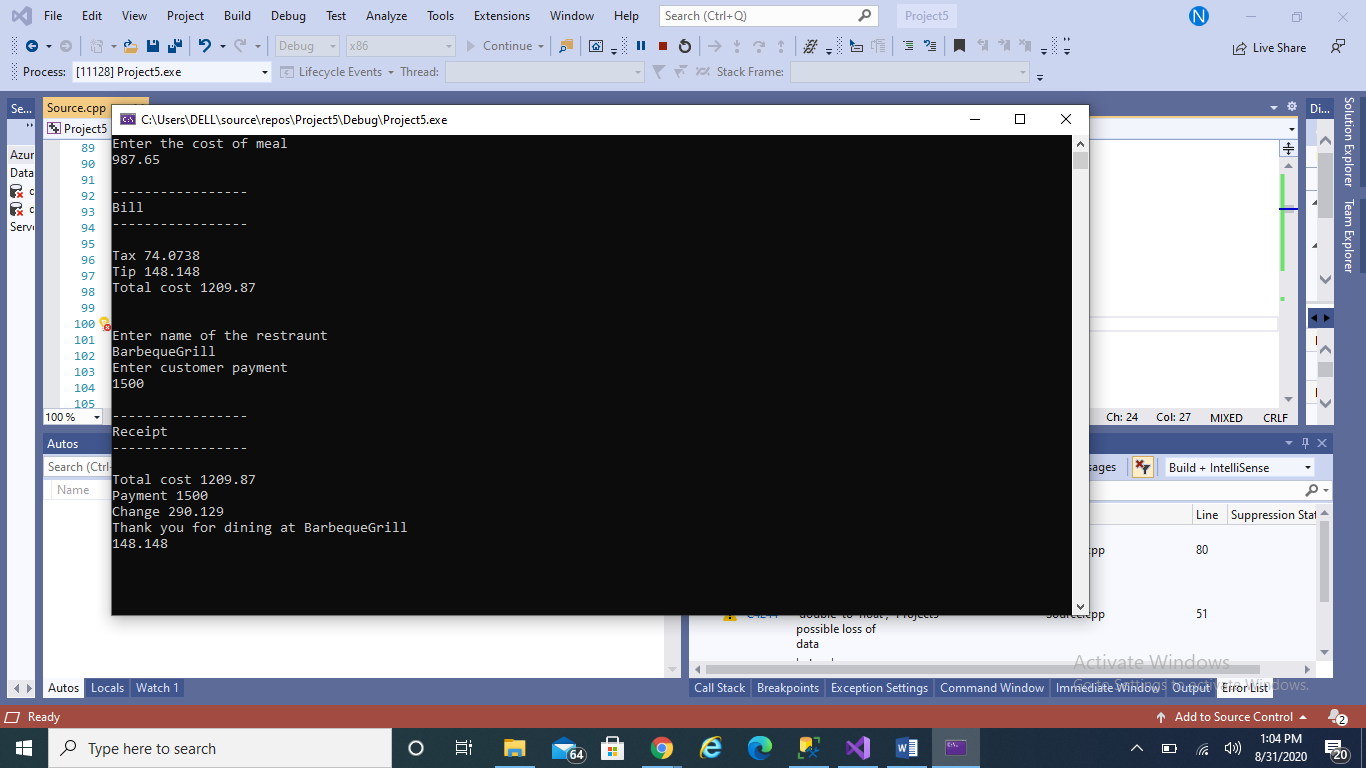
cin >> paymment;

\_getch();

return(0);

}

OUTPUT:



**Q3)** Write a program for The University Summit Restaurant, which allows a student

Ahmed Ali to charge the cost of a meal to his student account.

PROGRAM:

class MealBill

{

private:

float MealCost, Tax, Tip, TotalCost, Payment, Change,Balance;

string ID, Name;

public:

float GetMealCost()

{

return MealCost;

}

void SetMealCost(float m)

{

MealCost = m;

}

float GetTax()

{

return Tax;

}

void SetTax(float t)

{

Tax = t;

}

float CalculateTax()

{

return (Tax / 100) \* MealCost;

}

float CalculateTip()

{

return 0.15 \* MealCost;

}

float CalculateTotalAmount()

{

return MealCost + CalculateTax() + CalculateTip();

}

void ShowBill(string n)

{

cout << "Amount of " << n << "'s Bill:" << CalculateTotalAmount()<< endl;

cout << "\n-----------------\nBill\n-----------------\n" << endl;

cout << "Tax " << CalculateTax() << endl;

cout << "Tip " << CalculateTip() << endl;

cout << "Total cost " << CalculateTotalAmount() << endl;

}

void ShowReceipt(float p, string r, string n)

{

cout << "Charge the meal to " << n << "'s Account:" << endl;

cout << "\n-----------------\nReceipt\n-----------------\n" << endl;

cout << "\Total cost " << CalculateTotalAmount() << endl;

cout << "Payment " << p << endl;

cout << "Change " << p - CalculateTotalAmount() << endl;

cout << "Thank you for dining at " << r << endl;

cout << CalculateTip() << endl;

}

void ShowCurrentBalance(string id, string n,string FN, float B = 2000)

{

cout << "\n-----------------\nCurrent Status of " << n << "'s account:\n-----------------\n" << endl;

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

cout << "Name: " << FN << endl;

cout << "Balance: " << B - CalculateTotalAmount() << endl;

}

};

int main()

{

MealBill AhmedAli;

float meal;

float paymment,tax;

string Restaunt;

string stdName, fName;

string id;

cout << "Enter the cost of meal " << endl;

cin >> meal;

AhmedAli.SetMealCost(meal);

cout << "Enter sale of tax in percentage " << endl;

cin >> tax;

AhmedAli.SetTax(tax);

cout << "Enter student name" << endl;

cin >> stdName;

cout << "Enter Student ID" << endl;

cin >> id;

cout << "Enter Full name" << endl;

cin >> fName;

AhmedAli.ShowBill(stdName);

cout << endl << endl;

cout << "Enter name of the restraunt" << endl;

cin >> Restaunt;

cout << "Enter customer payment" << endl;

cin >> paymment;

AhmedAli.ShowReceipt(paymment, Restaunt, stdName);

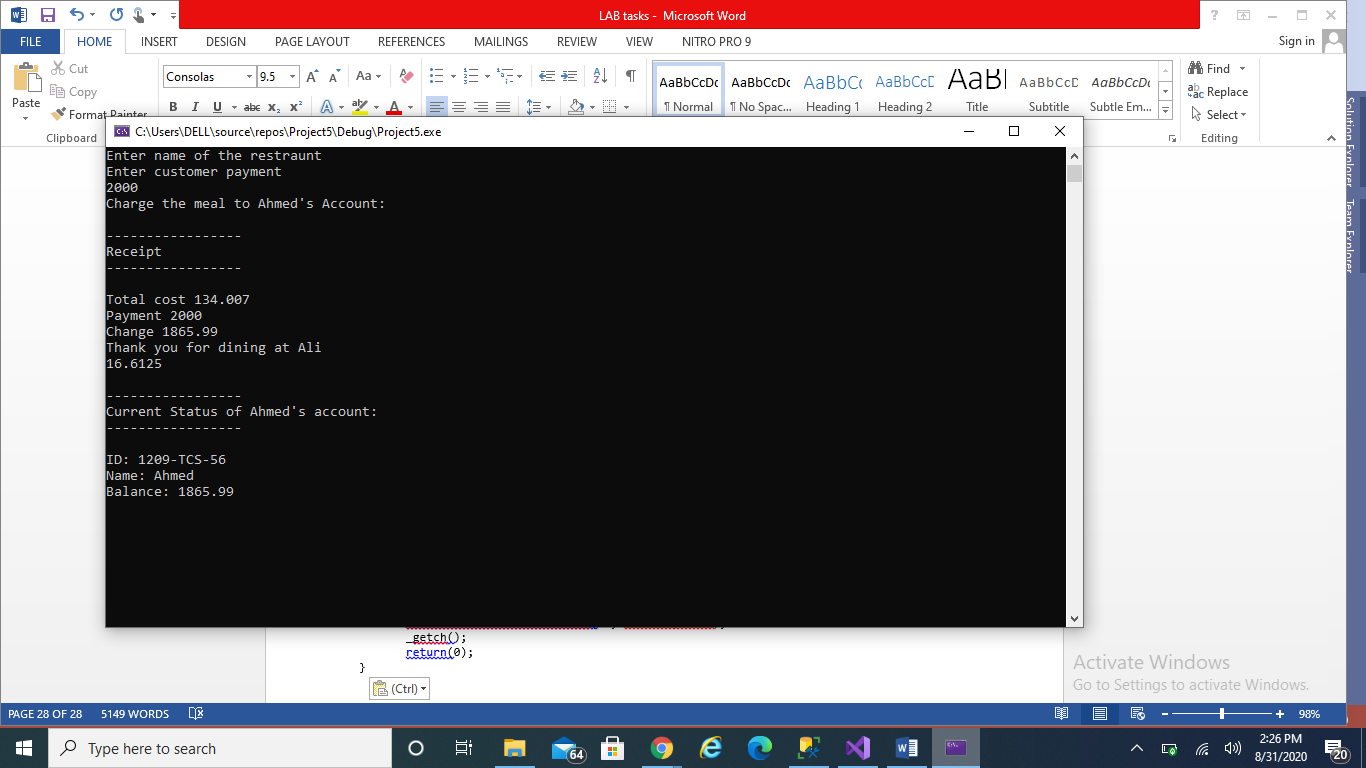
AhmedAli.ShowCurrentBalance(id, stdName,fName);

\_getch();

return(0);

}

OUTPUT:



**Q4)** A student’s grade record is initialized with his/her ID, number of credits and the grade points earned. The number of credits and grade points default to 0 (GPA 0.0), for anew

student. The student record is used for the purpose of determining the GPA – the total credits and the grade points are used to compute the GPA.

PROGRAM:

class record {

private:

string id, name;

float creditsAttempted = 0;

float gradePts = 0;

float GPA = 0;

public:

string getname() {

return name;

}

record(string n, string id, float cA, float gPts) : name(n), id(id), creditsAttempted(cA), gradePts(gPts)

{};

void setter(float creditsAttempted, float gradePts) {

creditsAttempted += creditsAttempted;

gradePts += gradePts;

}

void setCredits(float c)

{

creditsAttempted += c;

}

float getCredits()

{

return creditsAttempted;

}

void setPoints(float p)

{

gradePts += p;

}

float getGrade()

{

return gradePts;

}

float gpaCalc() {

GPA = gradePts / creditsAttempted;

return GPA;

}

void showdata()

{

cout << """Student:" << id << " Units:" << creditsAttempted << " GradePts:" << gradePts << " GPA:" << gpaCalc()<< endl;

}

};

int main()

{

record student1("Nimrah", "SE-0021377", 102, 350);

cout << "\n----------------\nFIRST SEMESTER\n----------------\n";

student1.showdata();

// for next semester

student1.setCredits(16);

student1.setPoints(60);

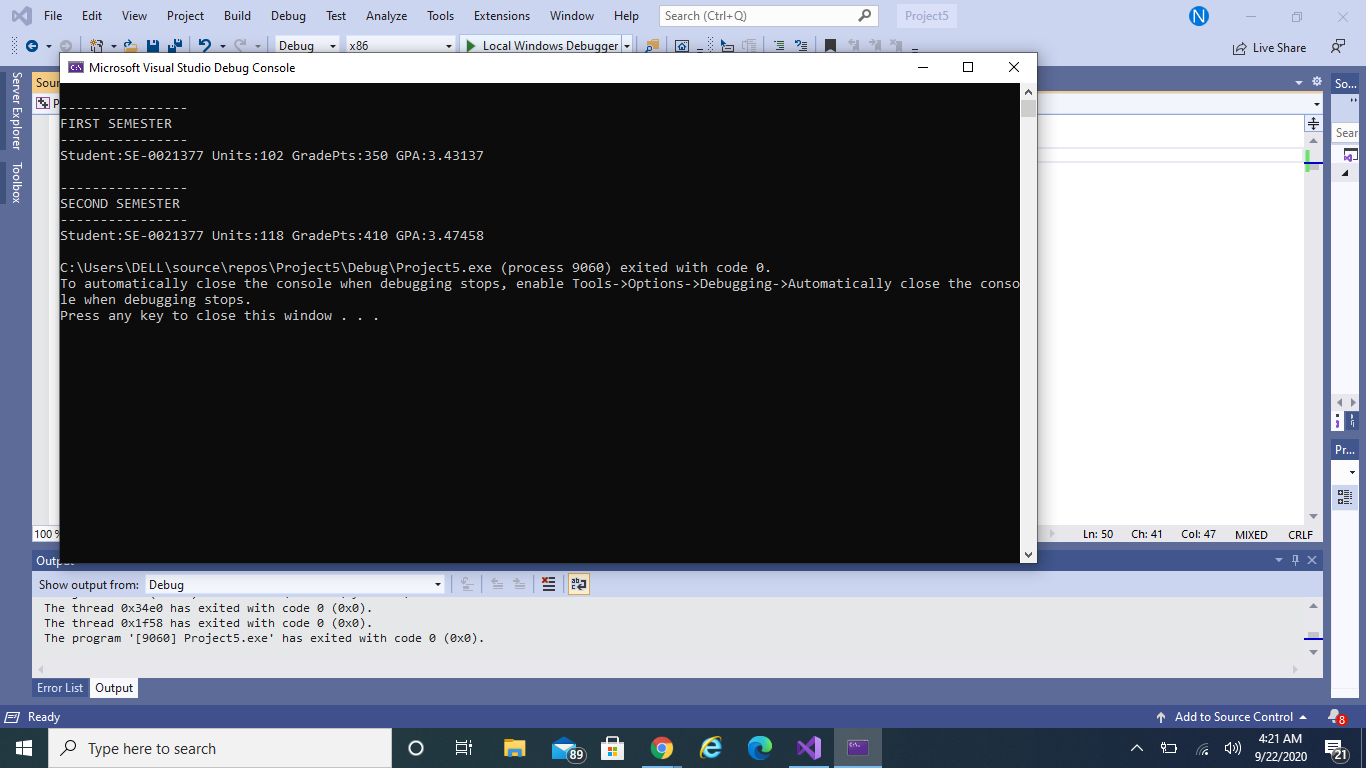
cout << "\n----------------\nSECOND SEMESTER\n----------------\n";

student1.getCredits();

student1.showdata(); // outputs the total GPA for 2 semesters

}

OUTPUT:



**Q5)** Write a program that creates a student grade record for a new student.

PROGRAM:

class record {

private:

string id, name;

float creditsAttempted = 0;

float gradePts = 0;

float GPA = 0.00;

public:

string getname() {

return name;

}

record(string n, string id, float cA, float gPts) : name(n), id(id), creditsAttempted(cA), gradePts(gPts)

{};

void setter(float creditsAttempted, float gradePts) {

creditsAttempted += creditsAttempted;

gradePts += gradePts;

}

void setCredits(float c)

{

creditsAttempted += c;

}

float getCredits()

{

return creditsAttempted;

}

void setPoints(float p)

{

gradePts += p;

}

float getGrade()

{

return gradePts;

}

float getGpa()

{

return GPA;

}

float gpaCalc() {

GPA = gradePts / creditsAttempted;

return GPA;

}

void showdata()

{

cout << "Student:" << id << " Units:" << creditsAttempted << " GradePts:" << gradePts << " GPA:" << gpaCalc()<< endl;

}

};

int main()

{

record student1("Nimrah", "SE-0021377", 102, 350);

cout << student1.getname() << "'s GPA is: " << student1.getGpa() << endl;

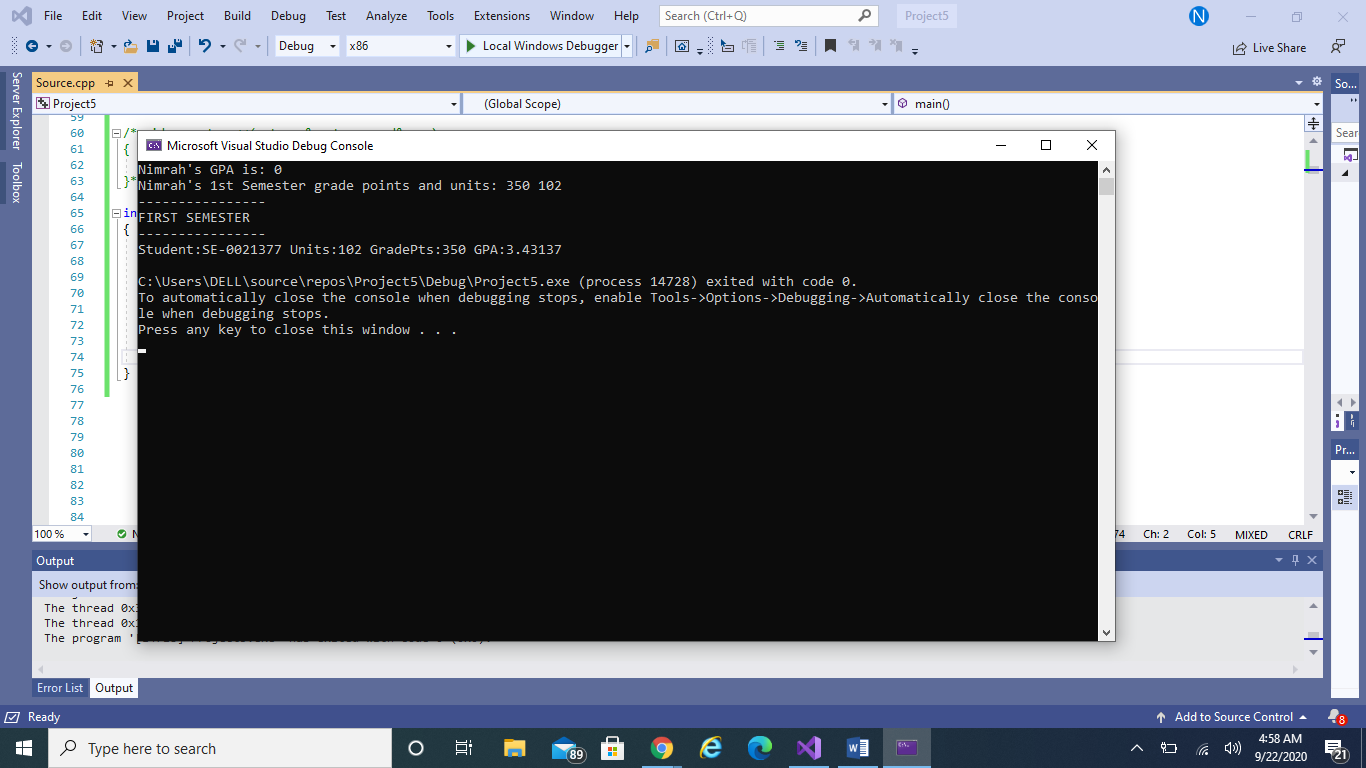
cout << student1.getname() << "'s 1st Semester grade points and units: " << student1.getGrade() << " " << student1.getCredits();

cout << "\n----------------\nFIRST SEMESTER\n----------------\n";

student1.showdata();

}

OUTPUT:



**Q6)** Create a Base Class Person having attributes( name,age,gender) with behavior of

showdata(displaying all the members) and override same method in derived classes.

Derive student Class and from student class derive graduate student with appropriate

data members and behavior.

PROGRAM:

class person {

protected:

string name;

int age;

string gender;

public:

virtual void showdata() = 0;

};

class student :protected person {

protected:

int studentId;

float studentGpa;

public:

student() {

}

student(string name, int age, string gender, int studentId, float studentGpa) {

this->name = name;

this->age = age;

this->gender = gender;

this->studentId = studentId;

this->studentGpa = studentGpa;

}

void showdata() {

cout << "Name of Student : " << name;

cout << "\nAge of Student : " << age;

cout << "\nGender of Student : " << gender;

cout << "\nStudent ID : " << studentId;

cout << "\nStudent GPA : " << studentGpa;

}

};

class graduateStudent :student {

private:

string majorSubject;

public:

graduateStudent(string name, int age, string gender, int studentId, float studentGpa, string majorSubject) {

this->name = name;

this->age = age;

this->gender = gender;

this->studentId = studentId;

this->studentGpa = studentGpa;

this->majorSubject = majorSubject;

}

void showdata() {

cout << "Name of Student : " << name;

cout << "\nAge of Student : " << age;

cout << "\nGender of Student : " << gender;

cout << "\nStudent ID : " << studentId;

cout << "\nStudent GPA : " << studentGpa;

cout << "\nMajor Subject : " << majorSubject;

}

};

int main()

{

student s("Nimrah", 20, "Female", 77, 3.7);

graduateStudent gs("Iqra", 20, "Female", 87, 3.5, "Software Engineering");

cout << "\n----------------------------------------\nStudent's Data:\n----------------------------------------\n";

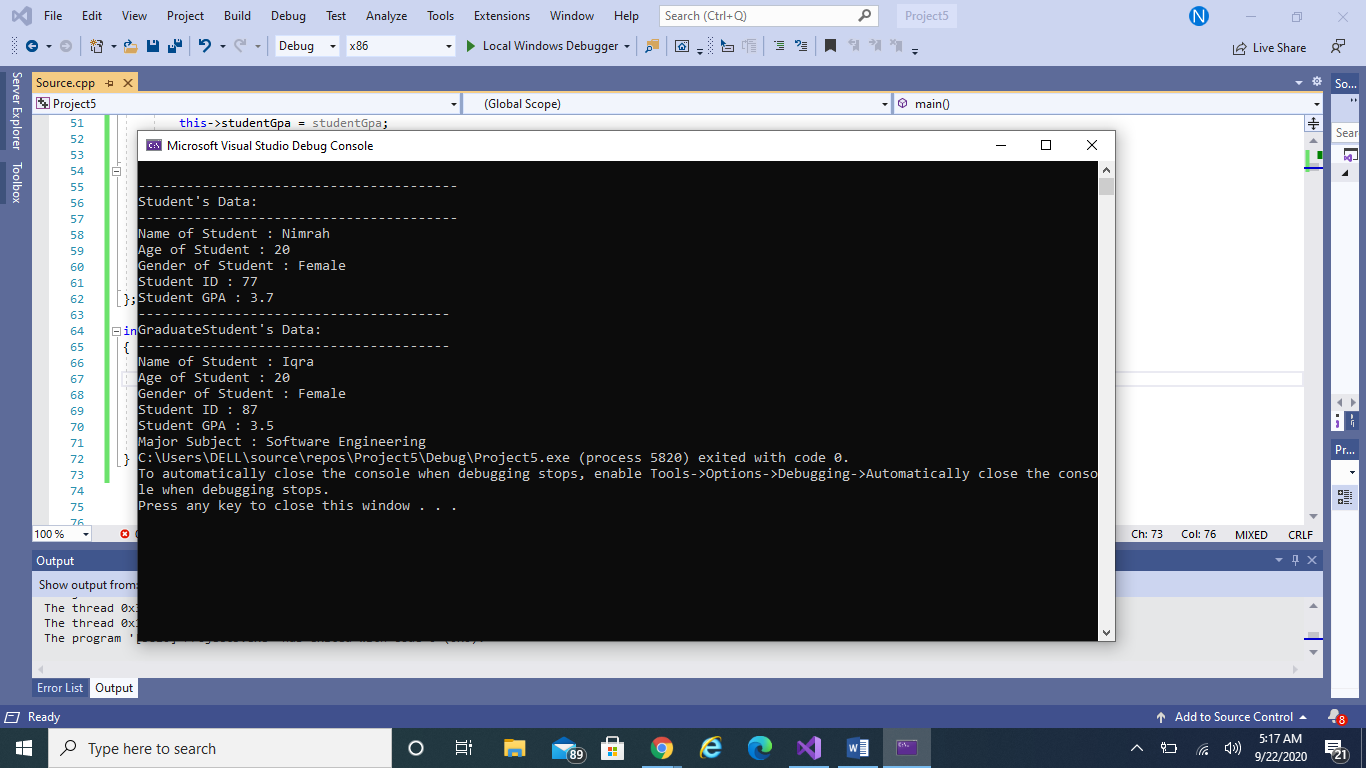
s.showdata();

cout << "\n---------------------------------------\nGraduateStudent's Data:\n---------------------------------------\n";

gs.showdata();

}

OUTPUT:



**LABTASK 6**

**Q1)** Write the classes below containing the given inheritance variables and methods, following the inherited hiearchchy.

PROGRAM:

class circle

{

protected:

double radius;

public:

circle()

{ }

circle(double radius)

{

this->radius=radius;

}

void setRadius(double radius)

{

this->radius=radius;

}

double getRadius()

{

return radius;

}

virtual double calcArea()

{

double area = pi\*radius\*radius;

return area;

}

double calcCircumference()

{

double circumference = 2\*pi\*radius;

return circumference;

}

virtual bool equals(circle\* c)

{ }

virtual string toString()

{

ostringstream ro,ao,co ;

ro << radius;

ao << circle::calcArea();

co << circle::calcCircumference();

string r = ro.str();

string a = ao.str();

string c = co.str();

string result = "Radius = " + r + "\nArea = " + a +"\nCircumference = "+ c ;

return result;

}

};

class cylinder:circle

{

private:

double height;

public:

cylinder()

{

}

cylinder(double radius,double height)

{

this->radius=radius;

this->height=height;

}

void setHeight(double height)

{

this->height=height;

}

double getHeight()

{

return height;

}

double calcArea()

{

return (2\*pi\*radius\*height)+(2\*pi\*radius\*radius);

}

double calcVolume()

{

return pi\*radius\*radius\*height;

}

string toString()

{

ostringstream ro;

ostringstream ho;

ostringstream ao;

ostringstream vo;

ro << radius;

ho << height;

ao << calcArea();

vo << calcVolume();

string r = ro.str();

string h = ho .str();

string a = ao .str();

string v = vo .str();

string result = "Radius = "+r+"\nHeight = "+h+"\nArea = "+a+"\nVolume = "+v;

return result;

}

bool equals(cylinder\* cy)

{ }

};

int main()

{

cout << "\n-----------------\nFor Circle:\n-----------------\n";

circle c(3);

cout << c.toString();

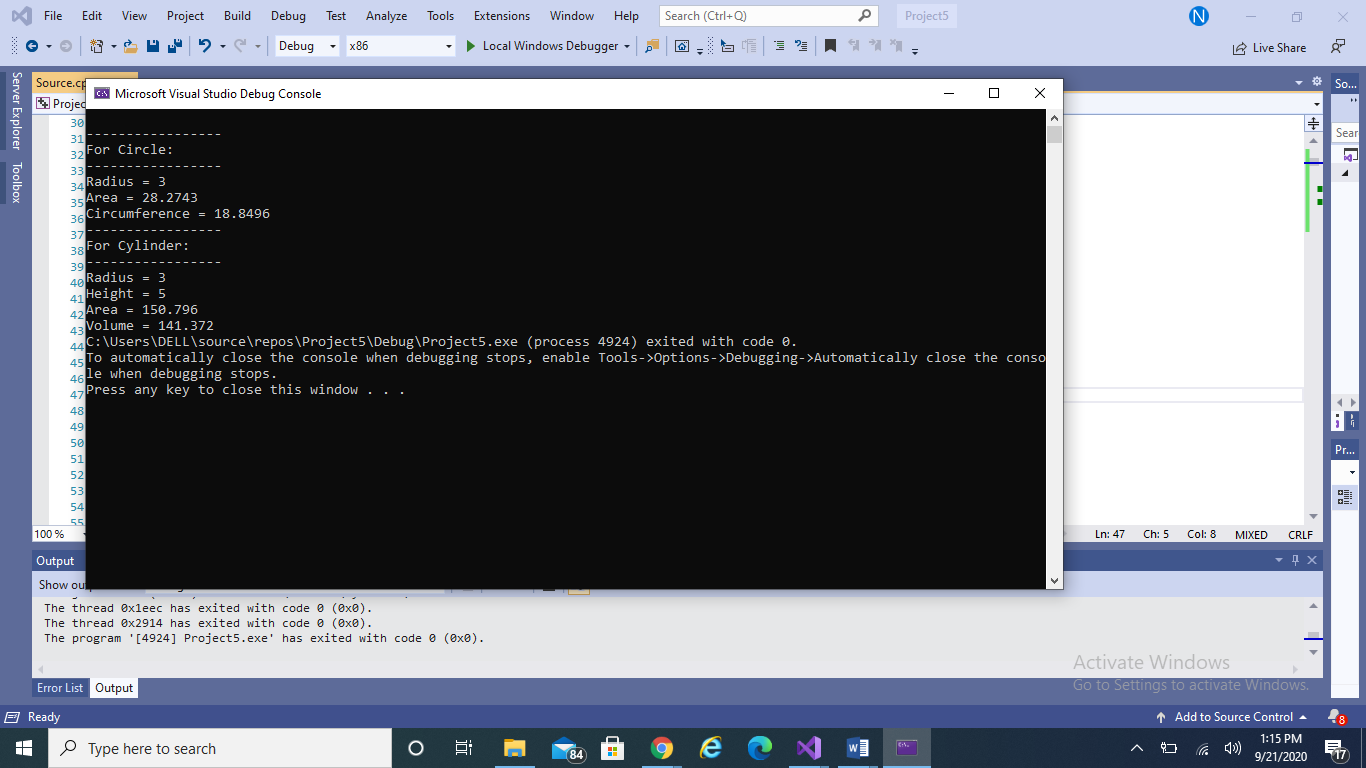
cout << "\n-----------------\nFor Cylinder:\n-----------------\n";

cylinder cy(3, 5);

cout << cy.toString();

}

OUTPUT:



**Q2)** Apply the Concept of Composition by Creating part classes (doors,windows,Engine,Wheels) and then Create a whole class Car that is composed of all these defined parts.

PROGRAM:

class door

{

private:

int length;

int width;

public:

door()

{

};

door(int l, int w) : length(l), width(w)

{};

int getlength()

{

return length;

}

void setlength(int l)

{

length = l;

}

int getwidth()

{

return width;

}

void setwidth(int w)

{

width = w;

}

void showdata()

{

cout << "Door Length: " << length << endl;

cout << "Door Width: " << width << endl;

}

};

class window

{

private:

int length;

int width;

public:

window()

{

};

window(int l, int w) : length(l), width(w)

{};

int getlength()

{

return length;

}

void setlength(int l)

{

length = l;

}

int getwidth()

{

return width;

}

void setwidth(int w)

{

width = w;

}

void showdata()

{

cout << "Window Length: " << length << endl;

cout << "Window Width: " << width << endl;

}

};

class engine

{

private:

int size;

int model;

string type;

public:

engine()

{};

engine(int s, int m, string t) : size(s), model(m), type(t)

{};

;

int getsize()

{

return size;

}

void setlsize(int s)

{

size = s;

}

int getmodel()

{

return model;

}

void setmodel(int m)

{

model = m;

}

string gettype()

{

return type;

}

void settype(string t)

{

type = t;

}

void showdata()

{

cout << "Engine Size: " << size << endl;

cout << "Engine Model: " << model << endl;

cout << "Engine Type: " << type << endl;

}

};

class wheels

{

private:

int size;

int model;

string color;

public:

wheels()

{};

wheels(int s, int m, string c) : size(s), model(m), color(c)

{};

;

int getsize()

{

return size;

}

void setlsize(int s)

{

size = s;

}

int getmodel()

{

return model;

}

void setmodel(int m)

{

model = m;

}

string getcolor()

{

return color;

}

void setcolor(string c)

{

color = c;

}

void showdata()

{

cout << "Wheel Size: " << size << endl;

cout << "Wheel Model: " << model << endl;

cout << "Wheel Color: " << color << endl;

}

};

class Car

{

private:

door d1, d2, d3, d4;

window w1, w2, w3, w4;

engine e1;

wheels wh1, wh2, wh3, wh4;

public:

Car()

{};

Car(int doorL, int doorW, int windowL, int windowW, int engineS, int engineM, string engineT, int wheelS, int wheelM, string wheelC): d1(doorL, doorW), d2(doorL, doorW), d3(doorL, doorW), d4(doorL, doorW), w1(windowL, windowW), w2(windowL, windowW), w3(windowL, windowW), w4(windowL, windowW), e1(engineS, engineM, engineT), wh1(wheelS, wheelM,wheelC), wh2(wheelS, wheelM, wheelC), wh3(wheelS, wheelM, wheelC), wh4(wheelS, wheelM, wheelC)

{};

void showdata()

{

cout << " CAR DESCRIPTION " << endl << endl;

for (int i=1; i <= 4; i++)

{

cout << "Door " << i << ": " << endl;

d1.showdata();

cout << endl;

}

for (int i = 1; i <= 4; i++)

{

cout << "Window " << i << ": " << endl;

w1.showdata();

cout << endl;

}

cout << "Engine: ";

e1.showdata();

cout << endl;

for (int i = 1; i <= 4; i++)

{

cout << "Wheel " << i << ": " << endl;

wh1.showdata();

cout << endl;

}

}

};

int main()

{

door d1, d2, d3, d4;

window w1, w2, w3, w4;

engine e1;

wheels wh1, wh2, wh3, wh4;

Car c1(10, 20, 5, 6, 40, 8, "Electrical", 15, 4, "Black");

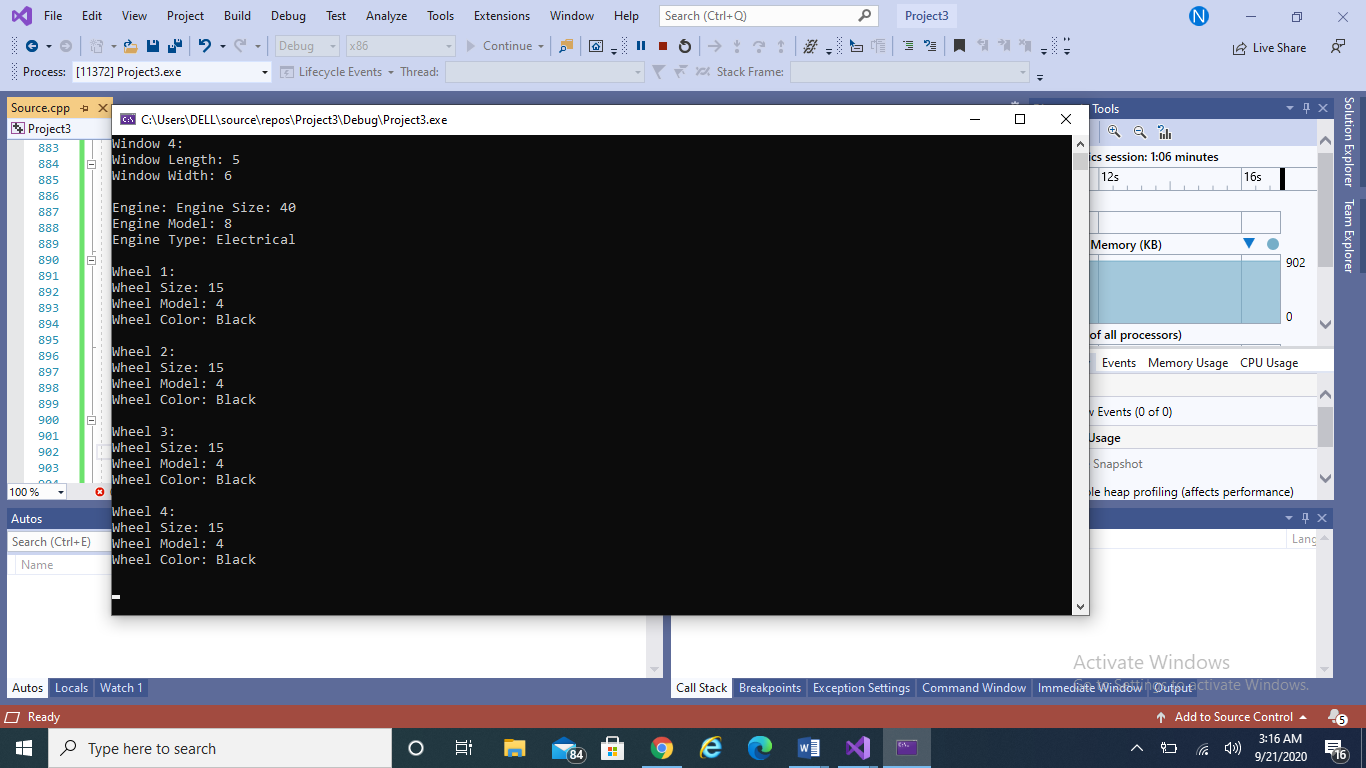
c1.showdata();

\_getch();

return 0;

}





**Q3)** Create Box class containing data members (width ,height,Length) and Member functions showdata and Volume.. Apple the concept of Operator Overloading and overload operators (+,-,\* ,/) for addition subtraction multiplication and division of Objects of Box class.

PROGRAM:

class Box

{

private:

double height;

double width;

double length;

public:

Box()

{};

Box(double h, double w, double l) : height(h), width(w), length(l)

{};

void setheight(double h)

{

height = h;

}

double getheight()

{

return height;

}

void setwidth(double w)

{

width = w;

}

double getwidth()

{

return width;

}

void setlength(double l)

{

length = l;

}

double getlengtht()

{

return length;

}

double voulme()

{

return height \* width \* length;

}

Box operator+(const Box &b)

{

Box box;

box.height = this->height + b.height;// you will add the height of current obeject to the height of past object to a newly defined object

box.width = this->width + b.width;

box.length = this->length + b.length;

return box;

}

Box operator-(const Box& b)

{

Box box;

box.height = this->height - b.height;// you will add the height of current obeject to the height of past object to a newly defined object

box.width = this->width - b.width;

box.length = this->length - b.length;

return box;

}

Box operator\*(const Box& b)//this->-- FUNCTION??

{

Box box;

box.height = this->height \* b.height;// you will add the height of current obeject to the height of past object to a newly defined object

box.width = this->width \* b.width;

box.length = this->length \* b.length;

return box;

}

Box operator/(const Box& b)//this->-- FUNCTION??

{

Box box;

box.height = this->height / b.height;// you will add the height of current obeject to the height of past object to a newly defined object

box.width = this->width / b.width;

box.length = this->length / b.length;

return box;

}

};

int main()

{

Box b1(5.6, 7.8, 9.1);

Box b2(1.2, 4.5, 3.9);

Box b3;

b3 = b1 + b2;

cout << "Details of b3 are: " << endl;

cout << "The height of b3 is: " << b3.getheight() << endl;

cout << "The width of b3 is: " << b3.getwidth() << endl;

cout << "The length of b3 is: " << b3.getlengtht() << endl;

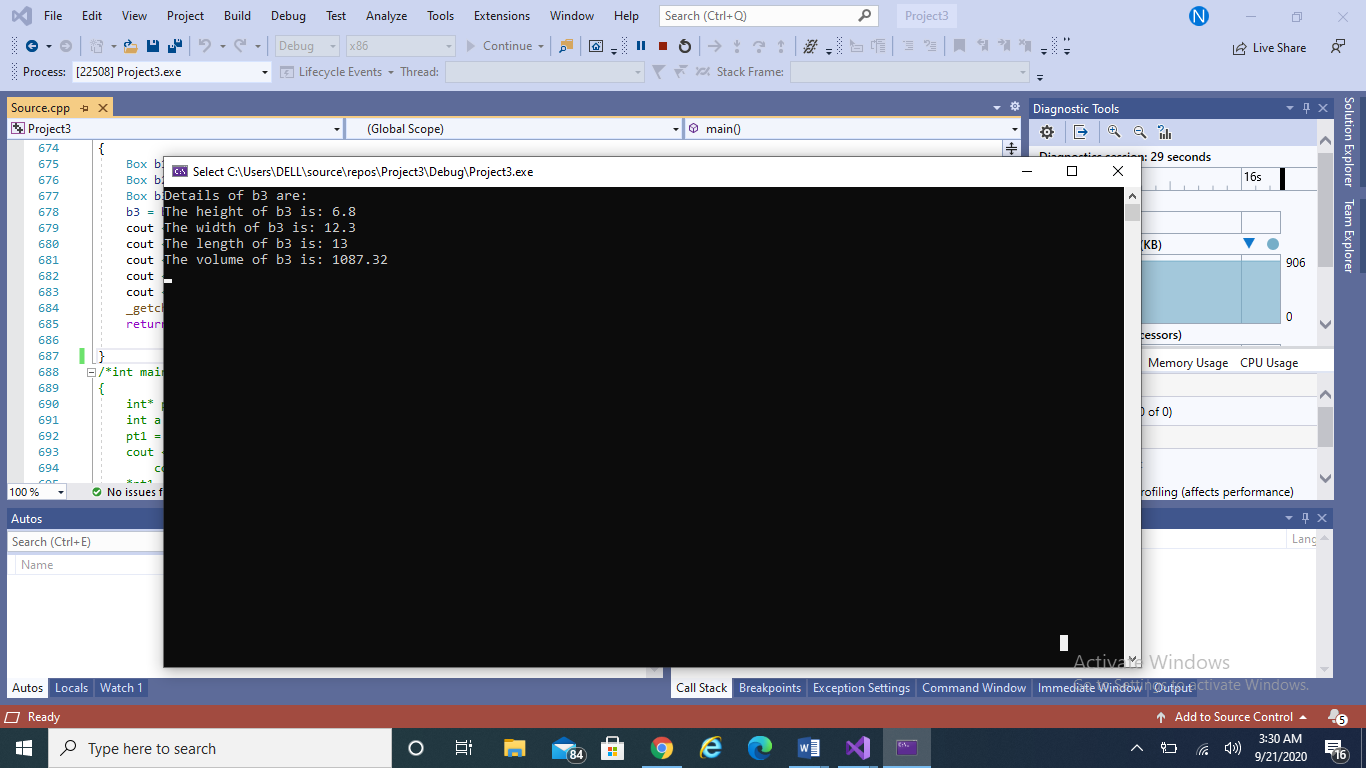
cout << "The volume of b3 is: " << b3.voulme() << endl;

\_getch();

return 0;

}

OUTPUT:



**LABTASK 7**

Create a Calculator class that offers four methods. Add, subtract, multiply and Divide. Consisting of two private members of type double to take input from the user. Create object of a class and start using the calculator class.

PROGRAM:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace WindowsFormsApp5\_WEEK11\_12\_

{

public partial class Form6 : Form

{

public Form6()

{

InitializeComponent();

}

private void button2\_Click(object sender, EventArgs e)

{

txtResult.Text += 1.ToString();

}

private void button3\_Click(object sender, EventArgs e)

{

txtResult.Text += 2.ToString();

}

private void button4\_Click(object sender, EventArgs e)

{

txtResult.Text += 3.ToString();

}

private void button5\_Click(object sender, EventArgs e)

{

txtResult.Text += 4.ToString();

}

private void button6\_Click(object sender, EventArgs e)

{

txtResult.Text += 5.ToString();

}

private void button7\_Click(object sender, EventArgs e)

{

txtResult.Text += 6.ToString();

}

private void button12\_Click(object sender, EventArgs e)

{

txtResult.Text += 7.ToString();

}

private void button10\_Click(object sender, EventArgs e)

{

txtResult.Text += 8.ToString();

}

private void button9\_Click(object sender, EventArgs e)

{

txtResult.Text += 9.ToString();

}

private void button11\_Click(object sender, EventArgs e)

{

txtResult.Text += 0.ToString();

}

int No1;

int No2;

char op;

float result;

private void Add\_Click(object sender, EventArgs e)

{

No1 = Convert.ToInt32(this.txtResult.Text);

txtResult.Clear();

op = '+';

}

private void Equal\_Click(object sender, EventArgs e)

{

if(op == '+')

{

No2 = Convert.ToInt32(this.txtResult.Text);

txtResult.Clear();

result = No1 + No2;

txtResult.Text = result.ToString();

}

else if (op == '-')

{

No2 = Convert.ToInt32(this.txtResult.Text);

txtResult.Clear();

result = No1 - No2;

txtResult.Text = result.ToString();

}

else if (op == '\*')

{

No2 = Convert.ToInt32(this.txtResult.Text);

txtResult.Clear();

result = No1 \* No2;

txtResult.Text = result.ToString();

}

else if (op == '/')

{

No2 = Convert.ToInt32(this.txtResult.Text);

txtResult.Clear();

result = No1 / No2;

txtResult.Text = result.ToString();

}

}

private void Subtract\_Click(object sender, EventArgs e)

{

No1 = Convert.ToInt32(this.txtResult.Text);

txtResult.Clear();

op = '-';

}

private void Multiply\_Click(object sender, EventArgs e)

{

No1 = Convert.ToInt32(this.txtResult.Text);

txtResult.Clear();

op = '\*';

}

private void Divide\_Click(object sender, EventArgs e)

{

No1 = Convert.ToInt32(this.txtResult.Text);

txtResult.Clear();

op = '/';

}

private void button8\_Click(object sender, EventArgs e)

{

txtResult.Clear();

No1 = 0;

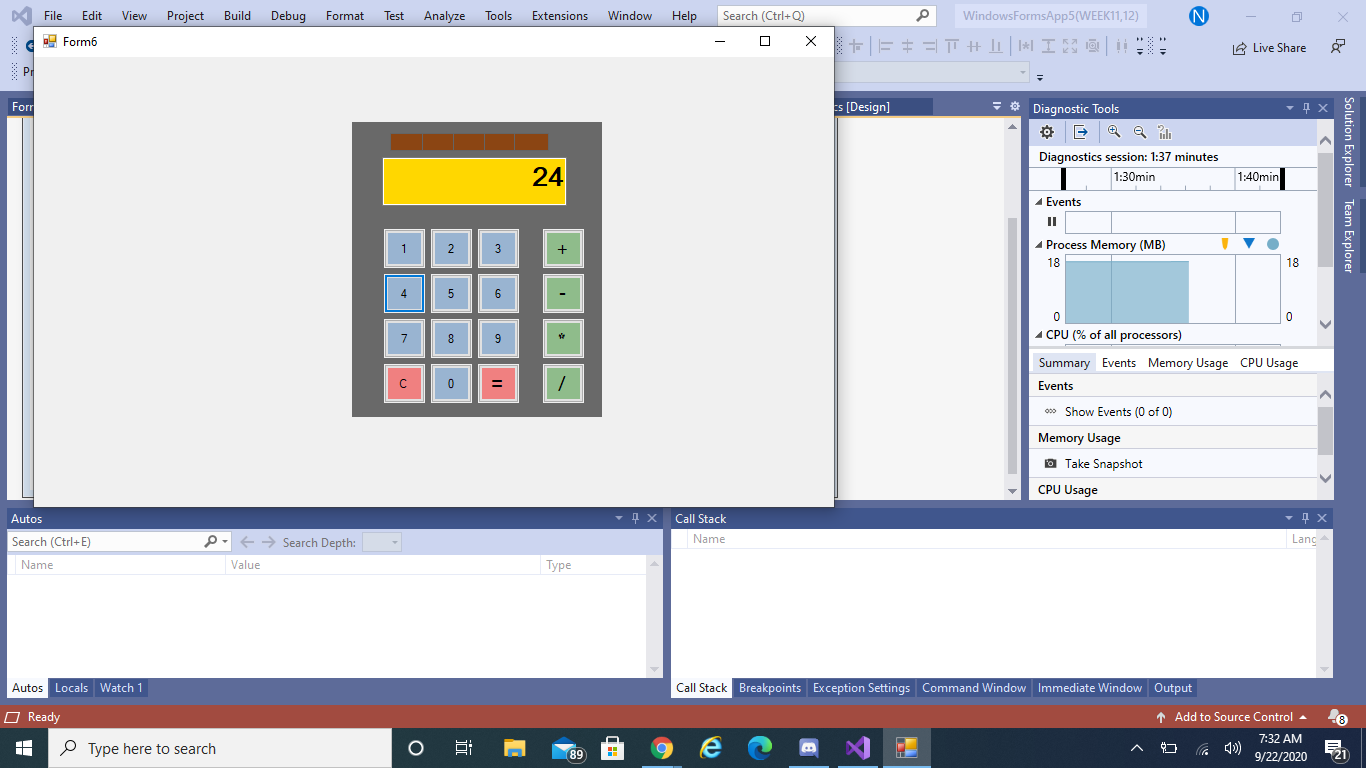
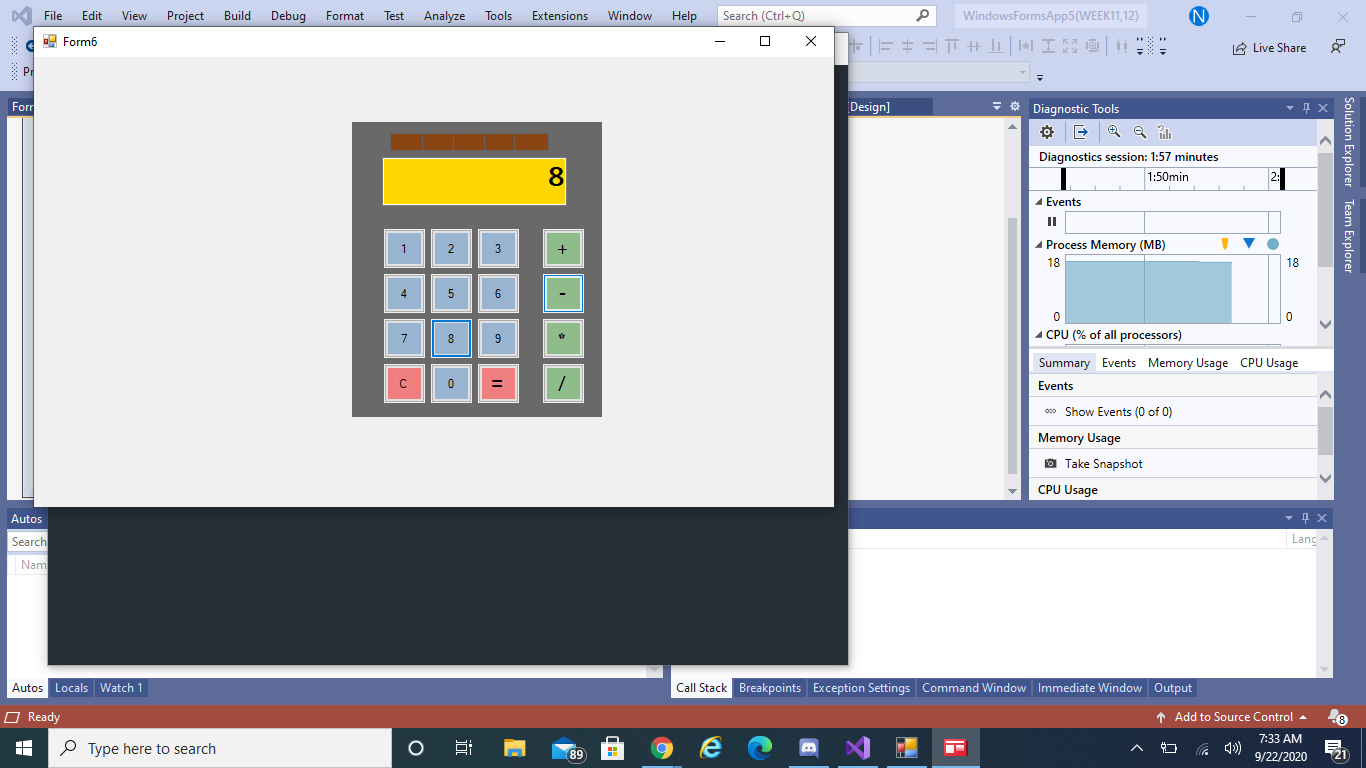
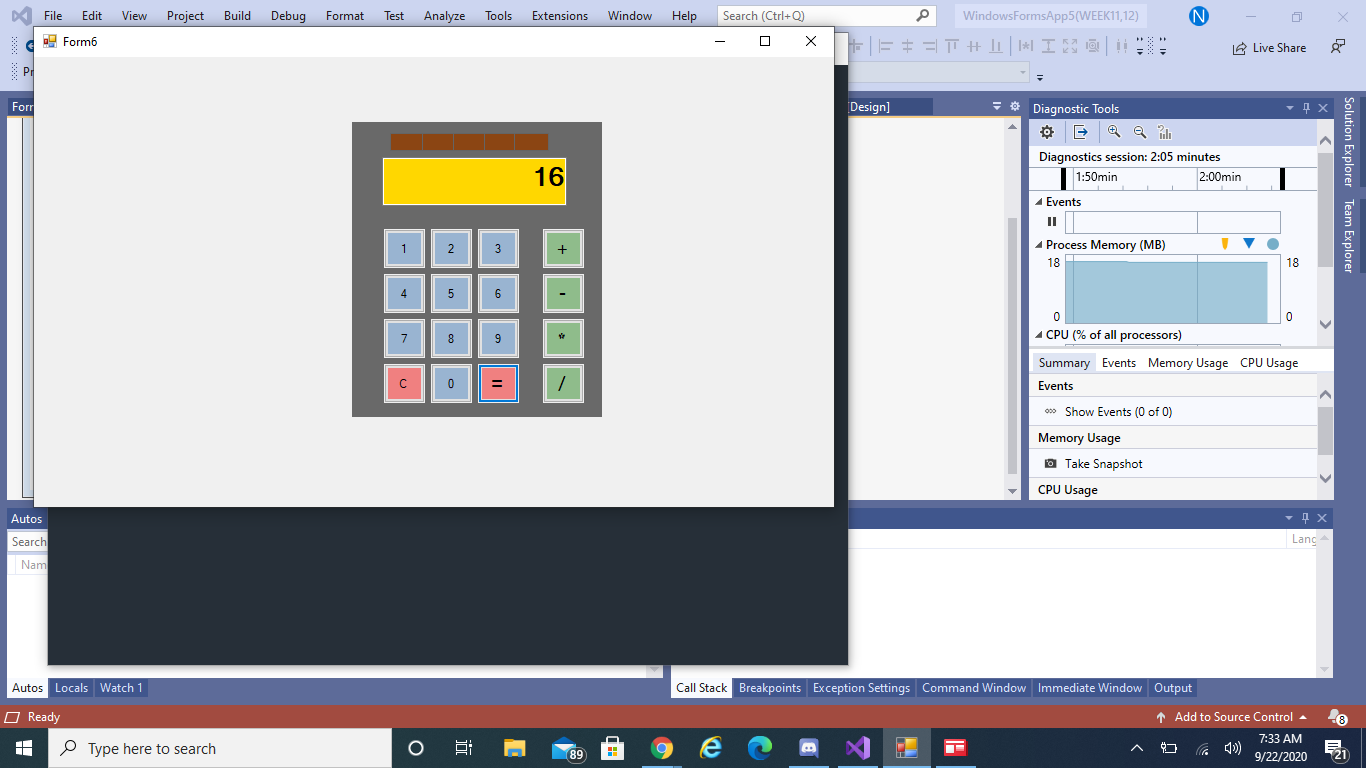
No2 = 0;

}

}

}

OUTPUT:

**LABTASK 8**

**Q1)** Create an array of Buttons that would display buttons dynamically at run time.

PROGRAM:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace WindowsFormsApp5\_WEEK11\_12\_

{

public partial class Form4 : Form

{

public Form4()

{

InitializeComponent();

}

Button[] b = new Button[6];

private void Form4\_Load(object sender, EventArgs e)

{

this.BackColor = Color.WhiteSmoke;

for (int i = 0; i <= b.Length - 1; i++)

{

b[i] = new Button();

b[i].Text = "button" +i;

b[i].Size = new Size(70, 30);

b[i].BackColor = Color.CornflowerBlue;

b[i].Location = new Point(i + 70, i + 100);

b[i].Left = 60 \* i;

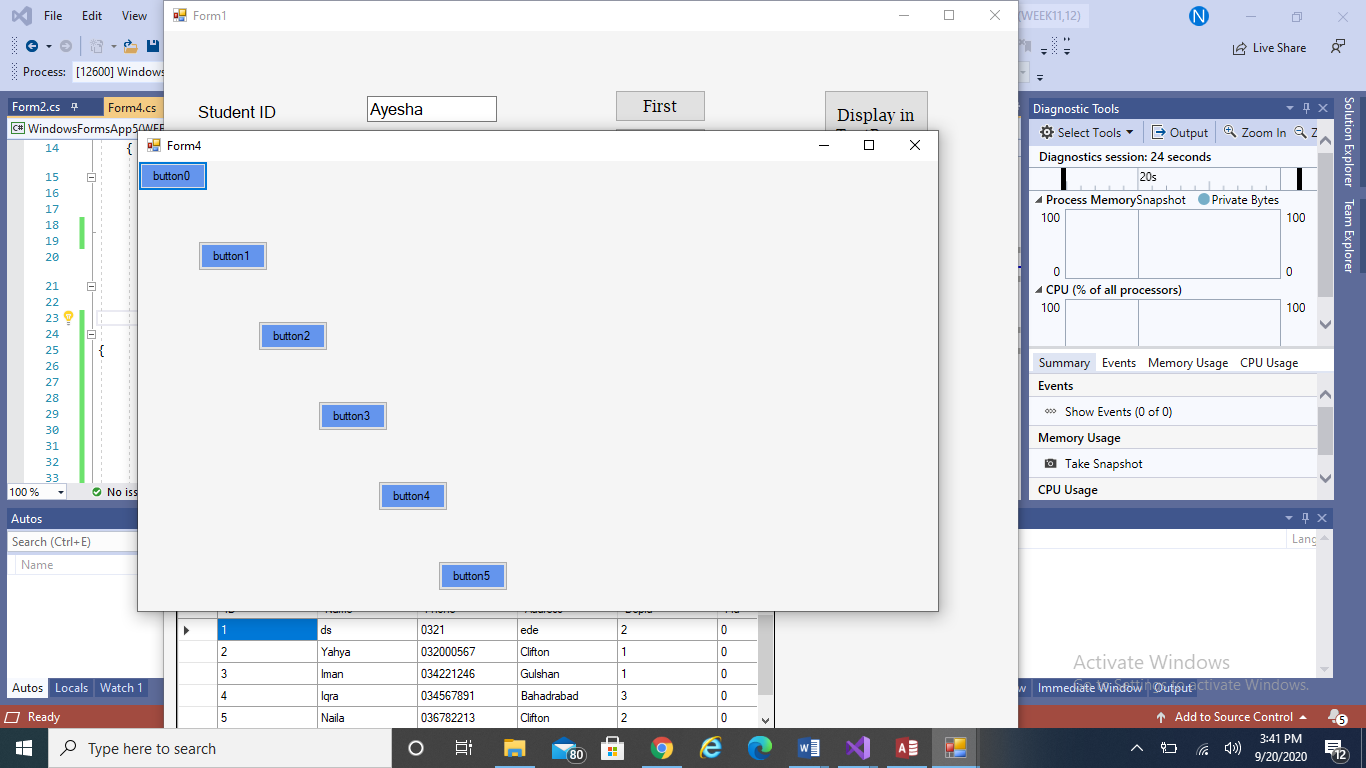
b[i].Top = 80 \* i;

this.Controls.Add(b[i]);

}

}

OUTPUT:



**Q2)** Create a slideshow of pictures by using picture box control and timer control. Make a folder on any drive containing all of your pictures, you want to include in a slideshow. Rename all the pictures starting with 1.jpg, 2.jpg… n.jpg.

PROGRAM:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace WindowsFormsApp5\_WEEK11\_12\_

{

public partial class Form5 : Form

{

public Form5()

{

InitializeComponent();

}

int count = 1;

private void timer1\_Tick(object sender, EventArgs e)

{

if (count == 7)

{

count = 1;

}

pictureBox1.ImageLocation = string.Format(@"d:/{0}.png", count);

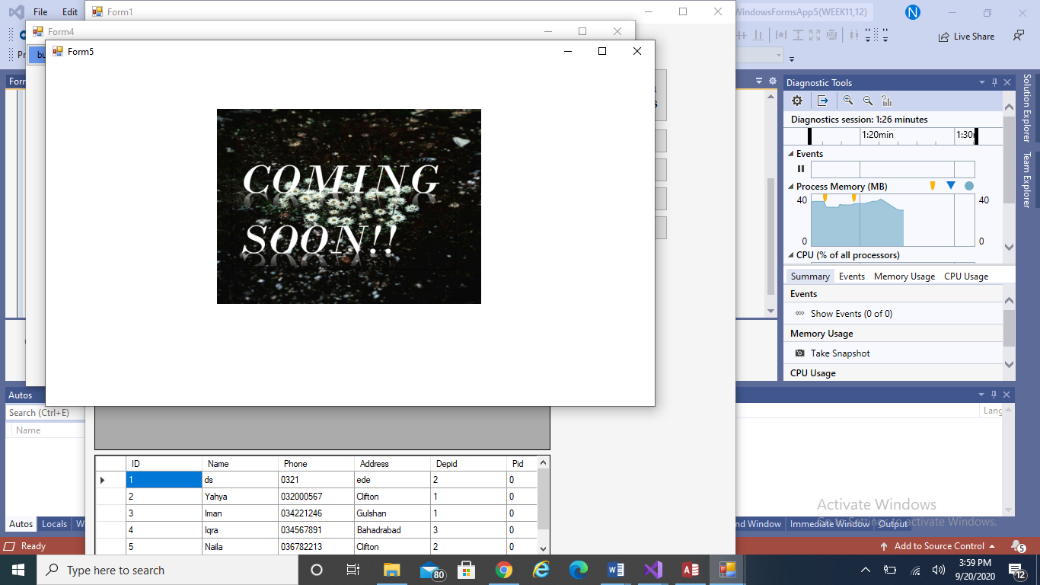
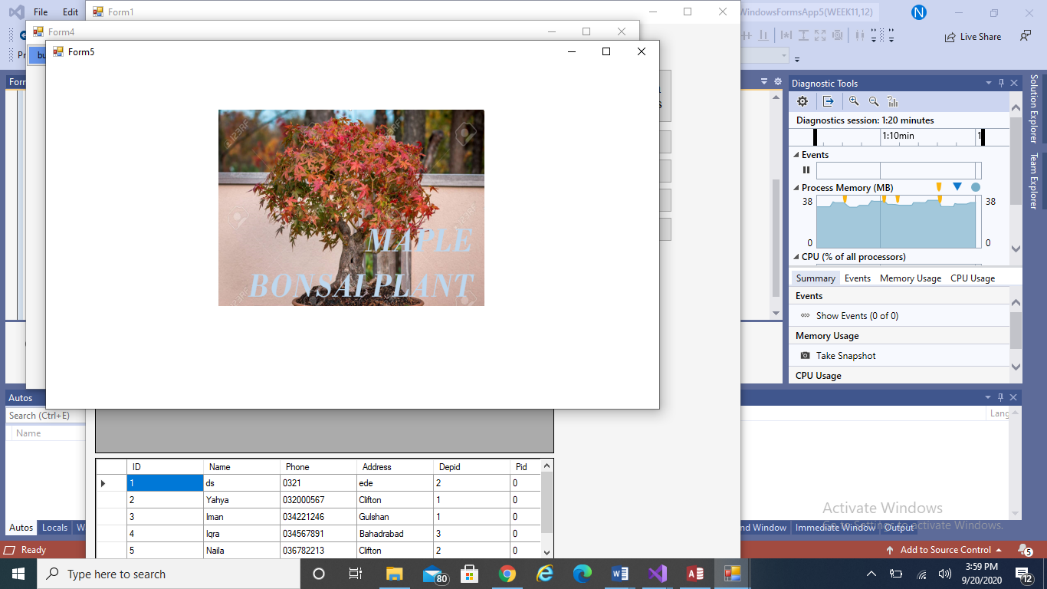
count++;

}

}

}

OUTPUT:

**Q3)** Using Mouse up, down and Mouse move events, Create an application that can draw freehand drawing.

PROGRAM:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace lab\_10

{

public partial class Form3 : Form

{

public Form3()

{

InitializeComponent();

}

bool draw;

private void Form3\_Load(object sender, EventArgs e)

{

}

private void Form3\_MouseMove(object sender, MouseEventArgs e)

{

if (draw == true)

{

Graphics g = base.CreateGraphics();

SolidBrush sb = new SolidBrush(Color.BlueViolet);

g.FillEllipse(sb, e.X, e.Y, 10, 10);

}

}

private void Form3\_MouseDown(object sender, MouseEventArgs e)

{

draw = true;

}

private void Form3\_MouseUp(object sender, MouseEventArgs e)

{

draw = false;

}

}

}

OUTPUT:



**LABTASK 9**

TASK 1

PROGRAM:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.OleDb;

namespace Working\_with\_Dbs\_in\_CSharp

{

public partial class labtask3 : Form

{

OleDbConnection con = new OleDbConnection(&quot; Provider=Microsoft.Jet.OLEDB.4.0;Data

Source = C:/Users/Hp/Desktop/DB/db for c.mdb&quot;);

OleDbDataAdapter adap = new OleDbDataAdapter(&quot; Select\* from student&quot;,

&quot;Provider=Microsoft.Jet.OLEDB.4.0;Data Source = C:/Users/Hp/Desktop/DB/db for c.mdb&quot;);

DataSet ds = new DataSet(&quot; Student&quot;);

int counter = 0;

public labtask3()

{

InitializeComponent();

adap.Fill(ds);

}

private void button1\_Click(object sender, EventArgs e)

{

con.Open();

if (counter & gt; 0)

{

counter = 0;

textBox1.Text = ds.Tables[0].Rows[counter][&quot; ID & quot;].ToString();

textBox2.Text = ds.Tables[0].Rows[counter][&quot; Name & quot;].ToString();

textBox3.Text = ds.Tables[0].Rows[counter][&quot; Phone & quot;].ToString();

textBox4.Text = ds.Tables[0].Rows[counter][&quot; Address & quot;].ToString();

textBox5.Text = ds.Tables[0].Rows[counter][&quot; dptid & quot;].ToString();

}

else

{

MessageBox.Show(&quot; You are already on the first record & quot;);

}

con.Close();

}

private void button13\_Click(object sender, EventArgs e)

{

try

{

con.Open();

OleDbDataAdapter adapt = new OleDbDataAdapter(&quot; select student.ID , dept.dptid from

student inner join dept on dept.dptid = student.dptid & quot;, con);

DataSet ds2 = new DataSet();

adapt.Fill(ds2);

OleDbDataAdapter adapt2 = new OleDbDataAdapter(&quot; select courses.cname,emp.empname

from emp inner join courses on emp.empno = courses.empno & quot;, con);

adapt2.Fill(ds2, &quot; employee & quot;);

dataGrid2.DataSource = ds2;

con.Close();

}

catch (Exception ex)

{

MessageBox.Show(&quot; Error + &quot; +ex);

}

}

private void button9\_Click(object sender, EventArgs e)

{

con.Open();

try

{

textBox1.Text = ds.Tables[0].Rows[0][&quot; ID & quot;].ToString();

textBox2.Text = ds.Tables[0].Rows[0][&quot; Name & quot;].ToString();

textBox3.Text = ds.Tables[0].Rows[0][&quot; Phone & quot;].ToString();

textBox4.Text = ds.Tables[0].Rows[0][&quot; Address & quot;].ToString();

textBox5.Text = ds.Tables[0].Rows[0][&quot; dptid & quot;].ToString();

}

catch (Exception ex)

{

MessageBox.Show(&quot; Error & quot; +ex);

}

con.Close();

}

private void button8\_Click(object sender, EventArgs e)

{

con.Open();

dataGrid1.DataSource = ds;

con.Close();

}

private void button2\_Click(object sender, EventArgs e)

{

con.Open();

if (counter & gt; 0)

{

counter -= 1;

textBox1.Text = ds.Tables[0].Rows[counter][&quot; ID & quot;].ToString();

textBox2.Text = ds.Tables[0].Rows[counter][&quot; Name & quot;].ToString();

textBox3.Text = ds.Tables[0].Rows[counter][&quot; Phone & quot;].ToString();

textBox4.Text = ds.Tables[0].Rows[counter][&quot; Address & quot;].ToString();

textBox5.Text = ds.Tables[0].Rows[counter][&quot; dptid & quot;].ToString();

}

else

{

MessageBox.Show(&quot; You are already on the first record & quot;);

}

con.Close();

}

private void button3\_Click(object sender, EventArgs e)

{

con.Open();

if (counter & lt; ds.Tables[0].Rows.Count - 1)

{

counter = ds.Tables[0].Rows.Count - 1;

textBox1.Text = ds.Tables[0].Rows[0][&quot; ID & quot;].ToString();

textBox2.Text = ds.Tables[0].Rows[0][&quot; Name & quot;].ToString();

textBox3.Text = ds.Tables[0].Rows[0][&quot; Phone & quot;].ToString();

textBox4.Text = ds.Tables[0].Rows[0][&quot; Address & quot;].ToString();

textBox5.Text = ds.Tables[0].Rows[0][&quot; dptid & quot;].ToString();

}

con.Close();

}

private void button4\_Click(object sender, EventArgs e)

{

con.Open();

if (counter & lt; ds.Tables[0].Rows.Count - 1)

{

counter = counter + 1;

textBox1.Text = ds.Tables[0].Rows[counter][&quot; ID & quot;].ToString();

textBox2.Text = ds.Tables[0].Rows[counter][&quot; Name & quot;].ToString();

textBox3.Text = ds.Tables[0].Rows[counter][&quot; Phone & quot;].ToString();

textBox4.Text = ds.Tables[0].Rows[counter][&quot; Address & quot;].ToString();

textBox5.Text = ds.Tables[0].Rows[counter][&quot; dptid & quot;].ToString();

}

else

{

MessageBox.Show(&quot; You are on the last record&quot;);

}

con.Close();

}

private void button5\_Click(object sender, EventArgs e)

{

Form2 f2 = new Form2();

this.Hide();

f2.Show();

}

private void button6\_Click(object sender, EventArgs e)

{

Form3 f3 = new Form3();

this.Hide();

f3.Show();

}

private void button7\_Click(object sender, EventArgs e)

{

Form4 f4 = new Form4();

this.Hide();

f4.Show();

}

private void button11\_Click(object sender, EventArgs e)

{

try

{

con.Open();

OleDbCommand command = new OleDbCommand();

command.Connection = con;

command.CommandText = &quot; Delete from student where ID = &quot; +textBox1.Text + &quot; &quot; ;

command.ExecuteNonQuery();

MessageBox.Show(&quot; Your desired record has been deleted&quot;);

}

catch (Exception ex)

{

MessageBox.Show(&quot; Error & quot; +ex);

}

con.Close();

}

private void button12\_Click(object sender, EventArgs e)

{

try

{

con.Open();

OleDbCommand command = new OleDbCommand();

command.Connection = con;

command.CommandText = &quot; update student set Address = &#39;Lahore&#39; where ID=&quot; + textBox1.Text

+&quot; &quot; ;

command.ExecuteNonQuery();

MessageBox.Show(&quot; Your record has been updated & quot;);

con.Close();

}

catch (Exception ex)

{

MessageBox.Show(&quot; Error & quot; +ex);

}

}

private void button10\_Click(object sender, EventArgs e)

{

con.Open();

OleDbCommand com = new OleDbCommand();

com.Connection = con;

com.CommandText = &quot; insert into student(ID, Name, Address, dptid)

values(&quot; +textBox1.Text + &quot;,&#39;&quot;+textBox2.Text+&quot;&#39;,&quot;+textBox3.Text+&quot;,&#39;&quot;+textBox3.Text+&quot;&#39;,&#39;&quot;+textBox4.Text+&quot;&#39;,

&quot; +textBox5.Text + &quot;)&quot; ;

com.ExecuteNonQuery();

MessageBox.Show(&quot; New record has been added & quot;);

con.Close();

}

private void textBox1\_TextChanged(object sender, EventArgs e)

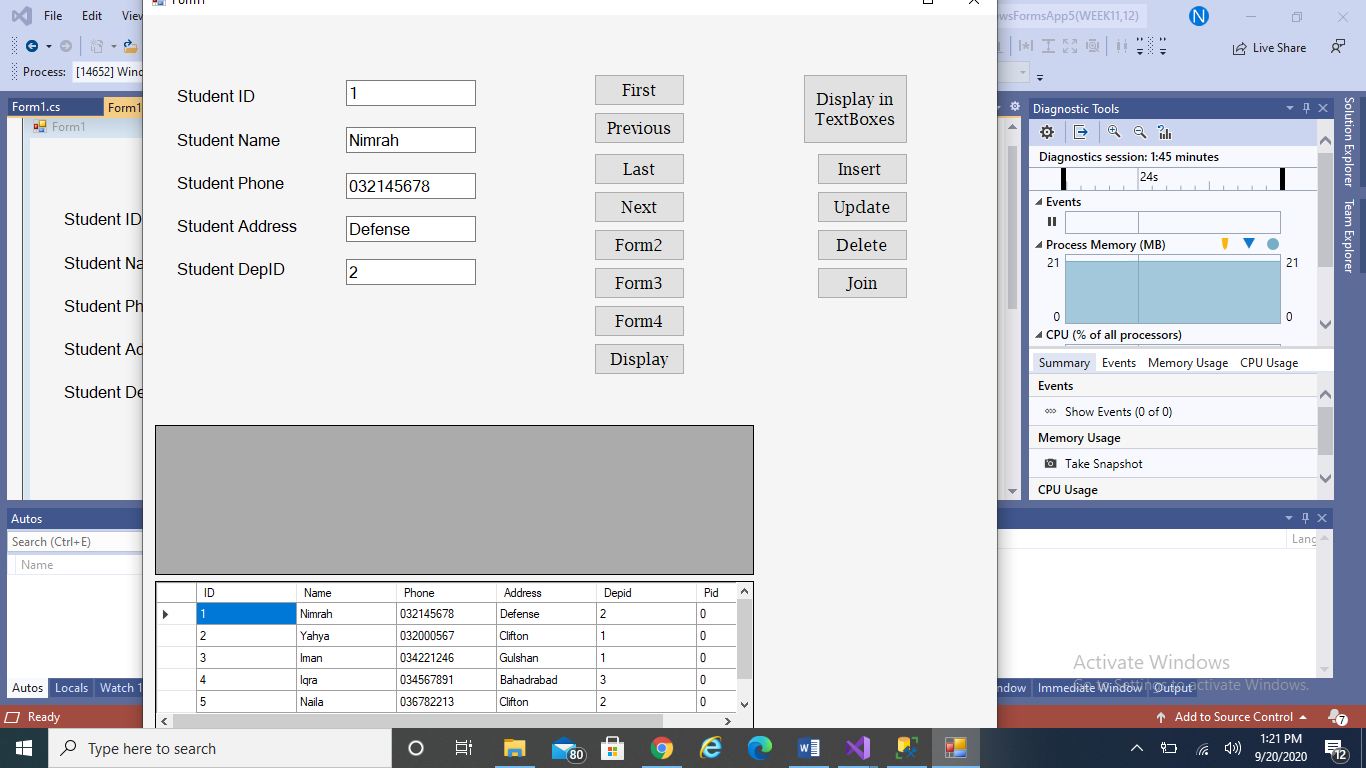
{

}

}

}

OUTPUT:



FORM2:

PROGRAM:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.OleDb;

namespace Working\_with\_Dbs\_in\_CSharp

{

public partial class Form2 : Form

{

public Form2()

{

InitializeComponent();

}

OleDbConnection con = new OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;Data Source=C:/Users/Hp/Desktop/DB/db for c.mdb");

OleDbDataAdapter adap = new OleDbDataAdapter("select \* from Student", "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=C:/Users/Hp/Desktop/DB/db for c.mdb");

DataSet ds = new DataSet("student");

private void Form2\_Load(object sender, EventArgs e)

{

con.Open();

OleDbCommand com = new OleDbCommand("select Name from Student", con);

OleDbDataReader read = com.ExecuteReader();

while (read.Read())

{

comboBox1.Items.Add(read["Name"].ToString());

}

}

private void button1\_Click(object sender, EventArgs e)

{

OleDbDataAdapter adapt = new OleDbDataAdapter("select \* from Student where Name='" + comboBox1.Text + "'", con);

DataSet d2 = new DataSet();

adapt.Fill(d2, "Student");

dataGrid1.DataSource = d2;

}

private void button2\_Click(object sender, EventArgs e)

{

try

{

con.Open();

OleDbCommand command = new OleDbCommand();

command.Connection = con;

command.CommandText = "Delete from student where ID=" + comboBox1.Text + "";

command.ExecuteNonQuery();

MessageBox.Show("Your desired record has been deleted");

}

catch (Exception ex)

{

MessageBox.Show("Error " + ex);

}

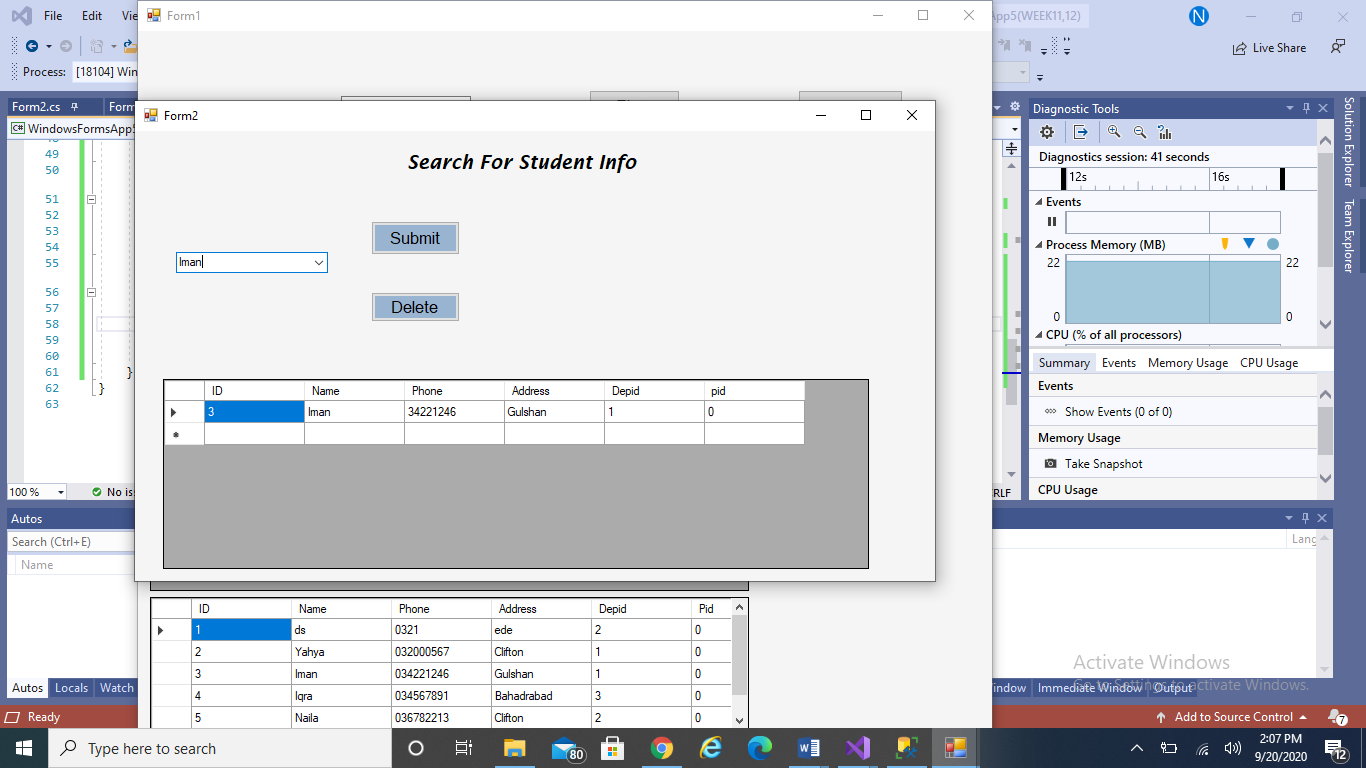
con.Close();

}

}

}

OUTPUT:



FORM3

PROGRAM:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.OleDb;

namespace Working\_with\_Dbs\_in\_CSharp

{

public partial class Form4 : Form

{

public Form4()

{

InitializeComponent();

}

OleDbConnection con = new OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;Data Source=C:/Users/Hp/Desktop/DB/db for c.mdb");

OleDbDataAdapter adap = new OleDbDataAdapter("Select \* from emp", "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=C:/Users/Hp/Desktop/DB/db for c.mdb");

DataSet ds = new DataSet();

private void Form4\_Load(object sender, EventArgs e)

{

con.Open();

adap.Fill(ds);

OleDbCommand cmd = new OleDbCommand("select \* from emp", con);

OleDbDataReader reader = cmd.ExecuteReader();

while (reader.Read())

{

comboBox1.Items.Add(reader["empname"].ToString());

}

con.Close();

}

private void comboBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

con.Open();

OleDbDataAdapter adapter = new OleDbDataAdapter();

OleDbCommand comm = new OleDbCommand("select courses.cname,emp.empname from emp inner join courses on emp.empno = courses.empno where empname='" + comboBox1.Text + "'", con);

adapter.SelectCommand = comm;

DataSet ds1 = new DataSet();

adapter.Fill(ds1, "employee");

dataGrid1.DataSource = ds1.Tables["employee"];

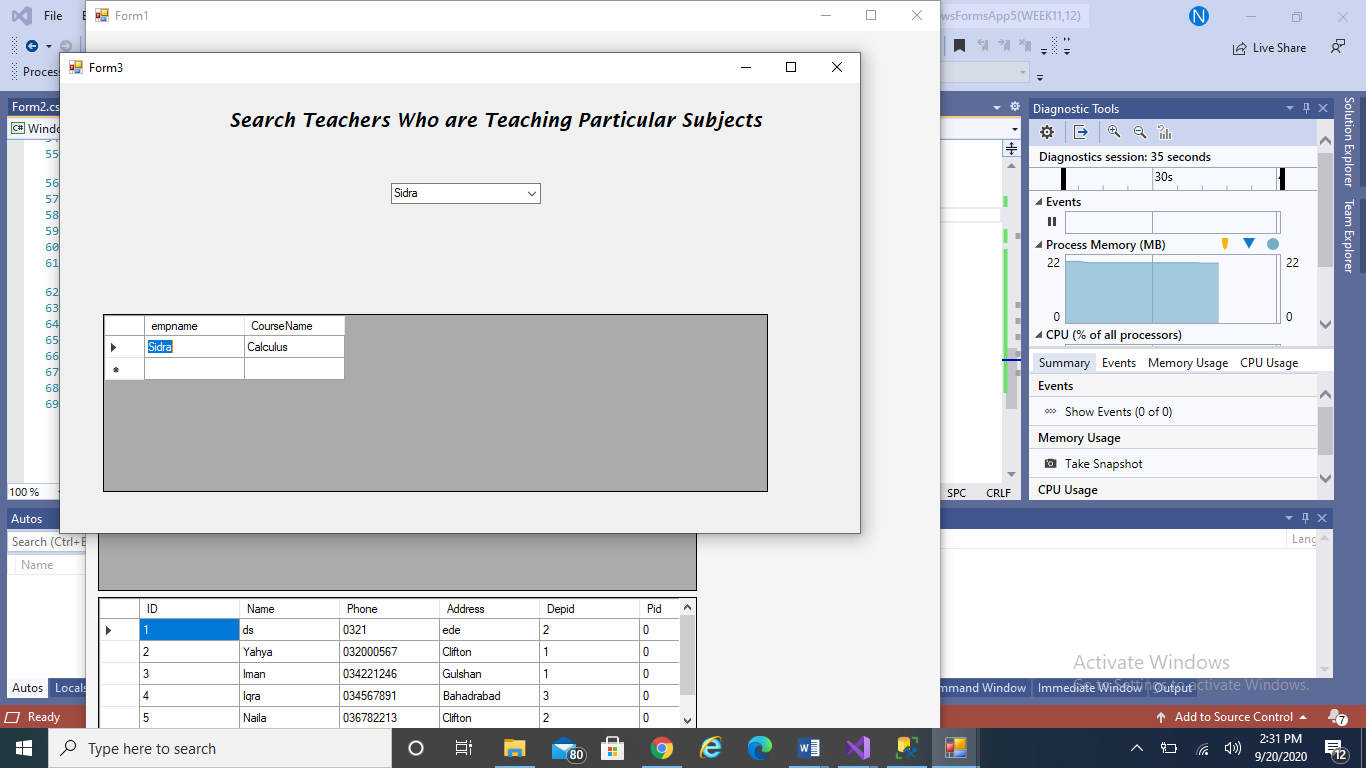
con.Close();

}

}

}

OUTPUT:



**LAB TASK 10**

PROGRAM 5

namespace\_Labtask\_11

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

int counter = 0;

OleDbConnection con = new OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;Data Source=C:/Users/USER/Documents/Database2.mdb");

OleDbDataAdapter adap = new OleDbDataAdapter("select\* from bank", "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=C:/Users/USER/Documents/Database2.mdb");

DataSet d1 = new DataSet("bank");

private void Form1\_Load(object sender, EventArgs e)

{

// TODO: This line of code loads data into the 'database2DataSet4.bank' table. You can move, or remove it, as needed.

this.bankTableAdapter.Fill(this.database2DataSet4.bank);

con.Open();

adap.Fill(d1, "bank");

}

private void button1\_Click(object sender, EventArgs e)

{

//next

if (counter < d1.Tables["bank"].Rows.Count - 1)

{

counter = counter + 1;

textBox1.Text = d1.Tables["bank"].Rows[counter]["AccountNo"].ToString();

textBox2.Text = d1.Tables["bank"].Rows[counter]["Name"].ToString();

textBox3.Text = d1.Tables["bank"].Rows[counter]["Balance"].ToString();

comboBox1.Text = d1.Tables["bank"].Rows[counter]["Branch"].ToString();

}

else if (counter <= d1.Tables["bank"].Rows.Count - 1)

{

MessageBox.Show("You are already on last record");

}

}

private void button2\_Click(object sender, EventArgs e)

{

//previous

if (counter > 0)

{

counter = counter - 1;

textBox1.Text = d1.Tables["bank"].Rows[counter]["AccountNo"].ToString();

textBox2.Text = d1.Tables["bank"].Rows[counter]["Name"].ToString();

textBox3.Text = d1.Tables["bank"].Rows[counter]["Balance"].ToString();

comboBox1.Text = d1.Tables["bank"].Rows[counter]["Branch"].ToString();

string c = comboBox1.Items[comboBox1.SelectedIndex].ToString();

comboBox1.Items.Add(c);

}

else

{

MessageBox.Show("You are already on the first record");

}

}

private void button6\_Click(object sender, EventArgs e)

{

Form2 f2 = new Form2();

f2.Show();

}

private void button3\_Click(object sender, EventArgs e)

{

OleDbCommand com1 = new OleDbCommand("Insert into bank(AccountNo,Name, Branch, Balance) values('" + textBox1.Text + "','" + textBox2.Text + "','" + comboBox1.Text + "','" + textBox3.Text + "')", con);

com1.ExecuteNonQuery();

MessageBox.Show(" One record has been added");

}

private void button7\_Click(object sender, EventArgs e)

{

textBox1.Text = "";

textBox3.Text = "";

comboBox1.Text = "";

}

private void button4\_Click(object sender, EventArgs e)

{

OleDbCommand com = new OleDbCommand("Update bank set Balance='" + textBox3.Text + "' where AccountNo=@AccountNo", con);

com.Parameters.Add("AccountNo", OleDbType.Integer).Value = textBox1.Text;

com.ExecuteNonQuery();

MessageBox.Show(" One record has been updated");

}

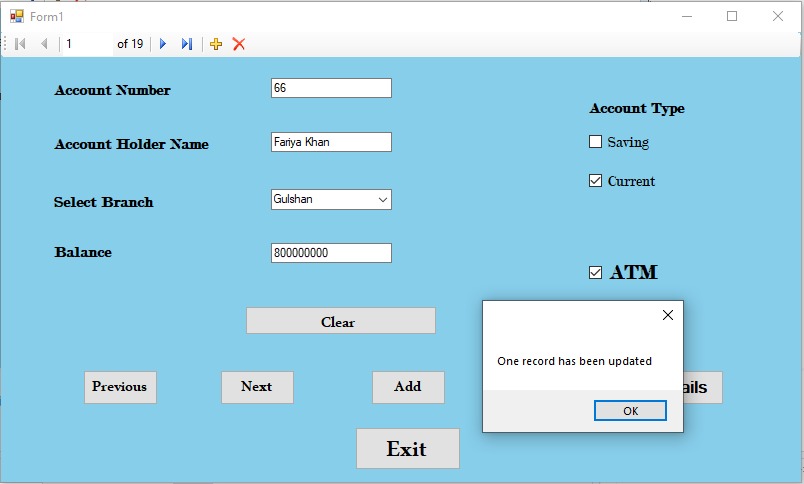
private void button5\_Click(object sender, EventArgs e)

{

con.Close();

}

}



FORM 2

**PROGRAM:**

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace\_Labtask\_11

{

public partial class Form2 : Form

{

public Form2()

{

InitializeComponent();

}

private void Form2\_Load(object sender, EventArgs e)

{

this.bankTableAdapter.Fill(this.database2DataSet4.bank);

}

}

OUTPUT:

