Eman Tahir

2023-BS-AI-015

Section-A

**Object Oriented Programming**

**Assignment-01**

Q1: Write a user-defined program to declare a class which stores a complex number. Demonstrate the use of constant objects, constant member function and constant arguments, using this class.

Solution:

Input: //file: file eman.cpp

//date: 1-05-2024

//name: eman tahir

//reg no. 2023-bs-ai-015

//question statement: Write a user-defined program to declare a class which stores a complex number. Demonstrate the use of constant objects, constant member function and constant arguments, using this class.

#include<iostream>

using namespace std;

class eman

{

private:

double real;

double imag;

public:

eman ()

{

real=(5.0);

imag=(7.0);

}

int complex()

const

{

cout<<real<<"+"<<imag<<"i"<<endl;

}

};

int main ()

{

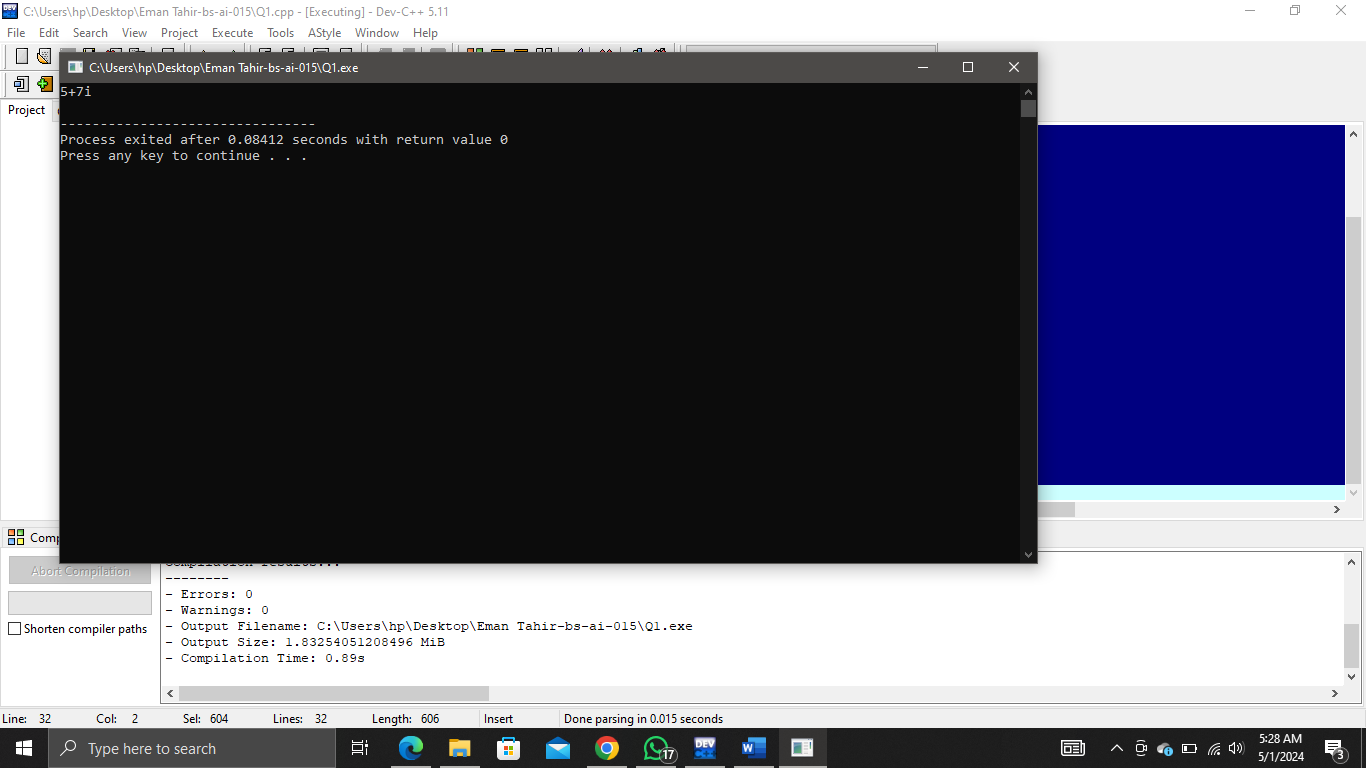
eman n;

n.complex();

return 0;

}

Output:



Q2: Write a class that contain the following attribute

* The name of car
* Direction of car (E, W, N, S)
* The position of car (from imaginary zero point)

The class has fallowing member function

The constructor to be initialize

* Turn function use to change the direction of car to one steps right side (e.g. if the direction is E,

should be change to S and so on)

* Overload the turn function to change the direction to any side directly. It should accept the

direction parameter.

* Move function to change the position of car away from zero point. It should accept the distance

as parameter.

Solution:

Input: //file: file eman.cpp

//date: 1-05-2024

//name: eman tahir

//reg no. 2023-bs-ai-015

//question statement:Write a class that contain the following attribute

//" The name of car

//" Direction of car (E, W, N, S)

//" The position of car (from imaginary zero point)

//The class has fallowing member function

//The constructor to be initialize

//" Turn function use to change the direction of car to one steps right side (e.g. if the direction is E, should be change to S and so on)

//" Overload the turn function to change the direction to any side directly. It should accept the direction parameter.

//" Move function to change the position of car away from zero point. It should accept the distance as parameter.

#include <iostream>

using namespace std;

class Car {

private:

string name;

char direction;

int positionx;

int positiony;

public:

Car(const string & name, char direction, int positionx, int positiony) : name(name), direction(direction), positionx(positionx), positiony(positiony) {}

int turn() {

switch(direction) {

case 'N':

direction = 'E';

break;

case 'E':

direction = 'S';

break;

case 'S':

direction = 'W';

break;

case 'W':

direction = 'N';

break;

default:

break;

}

}

Int turn(char newDirection) {

if (newDirection == 'N' || newDirection == 'E' || newDirection == 'S' || newDirection == 'W')

{

direction = newDirection;

}

}

int move(int distance) {

switch(direction) {

case 'N':

positiony += distance;

break;

case 'E':

positionx += distance;

break;

case 'S':

positiony -= distance;

break;

case 'W':

positionx -= distance;

break;

default:

break;

}

}

int display() {

cout << "Car: " << name <<endl;

cout<<" Position:" << positionx << ", " << positiony << endl;

cout<< " Direction: " << direction <<endl;

}

};

int main() {

Car eman("Honda", 'S', 0, 3);

eman.display();

eman.turn();

eman.display();

eman.turn('W');

eman.display();

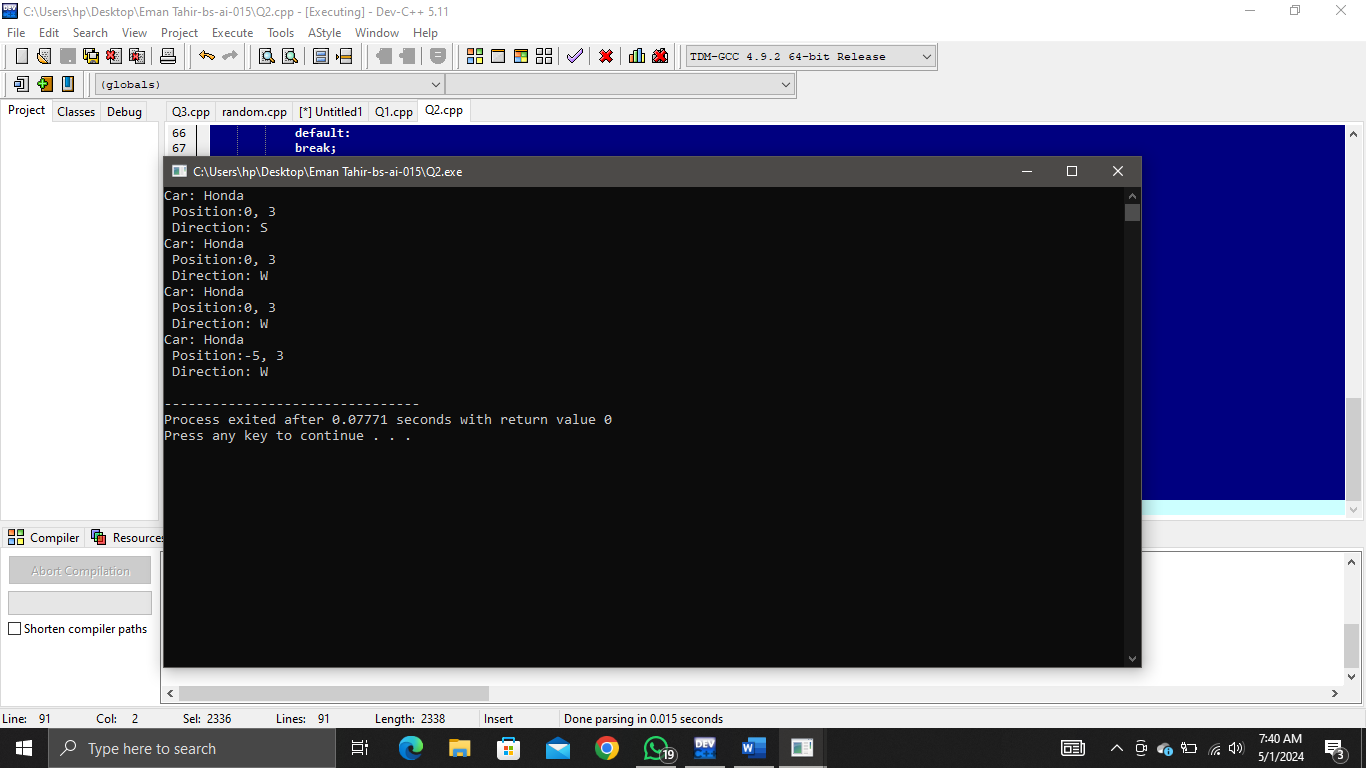
eman.move(5);

eman.display();

return 0;

}

Output:



Q3: Write a function find(…) that accepts a one-dimensional integer array of size 10 as an argument to the function. Your program then finds the location and value of the largest and second-largest elements in a one-dimensional array. Display answers in main().

Solution:

Input: //file: file eman.cpp

//date: 1-05-2024

//name: eman tahir

//reg no. 2023-bs-ai-015

//question statement:Write a function find(…) that accepts a one-dimensional integer array of size 10 as an argument to thefunction. Your program then finds the location and value of the largest and second-largest elements in aone-dimensional array. Display answers in main().

#include <iostream>

using namespace std;

int find(int eman[]) {

int m1 = eman[0];

int m2 = eman[0];

int index1 = 0;

int index2 = 0;

for (int i = 1; i < 10; i++) {

if (eman[i] > m1) {

m2 = m1;

m1 = eman[i];

index2 = index1;

index1 = i;

} else if (eman[i] > m2 && eman[i] != m1) {

m2 = eman[i];

index2 = i;

}

}

cout << "Largest element: " << m1 << " index:" << index1 <<endl;

cout << "Second largest element: " << m2 << " index: " << index2 <<endl;

}

int main() {

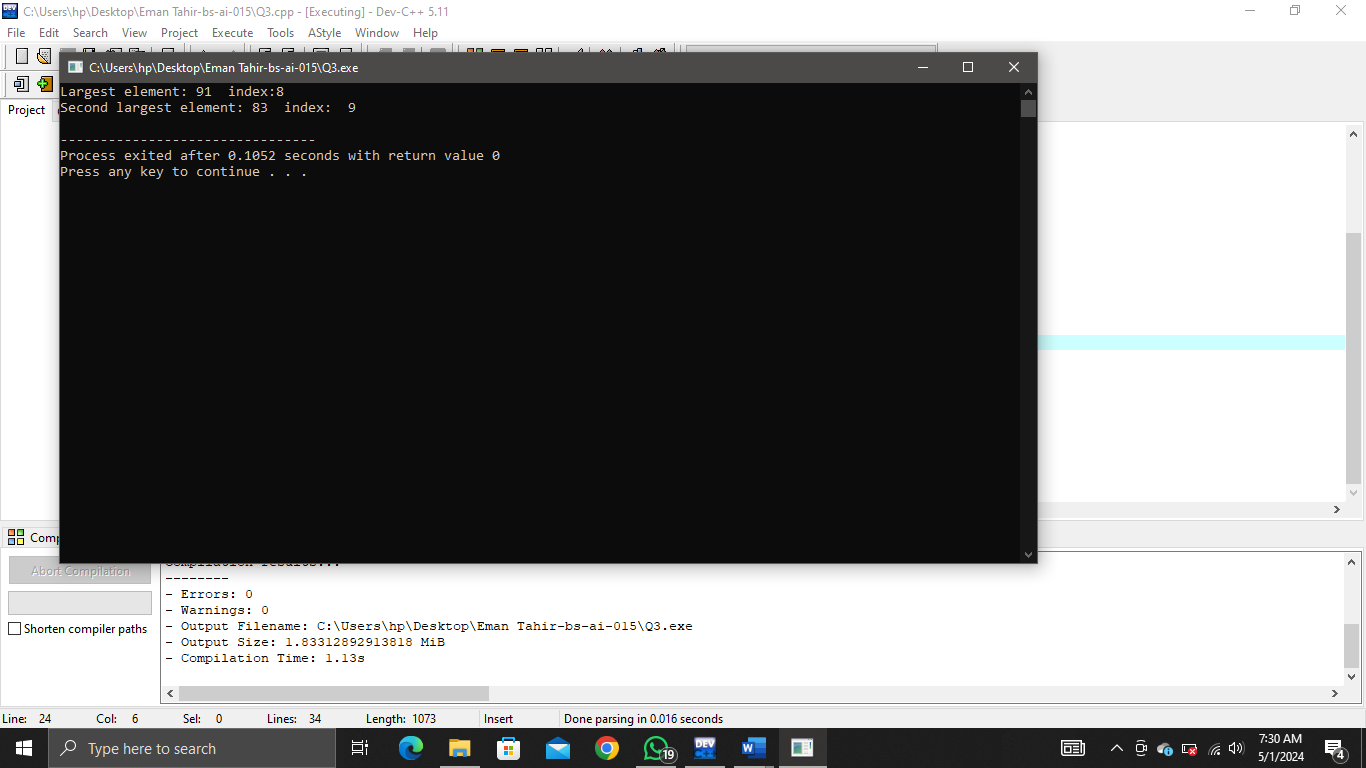
int eman[10] = {19,32,45,67,3,7,20,58,91,83};

find(eman);

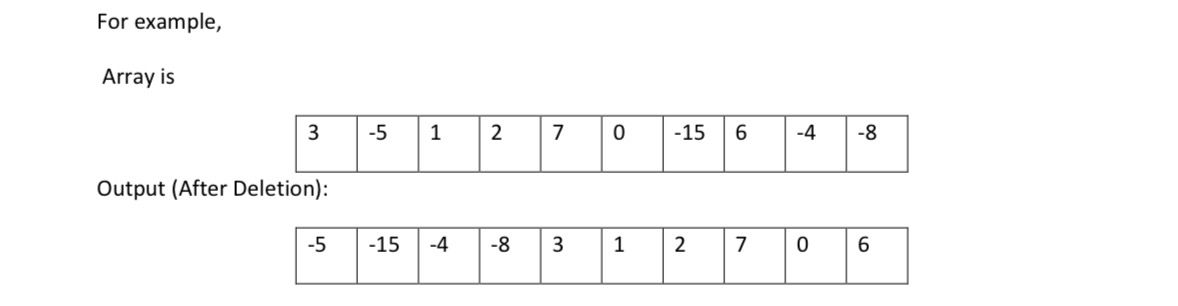
return 0;

}

Output:



Q4: Write a function arrange(…) that accepts a one-dimensional integer array of size 10 as an argument to the function. The program then shifts negative numbers to the left and positive numbers to the right side of the array.



Solution:

Input: //file: file eman.cpp

//date: 1-05-2024

//name: eman tahir

//reg no. 2023-bs-ai-015

//question statement:Write a function arrange(…) that accepts a one-dimensional integer array of size 10 as an argument to the function. The program then shifts negative numbers to the left and positive numbers to the right side of the array.

#include <iostream>

using namespace std;

int arrange(int eman[], int size)

{

int left = 0, right = size - 1;

while (left <= right)

{

if (eman[left] < 0 && eman[right] >= 0)

{

int temp = eman[left];

eman[left] = eman[right];

eman[right] = temp;

left++;

right--;

}

else

{

if (eman[left] >= 0)

{

left++;

}

if (eman [right] < 0)

{

right--;

}

}

}

}

int main() {

int eman[10] = {3, -5, 1, 2, 7, 0, -15, 6, -4, -8};

int size = 10;

arrange(eman, size);

cout << "Output: ";

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

cout << eman[i] << " ";

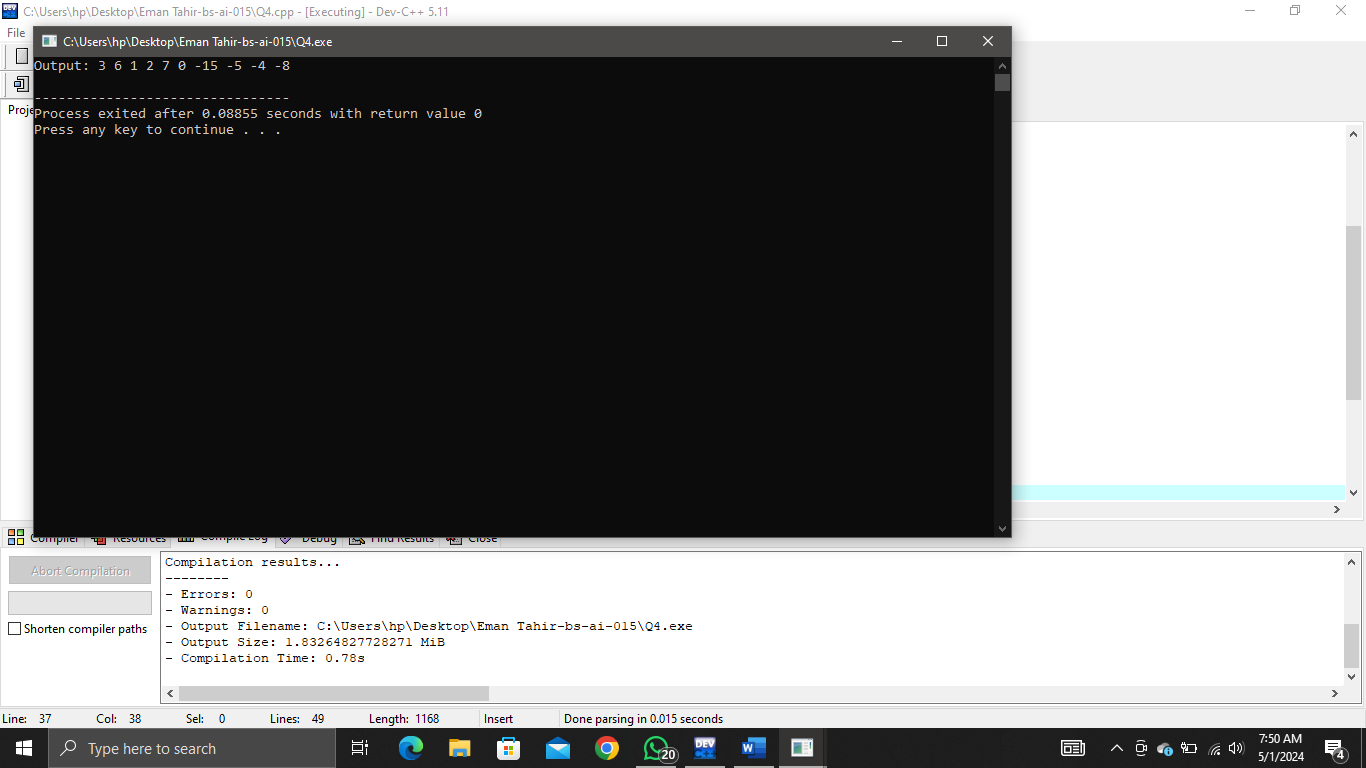
}

cout << endl;

return 0;

}

Output:



Q5: Create a class employee which stores is name, ID and salary of an employee by user input. The ID should be generated upon the creation of object, starting from 1. Include all the constructors and destructor in the class. Create one object using each of the constructors and display it.

Solution:

Input: //file: file eman.cpp

//date: 1-05-2024

//name: eman tahir

//reg no. 2023-bs-ai-015

//question statement: Create a class employee which stores is name, ID and salary of an employee by user input. The ID should be generated upon the creation of object, starting from 1. Include all the constructors and destructor in the class. Create one object using each of the constructors and display it.

#include <iostream>

using namespace std;

class Employee

{

private:

static int nextId;

int id;

string name;

double salary;

public:

Employee() {

id = 1;

name = "a";

salary = 20000.0;

}

Employee(const string& name, double salary) {

id = 2;

this->name = name;

this->salary = salary;

}

int display() {

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

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

cout << "Salary: " << salary << endl;

}

};

int main() {

Employee eman1;

eman1.display();

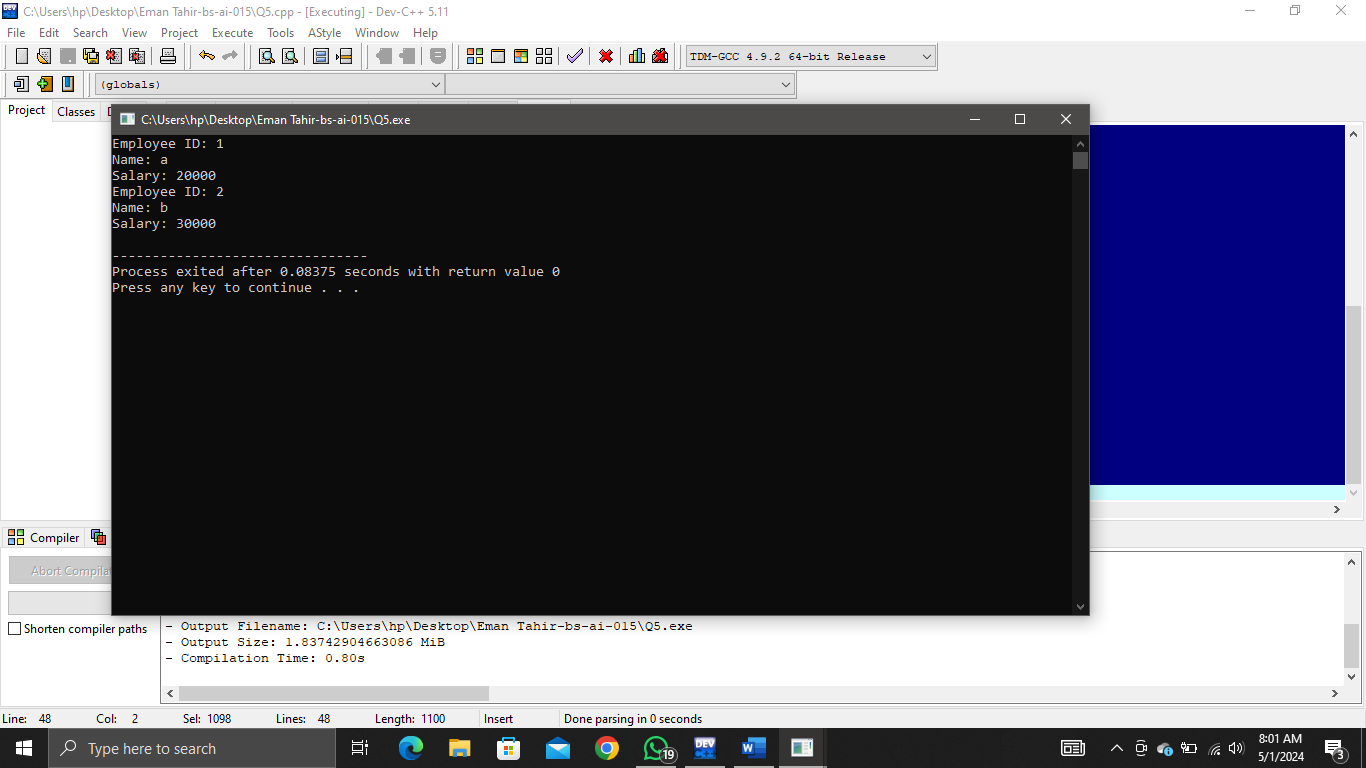
Employee eman2("b", 30000.0);

eman2.display();

return 0;

}

Output:



Q6: Write a C++ program for the class vehicle and its drive class water transport, road transport and air transport vehicles. Make suitable data variables and member functions. When you create an object must be count and display total no of object created also create every class objects and access member through the member functions.

Solution:

Input: //file: file eman.cpp

//date: 1-05-2024

//name: eman tahir

//reg no. 2023-bs-ai-015

//question statement:Write a C++ program for the class vehicle and its drive class water transport, road transport and air transport vehicles. Make suitable data variables and member functions. When you create an object must be count and display total no of object created also create every class objects and access member through the member functions.

#include <iostream>

using namespace std;

class Vehicle {

private:

static int count;

public:

Vehicle() {

count++;

}

virtual int display()

{

cout << "Vehicle" << endl;

}

static int Count() {

return count;

}

};

int Vehicle::count = 0;

class WaterTransport : public Vehicle {

public:

int display() {

cout << "Water Transport" << endl;

}

};

class RoadTransport : public Vehicle {

public:

int display() {

cout << "Road Transport" <<endl;

}

};

class AirTransport : public Vehicle {

public:

int display() {

cout << "Air Transport" <<endl;

}

};

int main() {

WaterTransport eman1;

RoadTransport eman2;

AirTransport eman3;

cout << "Total number of objects created: " << Vehicle::Count() << endl;

eman1.display();

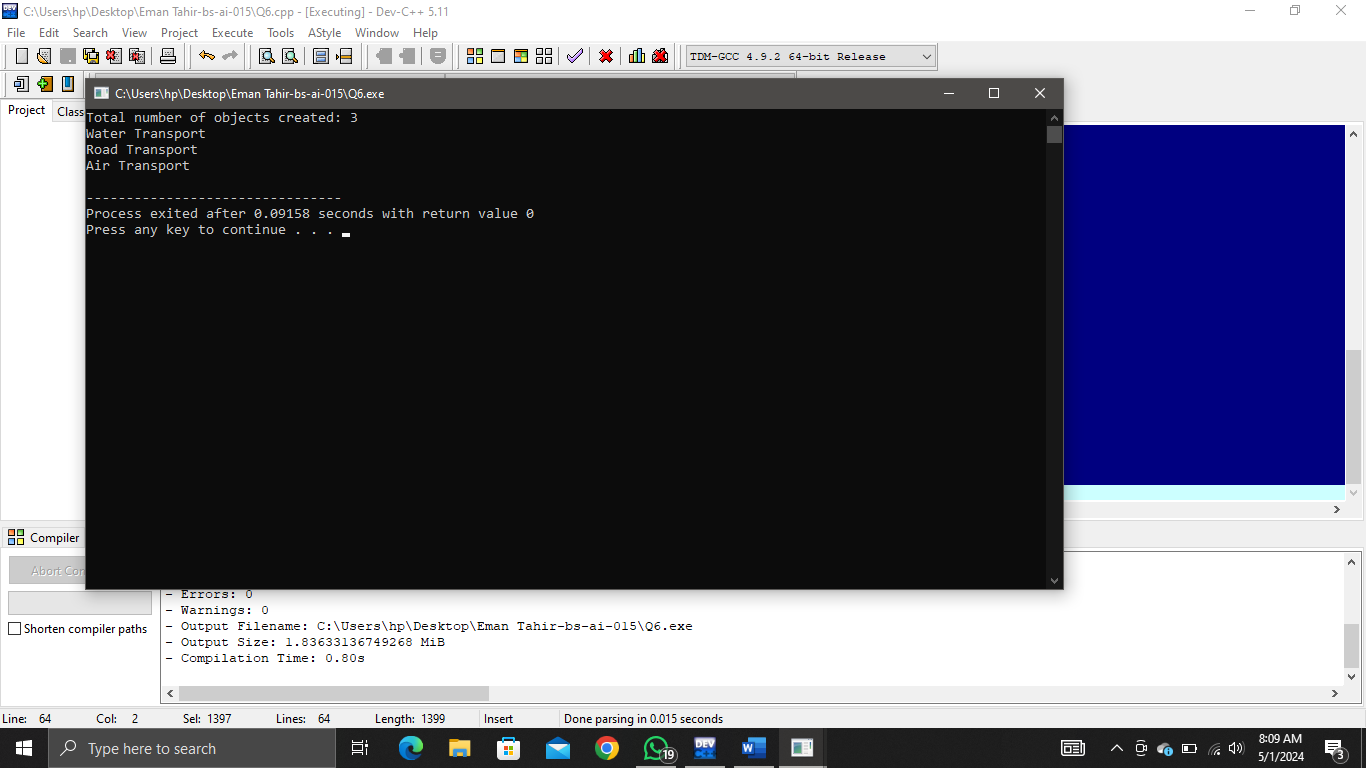
eman2.display();

eman3.display();

return 0;

}

Output:



Q7: Implement a C++ class named Employee with the following specifications:

* The class should have private data members name (string), id (integer), and salary (floating-

point).

* Implement a static data member totalEmployees to keep track of the total number of

employees.

* Implement a static member function averageSalary() that calculates and returns the average

salary of all employees.

* Provide member functions to set and get the values of name, id, and salary.
* Implement a constructor to initialize the name, id, and salary of an employee.
* Implement a destructor to decrement the totalEmployees count when an object is destroyed.

Solution:

Input: //file: file eman.cpp

//date: 1-05-2024

//name: eman tahir

//reg no. 2023-bs-ai-015

//question statement:Implement a C++ class named Employee with the following specifications:

//" The class should have private data members name (string), id (integer), and salary (floating-point).

//" Implement a static data member totalEmployees to keep track of the total number of employees.

//" Implement a static member function averageSalary() that calculates and returns the average salary of all employees.

//" Provide member functions to set and get the values of name, id, and salary.

//" Implement a constructor to initialize the name, id, and salary of an employee.

//" Implement a destructor to decrement the totalEmployees count when an object is destroyed

#include <iostream>

using namespace std;

class eman {

private:

string name;

int id;

double salary;

public:

static int totalEmployees;

static double totalSalary;

eman(string name, int id, double salary) : name(name), id(id), salary(salary) {

totalEmployees++;

totalSalary += salary;

}

static double averageSalary() {

return totalSalary / totalEmployees;

}

int Name(string name) {

this->name = name;

}

string Name() const {

return name;

}

int setId(int id) {

this->id = id;

}

int getId() const {

return id;

}

int setSalary(double salary) {

this->salary = salary;

totalSalary = totalSalary - salary + this->salary;

}

double Salary() const {

return salary;

}

};

int eman::totalEmployees = 0;

double eman::totalSalary = 0.0;

int main() {

eman e1("a", 1, 5000.0);

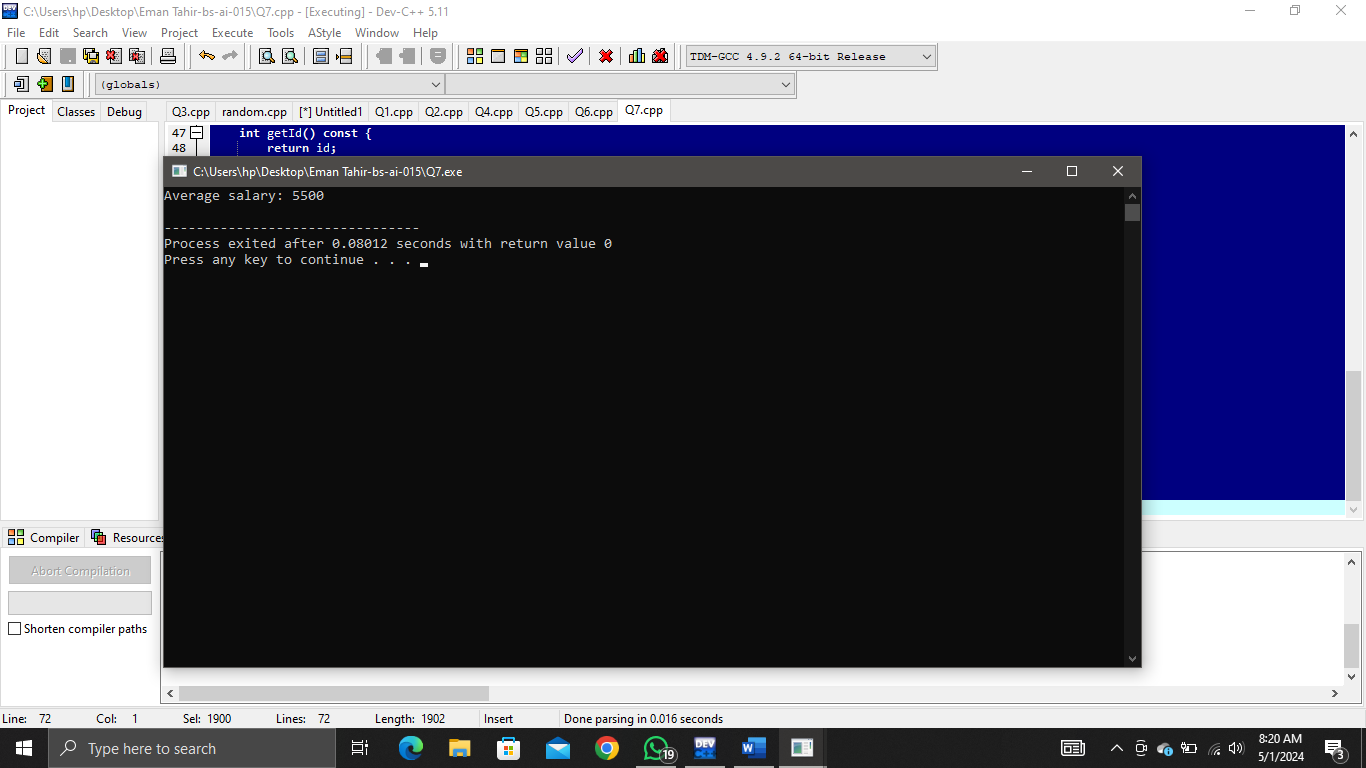
eman e2("b", 2, 6000.0);

cout << "Average salary: " << eman::averageSalary() << endl;

return 0;

}

Output:



Q8: (Car Pool Savings Calculator) Research several car-pooling websites. create an application that calculates your daily driving cost, so that you can estimate how much money could be saved by carpooling, which also has other advantages such as reducing carbon emission and reducing traffic congestion. The application should input the following and display the user's cost per day of driving to word:

a) Total miles driven per day.

b) Cost per gallon of gasoline.

c) Average miles per gallon

d) Parking fees per day.

e) Toll per day.

Solution:

Input: //file: file eman.cpp

//date: 1-05-2024

//name: eman tahir

//reg no. 2023-bs-ai-015

//question statement:(Car Pool Savings Calculator) Research several car-pooling websites. create an application that calculates your daily driving cost, so that you can estimate how much money could be saved by carpooling, which also has other advantages such as reducing carbon emission and reducing traffic congestion. The application should input the following and display the user's cost per day of driving to word:

//a) Total miles driven per day.

//b) Cost per gallon of gasoline.

//c) Average miles per gallon

//d) Parking fees per day.

//e) Toll per day.

#include <iostream>

using namespace std;

int main() {

double tmdpd;

double cpgog;

double ampg;

double pfpd;

double tpd;

cout << "Enter total miles driven per day: ";

cin >> tmdpd;

cout << "Enter cost per gallon of gasoline: ";

cin >> cpgog;

cout << "Enter average miles per gallon: ";

cin >> ampg;

cout << "Enter parking fees per day: ";

cin >> pfpd;

cout << "Enter toll per day: ";

cin >>tpd;

double tcpd = (tmdpd / ampg) \* cpgog + pfpd + tpd;

cout << "Your daily driving cost is: $" << tcpd << endl;

return 0;

}

Output:

