

Assignment No: 03

Submitted to : Sir Javed

Submitted by: Ayesha Imran

Registration No : BS-AI-O57

Department : BS-AI-(A)

// File: problem 1 assignment no 3 oop.cpp

// Date: 01-05-2024

// Name: Ayesha Imran

// Registration No: 2023-BS-AI-057

/\*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. \*/

**Code:**

#include <iostream>

using namespace std;

class Complex {

private:

double real;

double imag;

public:

Complex(double r = 0.0, double i = 0.0) : real(r), imag(i) {}

// Constant member function to display the complex number

void display() const {

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

}

// Addition of two complex numbers

Complex add(const Complex& c) const {

return Complex(real + c.real, imag + c.imag);

}

};

int main() {

// Constant object

const Complex c1(3.0, 4.0);

const Complex c2(1.0, 2.0);

// Displaying constant objects

cout << "Constant Object 1: ";

c1.display();

cout << "Constant Object 2: ";

c2.display();

// Adding constant objects and displaying the result

cout << "Addition of constant objects: ";

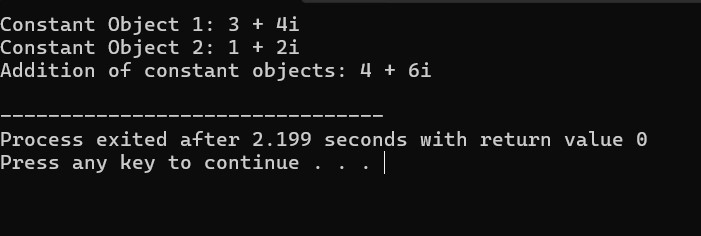
Complex result = c1.add(c2);

result.display();

return 0;

}

**Output:**



// File: problem 2 assignment no 3 oop.cpp

// Date: 01-05-2024

// Name: Ayesha Imran

// Registration No: 2023-BS-AI-057

/\*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\*/

**Code:**

#include <iostream>

#include <string>

using namespace std;

class Car {

private:

string name;

char direction;

int positionX;

int positionY;

public:

// Constructor to initialize the car attributes

Car(const string& carName, char initialDirection, int initialPositionX, int initialPositionY)

: name(carName), direction(initialDirection), positionX(initialPositionX), positionY(initialPositionY) {}

// Function to turn the car one step right

void 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;

}

}

// Overloaded turn function to turn the car to any side directly

void turn(char newDirection) {

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

direction = newDirection;

}

// Function to move the car away from zero point

void 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;

}

}

// Function to display the current state of the car

void display() {

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

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

cout << "Position: (" << positionX << ", " << positionY << ")" << endl;

}

};

int main() {

// Create a car object

Car myCar("MyCar", 'N', 0, 0);

// Display initial state

cout << "Initial State:" << endl;

myCar.display();

cout << std::endl;

// Turn the car and move it

myCar.turn(); // Turn one step right

myCar.move(5); // Move 5 units

cout << "After turning one step right and moving 5 units:" << endl;

myCar.display();

cout << std::endl;

// Turn the car to East directly and move it

myCar.turn('E'); // Turn directly to East

myCar.move(3); // Move 3 units

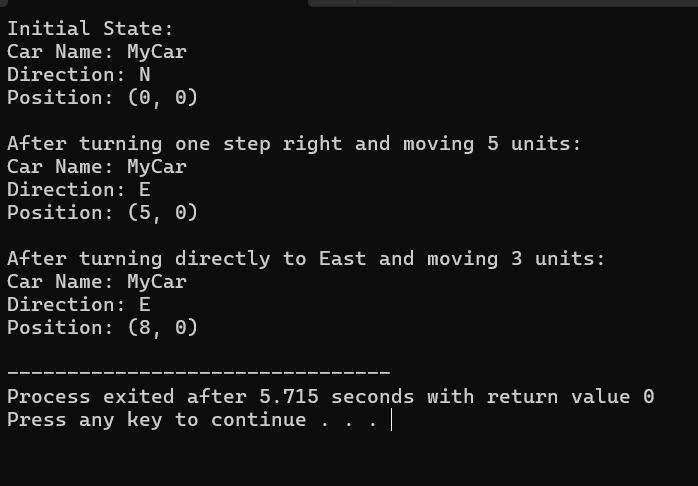
cout << "After turning directly to East and moving 3 units:" << endl;

myCar.display();

return 0;

}

**Output:**



// File: problem 3 assignment no 3 oop.cpp

// Date: 01-05-2024

// Name: Ayesha Imran

// Registration No: 2023-BS-AI-057

/\*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 inmain().\*/

**Code:**

#include <iostream>

using namespace std;

void find(const int arr[], int size, int& maxIndex, int& maxValue, int& secondMaxIndex, int& secondMaxValue) {

maxIndex = 0;

maxValue = arr[0];

secondMaxIndex = -1;

secondMaxValue = INT\_MIN;

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

if (arr[i] > maxValue) {

secondMaxIndex = maxIndex;

secondMaxValue = maxValue;

maxIndex = i;

maxValue = arr[i];

} else if (arr[i] > secondMaxValue) {

secondMaxIndex = i;

secondMaxValue = arr[i];

}

}

}

int main() {

const int size = 10;

int arr[size] = {12, 45, 78, 34, 98, 23, 56, 89, 67, 43};

int maxIndex, maxValue, secondMaxIndex, secondMaxValue;

find(arr, size, maxIndex, maxValue, secondMaxIndex, secondMaxValue);

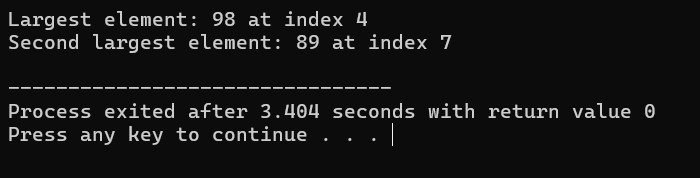
cout << "Largest element: " << maxValue << " at index " << maxIndex << endl;

cout << "Second largest element: " << secondMaxValue << " at index " << secondMaxIndex << endl;

return 0;

}

**Output:**



// File: problem 4 assignment no 3 oop.cpp

// Date: 01-05-2024

// Name: Ayesha Imran

// Registration No: 2023-BS-AI-057

/\*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.

For example,

Array is

3

-5

1

2

7

0

-15

6

-4

-8

Output (After Deletion):

-5

-15

-4

-8

3

1

2

7

0

6\*/

**Code:**

#include <iostream>

using namespace std;

void shiftArray(int arr[], int size) {

int left = 0, right = size - 1;

while (left <= right) {

if (arr[left] < 0) {

left++;

} else if (arr[right] >= 0) {

right--;

} else {

int temp = arr[left];

arr[left] = arr[right];

arr[right] = temp;

left++;

right--;

}

}

}

int main() {

const int size = 10;

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

cout << "Array before shifting:" <<endl;

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

cout << arr[i] <<endl;

}

shiftArray(arr, size);

cout << "\nArray after shifting:" <<endl;

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

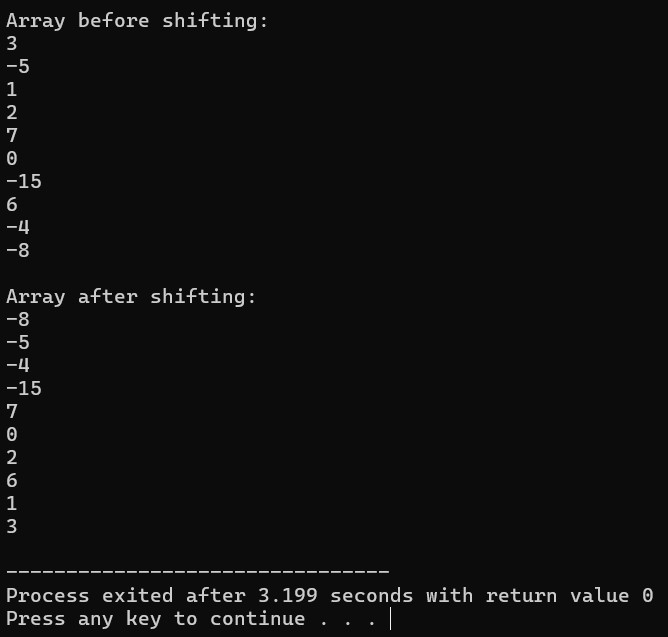
cout << arr[i] <<endl;

}

return 0;

}

**Output:**



// File: problem 5 assignment no 3 oop.cpp

// Date: 01-05-2024

// Name: Ayesha Imran

// Registration No: 2023-BS-AI-057

/\*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.\* /

**Code:**

#include <iostream>

#include <string>

using namespace std;

class Employee {

private:

static int nextID; // Static variable to track the next available ID

int ID;

string name;

double salary;

public:

// Default constructor

Employee() {

ID = ++nextID;

name = "Ayesha";

salary = 3900000;

}

// Parameterized constructor

Employee(const string& empName, double empSalary) {

ID = ++nextID;

name = empName;

salary = empSalary;

}

// Destructor

~Employee() {

cout << "Destructor called for Employee: " << name <<endl;

}

// Copy constructor

Employee(const Employee& other) {

ID = ++nextID;

name = other.name;

salary = other.salary;

}

// Function to display employee details

void display() const {

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

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

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

}

};

// Initializing static member variable

int Employee::nextID = 0;

int main() {

// Creating objects using different constructors

Employee emp1; // Default constructor

Employee emp2("Maham", 50000); // Parameterized constructor

Employee emp3 = emp2; // Copy constructor

// Displaying objects

cout << "Employee 1:" <<endl;

emp1.display();

cout << endl;

cout << "Employee 2:" <<endl;

emp2.display();

cout <<endl;

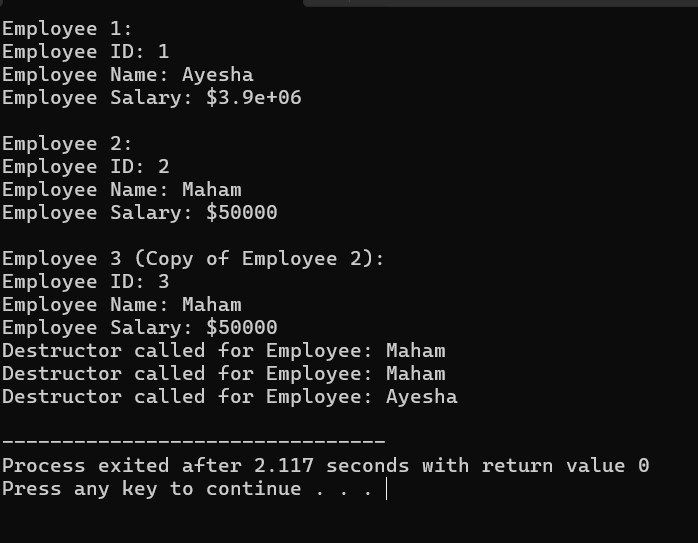
cout << "Employee 3 (Copy of Employee 2):" <<endl;

emp3.display();

return 0;

}

**Output:**



File: problem 6 assignment no 3 oop.cpp

// Date: 01-05-2024

// Name: //Ayesha Imran

// Registration No: 2023-BS-AI-057

/\*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.\*/

**Code:**

#include <iostream>

#include <string>

using namespace std;

class Vehicle {

protected:

static int totalVehicles;

public:

Vehicle() {

totalVehicles++;

}

virtual ~Vehicle() {}

virtual void displayType() const {

cout << "Vehicle" <<endl;

}

static void displayTotal() {

cout << "Total Vehicles: " << totalVehicles << endl;

}

};

int Vehicle::totalVehicles = 0;

class WaterTransport : public Vehicle {

public:

WaterTransport() : Vehicle() {}

void displayType() const override {

cout << "Water Transport" << endl;

}

};

class RoadTransport : public Vehicle {

public:

RoadTransport() : Vehicle() {}

void displayType() const override {

cout << "Road Transport" << endl;

}

};

class AirTransport : public Vehicle {

public:

AirTransport() : Vehicle() {}

void displayType() const override {

cout << "Air Transport" << endl;

}

};

int main() {

WaterTransport boat;

RoadTransport car;

AirTransport plane;

cout << "Types of vehicles created:" << endl;

boat.displayType();

car.displayType();

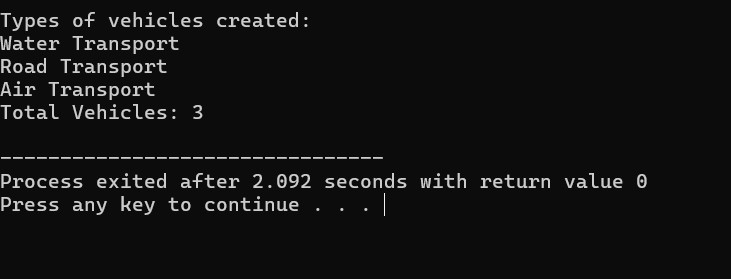
plane.displayType();

Vehicle::displayTotal();

return 0;

}

**Output:**



// File: problem 7 assignment no 3 oop.cpp

// Date: 01-05-2024

// Name: Ayesha Imran

// Registration No: 2023-BS-AI-057

/\*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 total Employees to keep track of the total number of employees.

 Implement a static member function average Salary() 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 employee.

 Implement a destructor to decrement the total Employees count when an object is destroyed.\*/

**Code:**

#include <iostream>

#include <string>

using namespace std;

class Employee {

public:

string name;

int id;

float salary;

static int totalEmployees;

static float totalSalary;

public:

// Constructor

Employee(string name, int id, float salary) : name(name), id(id), salary(salary) {

totalEmployees++;

totalSalary += salary;

}

// Destructor

~Employee() {

totalEmployees--;

totalSalary -= salary;

}

// Setter functions

void setName(string name) {

this->name = name;

}

void setId(int id) {

this->id = id;

}

void setSalary(float salary) {

totalSalary -= this->salary;

this->salary = salary;

totalSalary += this->salary;

}

// Getter functions

string getName() const {

return name;

}

int getId() const {

return id;

}

float getSalary() const {

return salary;

}

// Static member function to calculate average salary

static float averageSalary() {

if (totalEmployees == 0) {

return 0.0f;

}

return totalSalary / totalEmployees;

}

};

// Initialize static members

int Employee::totalEmployees = 0;

float Employee::totalSalary = 0.0f;

int main() {

Employee F1("Ayesha Imran", 1, 98065.0);

Employee F2("Maham Nisar", 2, 89078.0);

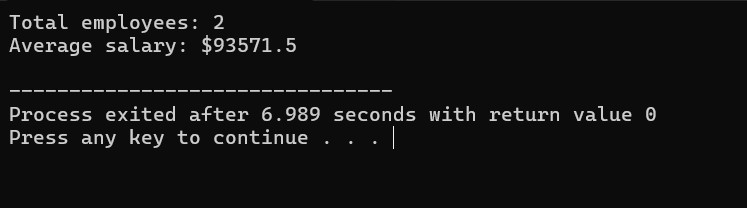
cout << "Total employees: " << Employee::totalEmployees << endl;

cout << "Average salary: $" << Employee::averageSalary() << endl;

return 0;

}

**Output:**



// File: problem 8 assignment no 3 oop.cpp

// Date: 01-05-2024

// Name: Ayesha Imran

// Registration No: 2023-BS-AI-057

/\*(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.\*/

**Code:**

#include <iostream>

#include <cmath>

#include <iomanip>

#include <string>

using namespace std;

// Function to convert number to words

string numberToWords(int num) {

string ones[] = {"", "One", "Two", "Three", "Four", "Five", "Six", "Seven", "Eight", "Nine"};

string tens[] = {"", "Ten", "Twenty", "Thirty", "Forty", "Fifty", "Sixty", "Seventy", "Eighty", "Ninety"};

string teens[] = {"", "Eleven", "Twelve", "Thirteen", "Fourteen", "Fifteen", "Sixteen", "Seventeen", "Eighteen", "Nineteen"};

if (num < 10)

return ones[num];

if (num < 20)

return teens[num - 10];

return tens[num / 10] + " " + ones[num % 10];

}

// Function to calculate and display cost per day in words

void calculateCostPerDay(double milesDriven, double costPerGallon, double avgMilesPerGallon, double parkingFees, double toll) {

double gallonsUsed = milesDriven / avgMilesPerGallon;

double costPerDay = gallonsUsed \* costPerGallon + parkingFees + toll;

cout << fixed << setprecision(2);

cout << "Your daily driving cost is $" << costPerDay << endl;

cout << "In words: ";

// Convert cost to words

int dollars = static\_cast<int>(costPerDay);

int cents = round((costPerDay - dollars) \* 100);

if (dollars == 0 && cents == 0) {

cout << "Zero dollars";

} else {

cout << numberToWords(dollars) << " dollars";

if (cents != 0) {

cout << " and " << numberToWords(cents) << " cents";

}

}

cout << " per day." << endl;

}

int main() {

double milesDriven, costPerGallon, avgMilesPerGallon, parkingFees, toll;

// Input values from the user

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

cin >> milesDriven;

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

cin >> costPerGallon;

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

cin >> avgMilesPerGallon;

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

cin >> parkingFees;

cout << "Enter toll per day: ";

cin >> toll;

// Calculate and display cost per day

calculateCostPerDay(milesDriven, costPerGallon, avgMilesPerGallon, parkingFees, toll);

return 0;

}

**Output:**

