**Assignment**

**Qandeel asim**

**037**

**Bs ai section a**

**Program 1**

#include <iostream>

using namespace std;

void arrange(int arr[]) {

int left = 0, right = 9;

while (left <= right) {

if (arr[left] < 0) {

left++;

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

right--;

} else {

swap(arr[left], arr[right]);

left++;

right--;

}

}

}

int main() {

int arr[10];

cout << "Enter 10 integers separated by spaces: ";

for (int i = 0; i < 10; i++) {

cin >> arr[i];

}

cout << "Original array: ";

for (int i = 0; i < 10; i++) {

cout << arr[i] << " ";

}

cout << endl;

arrange(arr);

cout << "Arranged array: ";

for (int i = 0; i < 10; i++) {

cout << arr[i] << " ";

}

cout << endl;

return 0;

}

**Program 2**

#include <iostream>

#include <string>

using namespace std;

class employee {

public:

employee() {

ID++;

name = "no name";

salary = 0.0;

}

employee(string nam) {

name = nam;

ID++;

salary = 0.0;

}

employee(float sala) {

salary = sala;

ID++;

name = "no name";

}

employee(float sal, string nam) {

salary = sal;

name = nam;

ID++;

}

void set\_name(string nam) {

name = nam;

}

void set\_salary(float sal) {

salary = sal;

}

string get\_name() {

return name;

}

float get\_salary() {

return salary;

}

void display() {

cout << "ID: " << ID << ", Name: " << name << ", Salary: " << salary << endl;

}

~employee() { ID--; }

private:

static int ID;

string name;

float salary;

};

int employee::ID = 0;

int main() {

employee F1, F2("Qandeel "), F3(4000), F4(4500 ” asim");

F1.display();

F2.display();

F3.display();

F4.display();

return 0;

}

**Program 3**

#include <iostream>

#include <string>

using namespace std;

class vehicle {

public:

vehicle() {

total\_objects++;

}

~vehicle() {

total\_objects--;

}

static int get\_total\_objects() {

return total\_objects;

}

void display() {

cout << "Name: " << name << ", Type: Vehicle, Total Objects: " << get\_total\_objects() << endl;

}

static int total\_objects;

string name;

};

int vehicle::total\_objects = 0;

class water\_transport : public vehicle {

public:

water\_transport(string n) {

name = n;

total\_objects++;

}

~water\_transport() { total\_objects--; }

};

class road\_transport : public vehicle {

public:

road\_transport(string n) {

name = n;

total\_objects++;

}

~road\_transport() { total\_objects--; }

};

class air\_transport : public vehicle {

public:

air\_transport(string n) {

name = n;

total\_objects++;

}

~air\_transport() { total\_objects--; }

};

int main() {

water\_transport W1("Boat"), W2("Ship");

W1.display();

W2.display();

road\_transport R1("Car"), R2("Truck");

R1.display();

R2.display();

air\_transport A1("Plane"), A2("Helicopter");

A1.display();

A2.display();

cout << "Total Vehicle Objects: " << vehicle::get\_total\_objects() << endl;

return 0;

}

**Program 4**

#include <iostream>

using namespace std;

double calculateDrivingCost(double milesPerDay, double costPerGallon, double milesPerGallon, double parkingFee, double toll, int numPeople) {

double gasCost = (milesPerDay / milesPerGallon) \* costPerGallon;

double totalCost = gasCost + parkingFee + toll;

double costPerPerson = totalCost / numPeople;

return costPerPerson;

}

int main() {

double milesPerDay, costPerGallon, milesPerGallon, parkingFee, toll;

int numPeople;

cout << "\t\tCar Pool Savings Calculator\n\n";

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

cin >> milesPerDay;

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

cin >> costPerGallon;

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

cin >> milesPerGallon;

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

cin >> parkingFee;

cout << "Enter toll per day: ";

cin >> toll;

cout << "Enter number of people in the carpool (including yourself): ";

cin >> numPeople;

double dailyCostPerPerson = calculateDrivingCost(milesPerDay, costPerGallon, milesPerGallon, parkingFee, toll, numPeople);

cout << "\nYour daily driving cost per person: $" << dailyCostPerPerson << "\n";

double totalSavings = (milesPerDay / milesPerGallon) \* costPerGallon - dailyCostPerPerson;

if (totalSavings > 0) {

cout << "Money saved by carpooling per day: $" << totalSavings << "\n";

} else {

cout << "Carpooling does not lead to savings compared to driving alone.\n";

}

return 0;

}

**Program 5**

#include<iostream>

using namespace std;

class complex{

int r,i;

public:

void setval(){

cout<<"Enter Real number: ";

cin>>r;

cout<<"Enter Imaginary number: ";

cin>>i;

}

void display(){

cout<<"Number is "<<r<<" + "<<i<<"i";

}

};

int main(){

complex obj;

obj.setval();

obj.display();

}

**Program 6**

#include<iostream>

using namespace std;

class Car{

string name;

char direction;

int position;

public:

// Constructor to initialize the car

Car(string n, char d, int p) {

name = n;

direction = d;

position = p;

}

// Turn function to change the direction to 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;

}

}

// Overloaded turn function to change the direction directly

void turn(char newDirection) {

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

direction = newDirection;

} else {

cout << "Invalid direction!" << endl;

}

}

// Move function to change the position of the car

void move(int distance) {

switch (direction) {

case 'N':

position += distance;

break;

case 'E':

position += distance;

break;

case 'S':

position -= distance;

break;

case 'W':

position -= distance;

break;

}

}

// Function to display the car's information

void display() {

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

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

cout << "Position: " << position << endl;

}

};

int main(){

Car myCar("Toyota", 'E', 0);

myCar.display();

myCar.turn();

myCar.display();

myCar.turn('N');

myCar.display();

myCar.move(10);

myCar.display();

return 0;

}

**Program 7**

#include <iostream>

using namespace std;

void find(int arr[10]) {

int max = arr[0];

int max\_index = 0;

int second\_max = arr[0];

int second\_max\_index = 0;

for (int i = 1; i <= 9; i++) {

if (arr[i] > max) {

second\_max = max;

second\_max\_index = max\_index;

max = arr[i];

max\_index = i;

} else if (arr[i] > second\_max) {

second\_max = arr[i];

second\_max\_index = i;

}

}

cout << "Largest Element: " << max << " at index " << max\_index << endl;

cout << "Second Largest Element: " << second\_max << " at index " << second\_max\_index << endl;

}

int main() {

int arr[10];

cout << "Enter 10 integer values:" << endl;

for (int i = 0; i < 10; i++) {

cin >> arr[i];

}

find(arr);

return 0;

}

**Program 8**

#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("Qandeel asim", 1, 50000.0);

Employee F2(“ saman gulzar", 3, 60000.0);

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

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

return 0;

}