OBJECT ORIENTED PROGRAM

NAME: ABEER AWAIS

REGISTRATION NO. : 2023-BS-AI-022

PROGRAM 1

```
#include <iostream>
using namespace std;
class abeer{
	int r,i;
	public:
	void set(){
		cout<<"Enter Real number: ";
		cin>>r;
		cout<<"Enter Imaginary number: ";
		cin>>i;
	}
	void display(){
		cout<<"Number is "<<r<<" + "<<i <<"i";
	}
};
int main(){
	abeer obj;
	obj.set();
	obj.display();
}
```

PROGRAM 2

```
#include <iostream>
```

```
using namespace std;
class Car{
    string name;
    char direction;
    int position;
public:
    Car(string n, char d, int p) {
        name = n;
        direction = d;
        position = p;
    }
    void turn() {
        switch (direction) {
            case 'N':
                direction = 'E';
                break;
            case 'E':
                direction = 'S';
                break;
            case 'S':
                direction = 'W';
                break;
            case 'W':
                direction = 'N';
                break;
        }
    }
    void turn(char newDirection) {
        if (newDirection == 'N' || newDirection == 'E' || newDirection == 'S' ||
newDirection == 'W') {
            direction = newDirection;
        } else {
            cout << "Invalid direction!" << endl;
        }
    }
    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;
        }
    }
    void show() {
        cout << "Car Name: " << name << endl;
        cout << "Direction: " << direction << endl;
        cout << "Position: " << position << endl;
    }
};
int main(){
    Car myCar("LEXUS", 'E', 0);
    myCar.show();
    myCar.turn();
    myCar.show();
    myCar.turn('N');
    myCar.show();
    myCar.move(10);
    myCar.show();
    return 0;
  }
```

<u>**PROGRAM 3**</u>

```
#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 4

```
#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 5

```
#include <iostream>
#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("Abeer"), F3(2000), F4(2500, "Awais");
    F1.display();
    F2.display();
    F3.display();
    F4.display();
    return 0;
}
```

PROGRAM 6

```
#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("Bike");
    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 7

```
#include <iostream>
#include <string>
using namespace std;
class Employee {
public:
    string name;
    int id;
    float salary;
    static int totalEmployees;
    static float totalSalary;
public:
    Employee(string name, int id, float salary) : name(name), id(id), salary(salary) {
        totalEmployees++;
        totalSalary += salary;
    }
    ~Employee() {
        totalEmployees--;
        totalSalary -= salary;
    }
    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;
    }
    string getName() const {
        return name;
    }
    int getId() const {
        return id;
    }
    float getSalary() const {
        return salary;
    }
    static float averageSalary() {
        if (totalEmployees == 0) {
            return 0.0f;
        }
        return totalSalary / totalEmployees;
    }
};
int Employee::totalEmployees = 0;
float Employee::totalSalary = 0.0f;
int main() {
    Employee F1("Abeer Awais", 1, 50000.0);
    Employee F2("Aiza Awais", 2, 60000.0);
    cout << "Total employees: " << Employee::totalEmployees << endl;
    cout << "Average salary: $" << Employee::averageSalary() << endl;
    return 0;
}
```

PROGRAM 8

```
#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 << "\tCar Pool Savings Calculator\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;                        }
```