#include <iostream>

#include <string>

#include <stdexcept>

using namespace std;

class Student {

private:

string name;

int roll\_number;

string student\_class;

char division;

string dob;

string blood\_group;

string contact\_address;

string telephone\_number;

string driving\_license\_no;

public:

// Static member to count the number of students

static int student\_count;

// Default constructor

Student() : name(""), roll\_number(0), student\_class(""), division(' '), dob(""), blood\_group(""), contact\_address(""), telephone\_number(""), driving\_license\_no("") {

student\_count++;

}

// Parameterized constructor

Student(string n, int roll, string cls, char div, string birth, string blood, string address, string phone, string license)

: name(n), roll\_number(roll), student\_class(cls), division(div), dob(birth), blood\_group(blood), contact\_address(address), telephone\_number(phone), driving\_license\_no(license) {

student\_count++;

}

// Copy constructor

Student(const Student& other) {

name = other.name;

roll\_number = other.roll\_number;

student\_class = other.student\_class;

division = other.division;

dob = other.dob;

blood\_group = other.blood\_group;

contact\_address = other.contact\_address;

telephone\_number = other.telephone\_number;

driving\_license\_no = other.driving\_license\_no;

student\_count++;

}

// Destructor

~Student() {

student\_count--;

}

// Static member function to get student count

static int getStudentCount() {

return student\_count;

}

// Friend class declaration

friend class StudentDatabase;

};

int Student::student\_count = 0;

class StudentDatabase {

public:

static void displayStudentInfo(const Student& student) {

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

cout << "Roll Number: " << student.roll\_number << endl;

cout << "Class: " << student.student\_class << endl;

cout << "Division: " << student.division << endl;

cout << "Date of Birth: " << student.dob << endl;

cout << "Blood Group: " << student.blood\_group << endl;

cout << "Contact Address: " << student.contact\_address << endl;

cout << "Telephone Number: " << student.telephone\_number << endl;

cout << "Driving License No: " << student.driving\_license\_no << endl;

}

static void addStudent(Student& student) {

// Example of exception handling

try {

if (student.name.empty() || student.roll\_number <= 0) {

throw invalid\_argument("Invalid student information provided.");

}

// Additional logic for adding student can be implemented here

cout << "Student added successfully." << endl;

} catch (const invalid\_argument& e) {

cerr << "Error: " << e.what() << endl;

}

}

};

int main() {

// Default constructor

Student student1;

// Parameterized constructor

Student student2("John Doe", 101, "FY", 'A', "01/01/2000", "B+", "123 Street, City", "1234567890", "DL12345");

// Copy constructor

Student student3(student2);

// Add students to the database

StudentDatabase::addStudent(student1);

StudentDatabase::addStudent(student2);

// Display student information

cout << "\nStudent 1 Information:" << endl;

StudentDatabase::displayStudentInfo(student1);

cout << "\nStudent 2 Information:" << endl;

StudentDatabase::displayStudentInfo(student2);

cout << "\nStudent 3 (Copy of Student 2) Information:" << endl;

StudentDatabase::displayStudentInfo(student3);

// Display student count

cout << "\nTotal number of students: " << Student::getStudentCount() << endl;

return 0;

}