```
/*Develop a program in C++ to create a database of student’s information system
containing the following information: Name, roll number, Class, Division, Date of
Birth, Blood group, Contact address, Telephone number, Driving license no. and
other. Construct the database with suitable member functions. Make use of
constructor, default constructor, copy constructor, destructor, static member
functions, friend class, this pointer, inline code and dynamic memory allocation
operators-new and delete as well as exception handling*/

#include <iostream>
#include <string>
#include <exception>

using namespace std;

// Forward declaration of class Student for friend class
class Student;

// Friend class for accessing private members of Student
class FriendClass {
public:
    void displayStudentInfo(Student& student);
};

class Student {
private:
    string name;
    int rollNumber;
    string studentClass;
    char division;
    string dob;
    string bloodGroup;
    string contactAddress;
    long long telephoneNumber;
    string drivingLicense;

    // Static member to track the number of students
    static int studentCount;

public:
    // Default constructor
    Student() : name(""), rollNumber(0), studentClass(""), division(' '), dob(""),
                bloodGroup(""), contactAddress(""), telephoneNumber(0),
drivingLicense("") {
        studentCount++;
    }

    // Parameterized constructor
    Student(string name, int rollNumber, string studentClass, char division, string
dob,
            string bloodGroup, string contactAddress, long long telephoneNumber,
string drivingLicense)
        : name(name), rollNumber(rollNumber), studentClass(studentClass),
division(division),
          dob(dob), bloodGroup(bloodGroup), contactAddress(contactAddress),
          telephoneNumber(telephoneNumber), drivingLicense(drivingLicense) {
        studentCount++;
    }

    // Copy constructor
    Student(Student& other) {
        this->name = other.name;
        this->rollNumber = other.rollNumber;
        this->studentClass = other.studentClass;
        this->division = other.division;
        this->dob = other.dob;
        this->bloodGroup = other.bloodGroup;
        this->contactAddress = other.contactAddress;
        this->telephoneNumber = other.telephoneNumber;
        this->drivingLicense = other.drivingLicense;
        studentCount++;
    }

    // Destructor
    ~Student() {
        studentCount--;
    }

    // Inline function to set data
    inline void setData(string name, int rollNumber, string studentClass, char
division, string dob,
                        string bloodGroup, string contactAddress, long long
telephoneNumber, string drivingLicense) {
        this->name = name;
        this->rollNumber = rollNumber;
        this->studentClass = studentClass;
        this->division = division;
        this->dob = dob;
        this->bloodGroup = bloodGroup;
        this->contactAddress = contactAddress;
        this->telephoneNumber = telephoneNumber;
        this->drivingLicense = drivingLicense;
    }

    // Inline function to display data
    inline void displayData() {
        cout << "Name: " << name << endl;
        cout << "Roll Number: " << rollNumber << endl;
        cout << "Class: " << studentClass << endl;
        cout << "Division: " << division << endl;
        cout << "Date of Birth: " << dob << endl;
        cout << "Blood Group: " << bloodGroup << endl;
        cout << "Contact Address: " << contactAddress << endl;
        cout << "Telephone Number: " << telephoneNumber << endl;
        cout << "Driving License: " << drivingLicense << endl;
    }

    // Static member function to get student count
    static int getStudentCount() {
        return studentCount;
    }

    // Friend class declaration
    friend class FriendClass;
};

// Initialize static member
int Student::studentCount = 0;
```

```
// Friend class function to display student's private data
void FriendClass::displayStudentInfo(Student& student) {
    cout << "[Friend Access] Name: " << student.name << ", Roll Number: " <<
student.rollNumber << endl;
}

int main() {
    try {
        // Dynamic memory allocation for student objects
        Student* student1 = new Student("Alice", 101, "10th Grade", 'A', "2005-04-
15", "O+", "123 Main Street",
                                         9876543210, "DL12345");
        Student* student2 = new Student(*student1);  // Copy constructor

        cout << "Student 1 Details:" << endl;
        student1->displayData();
        cout << endl;

        cout << "Student 2 (Copy) Details:" << endl;
        student2->displayData();
        cout << endl;

        cout << "Total Students: " << Student::getStudentCount() << endl;

        // Use of friend class
        FriendClass friendObj;
        friendObj.displayStudentInfo(*student1);

        // Deallocate memory
        delete student1;
        delete student2;
    } catch (bad_alloc& e) {
        cerr << "Memory allocation failed: " << e.what() << endl;
    } catch (exception& e) {
        cerr << "An error occurred: " << e.what() << endl;
    }

    cout << "Program ended." << endl;
    return 0;
}
```