



Sir Syed CASE
Institute of Technology

OOP Lab 11
Abstract class

SUBMITTED BY:

UBAID AHMAD

ROLL NO:

2410-0011

SUBMITTED TO:

Ma'am Laiba Tanveer

DATE:

19/12/2025

Task :

You're tasked with developing a student management system for a university. The system should be able

to handle various tasks related to student information, such as registration, course enrollment, and GPA

calculation. To ensure proper design and implementation, you decide to use abstract classes and

polymorphism.

Design and implement a C++ program to fulfill the following requirements:

1. Define an abstract class named Student with the following characteristics:

- ☐ It should have pure virtual functions for displaying student information (virtual void display() = 0;) and calculating the GPA (virtual float calculateGPA() = 0;).
- ☐ The class should contain data members to store basic information about a student, such as name, ID, and age.

2. Implement two derived classes: UndergraduateStudent and GraduateStudent, each representing a specific type of student. Each class should inherit from the Student class and provide implementations for the display() and calculateGPA() functions:

☐ For the UndergraduateStudent class:

- ☐ Implement the calculateGPA() function to calculate the GPA based on the grades obtained in undergraduate courses.
- ☐ Implement the display() function to display the details of the undergraduate student, including their name, ID, age, and GPA.

☐ For the GraduateStudent class:

- ☐ Implement the calculateGPA() function to calculate the GPA based on the grades obtained in graduate courses.
- ☐ Implement the display() function to display the details of the graduate student, including their name, ID, age, and GPA.

3. Write a main program to test your implementation:

- ☐ Create objects of each student class (UndergraduateStudent, GraduateStudent).
- ☐ Prompt the user to input information for each student, including name, ID, age, and grades.
- ☐ Calculate and display the GPA of each student using polymorphism.
- ☐ Ensure that the program compiles without errors and produces correct results for different types of students.

CODE:

```
#include <iostream>
using namespace std;
const int n = 5;
class student {
protected:
    string name;
    int id;
    int age;
public:
    virtual void display() = 0;
    virtual float calculateGpa() = 0;
};
class UnderG : public student {
public:
    int grades[n];
    void readStudent() {
        cin.ignore();
```

```
    cout << "Enter name: ";
    getline(cin, name);
    cout << "Enter ID : ";
    cin >> id;
    cout << "Enter age : ";
    cin >> age;
    cout << "Enter marks in 5 subjects: ";
    for (int i = 0; i < n; i++) {
        cin >> grades[i];
    }
}

float calculateGpa() {
    float totalPercent = 0;
    for (int i = 0; i < n; i++) {
        totalPercent += grades[i];
    }

    float avgPercent = totalPercent / n;
    return (avgPercent / 100) * 4; // GPA out of 4
}

void display() {
    cout << "\n--- Undergraduate Student ---\n";
    cout << "Name : " << name << endl;
    cout << "ID   : " << id << endl;
    cout << "Age  : " << age << endl;
```

```

        cout << "Grades: ";
        for (int i = 0; i < n; i++) {
            cout << grades[i] << " ";
        }
        cout << "\nGPA  : " << calculateGpa() << endl;
    }
};

class Graduate : public student {
public:
    int grades[n];
    void readStudent() {
        cin.ignore();
        cout << "Enter name: ";
        getline(cin, name);
        cout << "Enter ID  : ";
        cin >> id;
        cout << "Enter age : ";
        cin >> age;
        cout << "Enter marks in 5 subjects: ";
        for (int i = 0; i < n; i++) {
            cin >> grades[i];
        }
    }

    float calculateGpa() {
        float totalPercent = 0;

```

```

    for (int i = 0; i < n; i++) {
        totalPercent += grades[i];
    }

    float avgPercent = totalPercent / n;
    // Graduate grading slightly stricter
    return (avgPercent / 100) * 4;
}

void display() {
    cout << "\n--- Graduate Student ---\n";
    cout << "Name : " << name << endl;
    cout << "ID   : " << id << endl;
    cout << "Age  : " << age << endl;
    cout << "Grades: ";
    for (int i = 0; i < n; i++) {
        cout << grades[i] << " ";
    }
    cout << "\nGPA  : " << calculateGpa() << endl;
}

};

// ===== MAIN =====

int main() {
    UnderG ug1, ug2;
    Graduate g1, g2;

```


```
student* s[4];

s[0] = &ug1;
s[1] = &ug2;
s[2] = &g1;
s[3] = &g2;

cout << "\nEnter Undergraduate Student 1\n";
ug1.readStudent();
cout << "\nEnter Undergraduate Student 2\n";
ug2.readStudent();
cout << "\nEnter Graduate Student 1\n";
g1.readStudent();
cout << "\nEnter Graduate Student 2\n";
g2.readStudent();

cout << "\n\n==== STUDENT DETAILS (Using Polymorphism) =====\n";
for (int i = 0; i < 4; i++) {
    s[i]->display();
}
return 0;
}
```

OUTPUT:

 D:\4th semester data\OOP\lab _11 Abstract class\labtask.exe

Enter Undergraduate Student 1

Enter name: ubaid

Enter ID : 11

Enter age : 23

Enter marks in 5 subjects: 24

25

26

27

28

Enter Undergraduate Student 2

Enter name: didar

Enter ID : 22

Enter age : 20

Enter marks in 5 subjects: 40 50 60 70 80

Enter Graduate Student 1

Enter name: bangash

Enter ID : 33

Enter age : 40

Enter marks in 5 subjects: 25 56 72 31 53

Enter Graduate Student 2

Enter name: reyan

Enter ID : 44

Enter age : 50

Enter marks in 5 subjects: 30 40 50 60 20

Select D:\4th semesteer data\OOP\lab _11 Abstract class\labtask.exe

```
===== STUDENT DETAILS (Using Polymorphism) =====  
  
--- Undergraduate Student ---  
Name   : ubaid  
ID      : 11  
Age     : 23  
Grades: 24 25 26 27 28  
GPA     : 1.04  
  
--- Undergraduate Student ---  
Name   : didar  
ID      : 22  
Age     : 20  
Grades: 40 50 60 70 80  
GPA     : 2.4  
  
--- Graduate Student ---  
Name   : bangash  
ID      : 33  
Age     : 40  
Grades: 25 56 72 31 53  
GPA     : 1.896  
  
--- Graduate Student ---  
Name   : reyan  
ID      : 44  
Age     : 50  
Grades: 30 40 50 60 20  
GPA     : 1.6  
  
-----  
Process exited after 90.61 seconds with return value 0  
Press any key to continue . . .
```

END,,,
