/\* Program in C++ object concepts for declaring the student name, student code, and 3 marks and calculate the total, average, result and grade. Create required member functions for getting inputs, calculation and output the data. Create 2 objects in the main()

\*/

/\*

C++ program using **object-oriented programming (OOP)** concepts to:

 Declare **student name, student code, and 3 marks**

 Calculate **total, average, result (pass/fail), and grade**

 Use **member functions** for input, calculation, and output

 Create **two objects** in main()

\*/

#include <iostream>

#include <string>

using namespace std;

class Student {

private:

string name;

string code;

float mark1, mark2, mark3;

float total, average;

string result;

char grade;

public:

// Function to get student data

void getInput() {

cout << "Enter Student Name: ";

getline(cin, name);

cout << "Enter Student Code: ";

getline(cin, code);

cout << "Enter Mark 1: ";

cin >> mark1;

cout << "Enter Mark 2: ";

cin >> mark2;

cout << "Enter Mark 3: ";

cin >> mark3;

}

// Function to calculate total, average, result and grade

void calculate() {

total = mark1 + mark2 + mark3;

average = total / 3;

// Result logic

if (mark1 >= 35 && mark2 >= 35 && mark3 >= 35)

result = "Pass";

else

result = "Fail";

// Grade logic

if (result == "Pass") {

if (average >= 90)

grade = 'A';

else if (average >= 75)

grade = 'B';

else if (average >= 60)

grade = 'C';

else

grade = 'D';

} else {

grade = 'F';

}

}

// Function to display student data

void display() {

cout << "\n--- Student Details ---" << endl;

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

cout << "Code : " << code << endl;

cout << "Mark 1 : " << mark1 << endl;

cout << "Mark 2 : " << mark2 << endl;

cout << "Mark 3 : " << mark3 << endl;

cout << "Total : " << total << endl;

cout << "Average : " << average << endl;

cout << "Result : " << result << endl;

cout << "Grade : " << grade << endl;

}

};

int main() {

Student s1, s2;

cout << "Enter details for Student 1:" << endl;

s1.getInput();

s1.calculate();

s1.display();

cout << "\nEnter details for Student 2:" << endl;

s2.getInput();

s2.calculate();

s2.display();

return 0;

}

**Sample Output**

Enter details for Student 1:

Enter Student Name: Alice

Enter Student Code: A001

Enter Mark 1: 90

Enter Mark 2: 85

Enter Mark 3: 92

--- Student Details ---

Name : Alice

Code : A001

Mark 1 : 90

Mark 2 : 85

Mark 3 : 92

Total : 267

Average : 89

Result : Pass

Grade : B

The above student program with constructors and destructors

* **Default constructor**
* ✅ **Parameterized constructor**
* ✅ **Copy constructor**
* ✅ **Destructor**
* ✅ **3 objects** created in main() using different constructors

#include <iostream>

#include <string>

using namespace std;

class Student {

private:

string name;

string code;

float mark1, mark2, mark3;

float total, average;

string result;

char grade;

public:

// Default Constructor

Student() {

name = "No Name";

code = "000";

mark1 = mark2 = mark3 = 0;

total = average = 0;

result = "Not Calculated";

grade = 'N';

cout << "[Default Constructor Called]" << endl;

}

// Parameterized Constructor

Student(string n, string c, float m1, float m2, float m3) {

name = n;

code = c;

mark1 = m1;

mark2 = m2;

mark3 = m3;

calculate();

cout << "[Parameterized Constructor Called]" << endl;

* }

// Copy Constructor

Student(const Student &s) {

name = s.name;

code = s.code;

mark1 = s.mark1;

mark2 = s.mark2;

mark3 = s.mark3;

total = s.total;

average = s.average;

result = s.result;

grade = s.grade;

cout << "[Copy Constructor Called]" << endl;

}

// Destructor

~Student() {

cout << "[Destructor Called for: " << name << "]" << endl;

}

// Input function

void getInput() {

cout << "Enter Student Name: ";

getline(cin, name);

cout << "Enter Student Code: ";

getline(cin, code);

cout << "Enter Mark 1: ";

cin >> mark1;

cout << "Enter Mark 2: ";

cin >> mark2;

cout << "Enter Mark 3: ";

cin >> mark3;

cin.ignore(); // clear buffer

calculate();

}

// Calculation function

void calculate() {

total = mark1 + mark2 + mark3;

average = total / 3;

if (mark1 >= 35 && mark2 >= 35 && mark3 >= 35)

result = "Pass";

else

result = "Fail";

if (result == "Pass") {

if (average >= 90)

grade = 'A';

else if (average >= 75)

grade = 'B';

else if (average >= 60)

grade = 'C';

else

grade = 'D';

} else {

grade = 'F';

}

}

// Display function

void display() {

cout << "\n--- Student Details ---" << endl;

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

cout << "Code : " << code << endl;

cout << "Mark 1 : " << mark1 << endl;

cout << "Mark 2 : " << mark2 << endl;

cout << "Mark 3 : " << mark3 << endl;

cout << "Total : " << total << endl;

cout << "Average : " << average << endl;

cout << "Result : " << result << endl;

cout << "Grade : " << grade << endl;

}

};

int main() {

// Object 1: Default constructor

Student s1;

s1.getInput();

s1.display();

// Object 2: Parameterized constructor

Student s2("John", "S102", 85, 75, 95);

s2.display();

// Object 3: Copy constructor (copy of s2)

Student s3 = s2;

s3.display();

return 0;

}

**Sample output**

[Default Constructor Called]

Enter Student Name: Alice

Enter Student Code: S101

Enter Mark 1: 88

Enter Mark 2: 92

Enter Mark 3: 79

--- Student Details ---

Name : Alice

Code : S101

...

[Parameterized Constructor Called]

--- Student Details ---

Name : John

Code : S102

...

[Copy Constructor Called]

--- Student Details ---

Name : John

Code : S102

...

[Destructor Called for: John]

[Destructor Called for: John]

[Destructor Called for: Alice]

**Apply OOP Inheritance Concept:**

* **Base class**: Person — contains student name and code
* **Derived class**: Student — inherits from Person, and adds marks, calculation logic, and display

**C++ Program Using Inheritance**

#include <iostream>

#include <string>

using namespace std;

// Base class

class Person {

protected:

string name;

string code;

public:

// Function to input name and code

void getPersonData() {

cout << "Enter Student Name: ";

cin >> name;

cout << "Enter Student Code: ";

cin >> code;

}

// Function to display name and code

void displayPersonData() {

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

cout << "Code : " << code << endl;

}

};

// Derived class

class Student : public Person {

private:

float mark1, mark2, mark3;

float total, average;

string result;

char grade;

public:

// Function to get student marks

void getStudentData() {

getPersonData(); // call base class input

cout << "Enter Mark 1: ";

cin >> mark1;

cout << "Enter Mark 2: ";

cin >> mark2;

cout << "Enter Mark 3: ";

cin >> mark3;

cin.ignore(); // clear buffer

calculate();

}

// Calculation logic

void calculate() {

total = mark1 + mark2 + mark3;

average = total / 3;

result = (mark1 >= 35 && mark2 >= 35 && mark3 >= 35) ? "Pass" : "Fail";

if (result == "Pass") {

if (average >= 90)

grade = 'A';

else if (average >= 75)

grade = 'B';

else if (average >= 60)

grade = 'C';

else

grade = 'D';

} else {

grade = 'F';

}

}

// Display all student data

void displayStudentData()

{

cout << "\n--- Student Details ---" << endl;

displayPersonData(); // call base class display

cout << "Mark 1 : " << mark1 << endl;

cout << "Mark 2 : " << mark2 << endl;

cout << "Mark 3 : " << mark3 << endl;

cout << "Total : " << total << endl;

cout << "Average : " << average << endl;

cout << "Result : " << result << endl;

cout << "Grade : " << grade << endl;

}

};

// Main function

int main() {

Student s1;

cout << "Enter details for Student 1:" << endl;

s1.getStudentData();

s1.displayStudentData();

return 0;

}

### Inheritance Used Here:

* Student **inherits** name and code from Person using protected access modifier
* getPersonData() and displayPersonData() are reused inside Student class methods

**Sample Output**

Enter details for Student 1:

Enter Student Name: Alice

Enter Student Code: S101

Enter Mark 1: 88

Enter Mark 2: 79

Enter Mark 3: 91

**C++ Program Using Inheritance with constructor**

#include <iostream>

#include <string>

using namespace std;

// Base class

class Person {

protected:

string name;

string code;

public:

// Default constructor

Person() {

name = "No Name";

code = "000";

}

// Parameterized constructor

Person(string n, string c) {

name = n;

code = c;

}

// Function to input name and code

void getPersonData() {

cout << "Enter Student Name: ";

getline(cin, name);

cout << "Enter Student Code: ";

getline(cin, code);

}

// Function to display name and code

void displayPersonData() const {

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

cout << "Code : " << code << endl;

}

};

// Derived class

class Student : public Person {

private:

float mark1, mark2, mark3;

float total, average;

string result;

char grade;

public:

// Default constructor

Student() : Person() {

mark1 = mark2 = mark3 = 0;

total = average = 0;

result = "Not Calculated";

grade = 'N';

}

// Parameterized constructor

Student(string n, string c, float m1, float m2, float m3) : Person(n, c) {

mark1 = m1;

mark2 = m2;

mark3 = m3;

calculate();

}

// Function to get student marks

void getStudentData() {

getPersonData(); // call base class input

cout << "Enter Mark 1: ";

cin >> mark1;

cout << "Enter Mark 2: ";

cin >> mark2;

cout << "Enter Mark 3: ";

cin >> mark3;

cin.ignore(); // clear buffer

calculate();

}

// Calculation logic

void calculate() {

total = mark1 + mark2 + mark3;

average = total / 3;

result = (mark1 >= 35 && mark2 >= 35 && mark3 >= 35) ? "Pass" : "Fail";

if (result == "Pass") {

if (average >= 90)

grade = 'A';

else if (average >= 75)

grade = 'B';

else if (average >= 60)

grade = 'C';

else

grade = 'D';

} else {

grade = 'F';

}

}

// Display all student data

void displayStudentData() {

cout << "\n--- Student Details ---" << endl;

displayPersonData(); // call base class display

cout << "Mark 1 : " << mark1 << endl;

cout << "Mark 2 : " << mark2 << endl;

cout << "Mark 3 : " << mark3 << endl;

cout << "Total : " << total << endl;

cout << "Average : " << average << endl;

cout << "Result : " << result << endl;

cout << "Grade : " << grade << endl;

}

};

// Main function

int main() {

// Object 1: using default constructor and user input

Student s1;

cout << "Enter details for Student 1:" << endl;

s1.getStudentData();

s1.displayStudentData();

// Object 2: using parameterized constructor

Student s2("John", "S102", 90, 85, 88);

s2.displayStudentData();

return 0;

}

--- Student Details ---

Name : Alice

Code : S101

Mark 1 : 88

Mark 2 : 79

Mark 3 : 91

Total : 258

Average : 86

Result : Pass

Grade : B

--- Student Details ---

Name : John

Code : S102

Mark 1 : 90

Mark 2 : 85

Mark 3 : 88

Total : 263

Average : 87.6667

Result : Pass

Grade : B

**C++ program using function overriding** — but **without using virtual functions**

* The **base class Person** has a function display()
* The **derived class Student** has a function display() with the same name — so it's **overriding** the base class method
* Since we **don’t use virtual**, the function called depends on the **type of pointer or object**, not on runtime type

**// Function Overriding Example**

#include <iostream>

#include <string>

using namespace std;

// Base class

class Person {

protected:

string name;

string code;

public:

// Function to input data

void getInput() {

cout << "Enter Student Name: ";

getline(cin, name);

cout << "Enter Student Code: ";

getline(cin, code);

}

// Function to display (will be overridden)

void display() {

cout << "\n[Person Class Display]" << endl;

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

cout << "Code : " << code << endl;

}

};

// Derived class

class Student : public Person {

private:

float mark1, mark2, mark3, total, average;

string result;

char grade;

public:

// Input data (overriding input method separately)

void getStudentInput() {

getInput(); // base class

cout << "Enter Mark 1: ";

cin >> mark1;

cout << "Enter Mark 2: ";

cin >> mark2;

cout << "Enter Mark 3: ";

cin >> mark3;

cin.ignore(); // clear buffer

calculate();

}

// Function to calculate total, average, result, grade

void calculate() {

total = mark1 + mark2 + mark3;

average = total / 3;

result = (mark1 >= 35 && mark2 >= 35 && mark3 >= 35) ? "Pass" : "Fail";

if (result == "Pass") {

if (average >= 90)

grade = 'A';

else if (average >= 75)

grade = 'B';

else if (average >= 60)

grade = 'C';

else

grade = 'D';

} else {

grade = 'F';

}

}

// Overridden function (same name as base)

void display() {

cout << "\n[Student Class Display]" << endl;

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

cout << "Code : " << code << endl;

cout << "Mark 1 : " << mark1 << endl;

cout << "Mark 2 : " << mark2 << endl;

cout << "Mark 3 : " << mark3 << endl;

cout << "Total : " << total << endl;

cout << "Average : " << average << endl;

cout << "Result : " << result << endl;

cout << "Grade : " << grade << endl;

}

};

int main() {

// Student object using base

Student s1;

s1.getStudentInput();

s1.display(); // Calls derived class version

return 0;

}