ASSIGNMENT

**1.WAP to implement a C++ program to find out the area of the rectangle and triangle using hierarchical inheritance .**

#include <iostream>

using namespace std;

// Base class Shape

class Shape {

public:

virtual float area() {

return 0;

}

};

// Derived class Rectangle

class Rectangle : public Shape {

private:

float length, breadth;

public:

Rectangle(float l, float b) : length(l), breadth(b) {}

float area() override {

return length \* breadth;

}

};

// Derived class Triangle

class Triangle : public Shape {

private:

float base, height;

public:

Triangle(float b, float h) : base(b), height(h) {}

float area() override {

return 0.5 \* base \* height;

}

};

int main() {

Rectangle rect(5, 3);

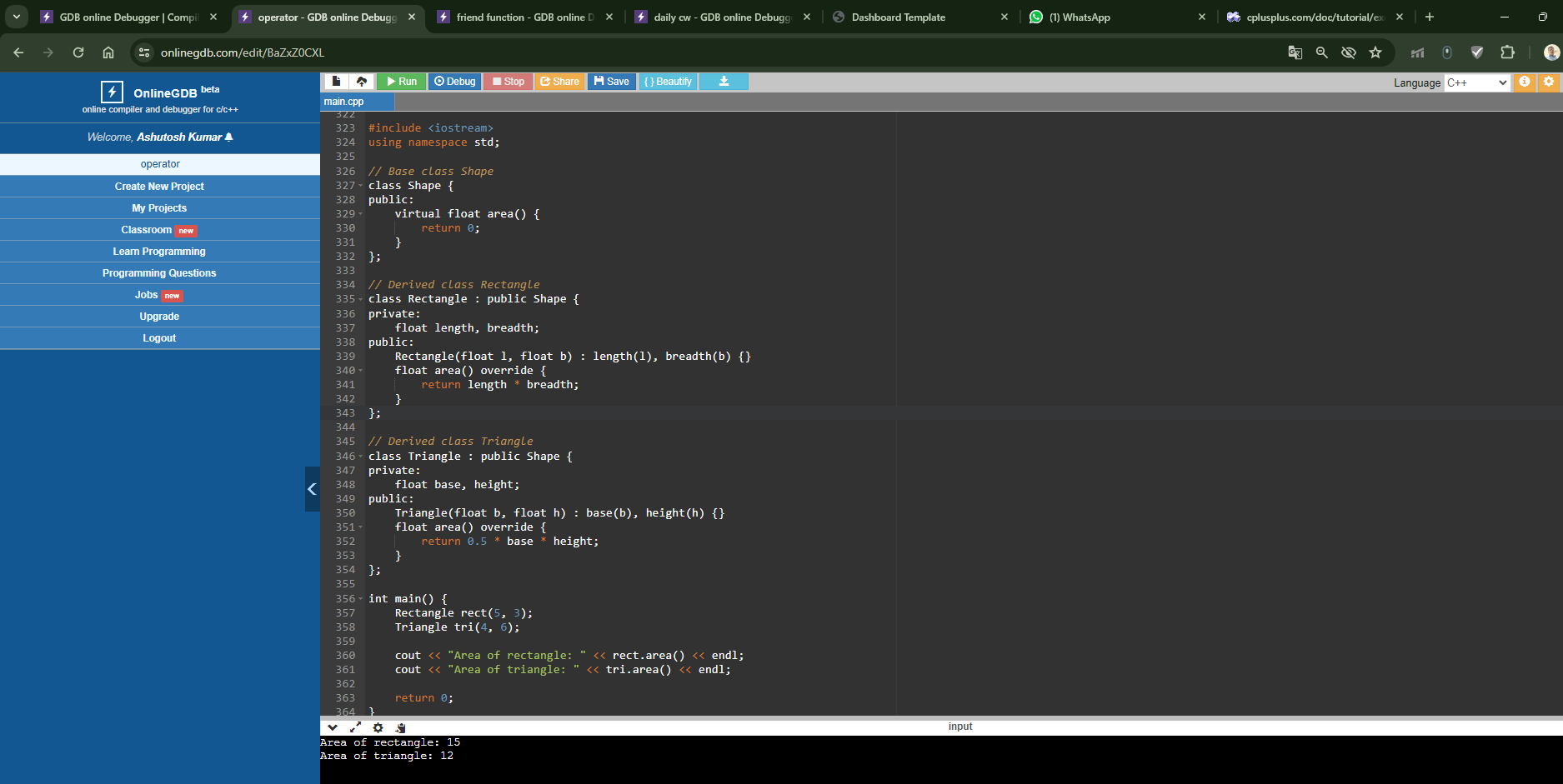
Triangle tri(4, 6);

cout << "Area of rectangle: " << rect.area() << endl;

cout << "Area of triangle: " << tri.area() << endl;

return 0;

}



**2.WAP to implement a C++ program to find out the student details using multilevel inheritance.**

#include <iostream>

using namespace std;

// Base class Student

class Student {

protected:

int rollNo;

string name;

public:

void input() {

cout << "Enter roll number: ";

cin >> rollNo;

cout << "Enter name: ";

cin >> name;

}

void display() {

cout << "Roll number: " << rollNo << endl;

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

}

};

// Derived class Exam

class Exam : public Student {

protected:

float mark1, mark2, mark3;

public:

void inputMarks() {

cout << "Enter marks of three subjects: ";

cin >> mark1 >> mark2 >> mark3;

}

void displayResult() {

cout << "Marks of three subjects: " << mark1 << ", " << mark2 << ", " << mark3 << endl;

}

};

// Derived class SportsScore

class SportsScore : public Exam {

protected:

float score;

public:

void inputScore() {

cout << "Enter sports score: ";

cin >> score;

}

void displayScore() {

cout << "Sports score: " << score << endl;

}

};

int main() {

SportsScore student1;

student1.input();

student1.inputMarks();

student1.inputScore();

cout << "\nStudent details:" << endl;

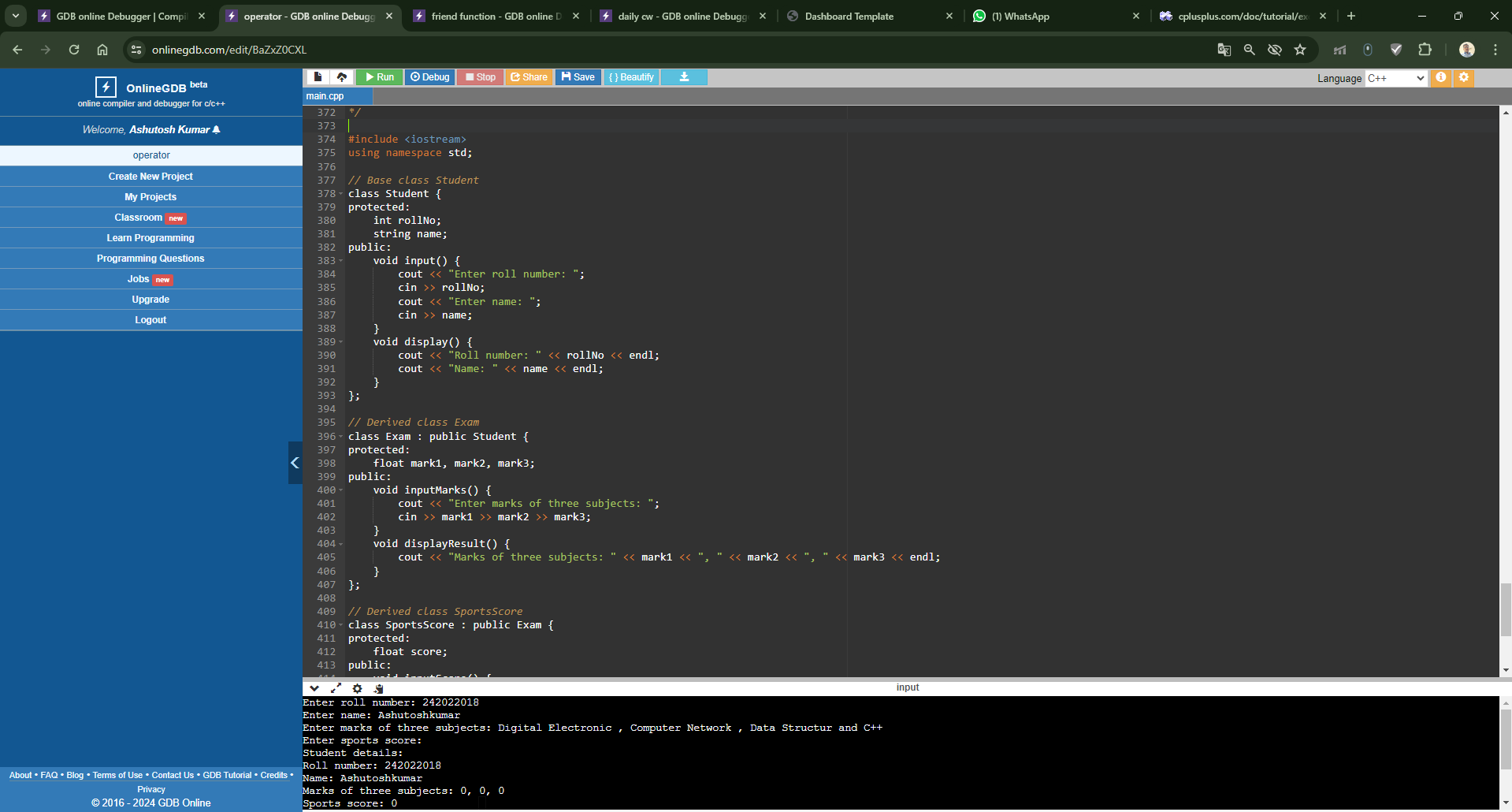
student1.display();

student1.displayResult();

student1.displayScore();

return 0;

}



**3.WAP to implement a C++ program to find out the student details and sport score using hybrid inheritance.**

#include <iostream>

using namespace std;

// Base class Student

class Student {

protected:

int rollNo;

string name;

public:

void input() {

cout << "Enter roll number: ";

cin >> rollNo;

cout << "Enter name: ";

cin >> name;

}

void display() {

cout << "Roll number: " << rollNo << endl;

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

}

};

// Derived class Exam

class Exam : virtual public Student {

protected:

float mark1, mark2, mark3;

public:

void inputMarks() {

cout << "Enter marks of three subjects: ";

cin >> mark1 >> mark2 >> mark3;

}

void displayResult() {

cout << "Marks of three subjects: " << mark1 << ", " << mark2 << ", " << mark3 << endl;

}

};

// Derived class SportsScore

class SportsScore : virtual public Student {

protected:

float score;

public:

void inputScore() {

cout << "Enter sports score: ";

cin >> score;

}

void displayScore() {

cout << "Sports score: " << score << endl;

}

};

// Derived class HybridStudent combining Exam and SportsScore

class HybridStudent : public Exam, public SportsScore {

public:

void inputAll() {

input(); // Input from base class Student

inputMarks(); // Input from class Exam

inputScore(); // Input from class SportsScore

}

void displayAll() {

display(); // Display from base class Student

displayResult();// Display from class Exam

displayScore(); // Display from class SportsScore

}

};

int main() {

HybridStudent student;

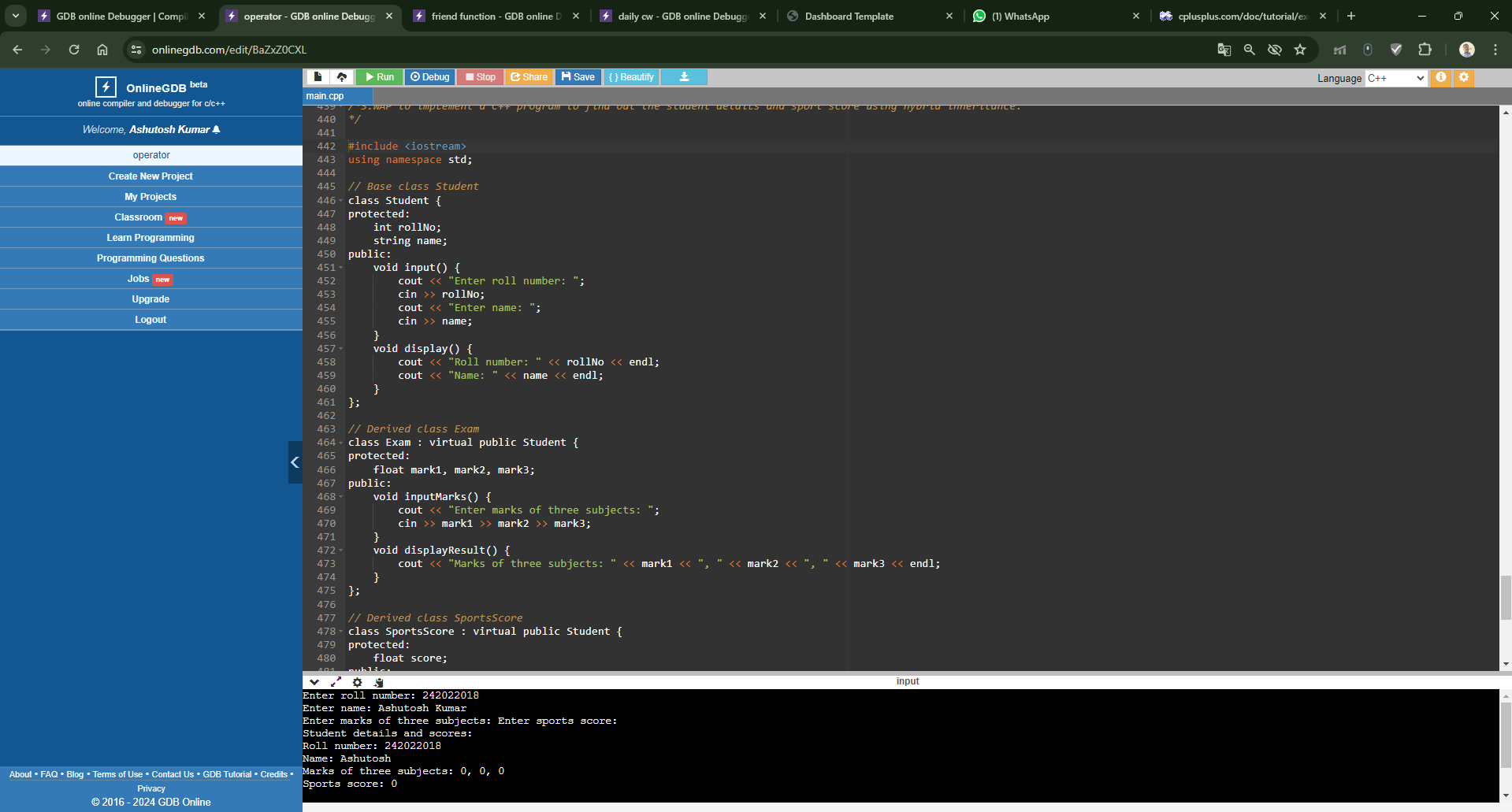
student.inputAll();

cout << "\nStudent details and scores:" << endl;

student.displayAll();

return 0;

}



**4.Implement function overriding by creating class shape through which area of figures are calculated.**

#include <iostream>

#include <cmath>

using namespace std;

// Base class Shape

class Shape {

public:

virtual float area() = 0; // Pure virtual function for calculating area

};

// Derived class Rectangle

class Rectangle : public Shape {

private:

float length, breadth;

public:

Rectangle(float l, float b) : length(l), breadth(b) {}

// Overriding area() function for rectangle

float area() override {

return length \* breadth;

}

};

// Derived class Circle

class Circle : public Shape {

private:

float radius;

public:

Circle(float r) : radius(r) {}

// Overriding area() function for circle

float area() override {

return M\_PI \* radius \* radius; // Using math constant PI for area calculation

}

};

int main() {

// Creating objects of derived classes

Rectangle rectangle(5, 4);

Circle circle(3);

// Calculating and displaying area of rectangle

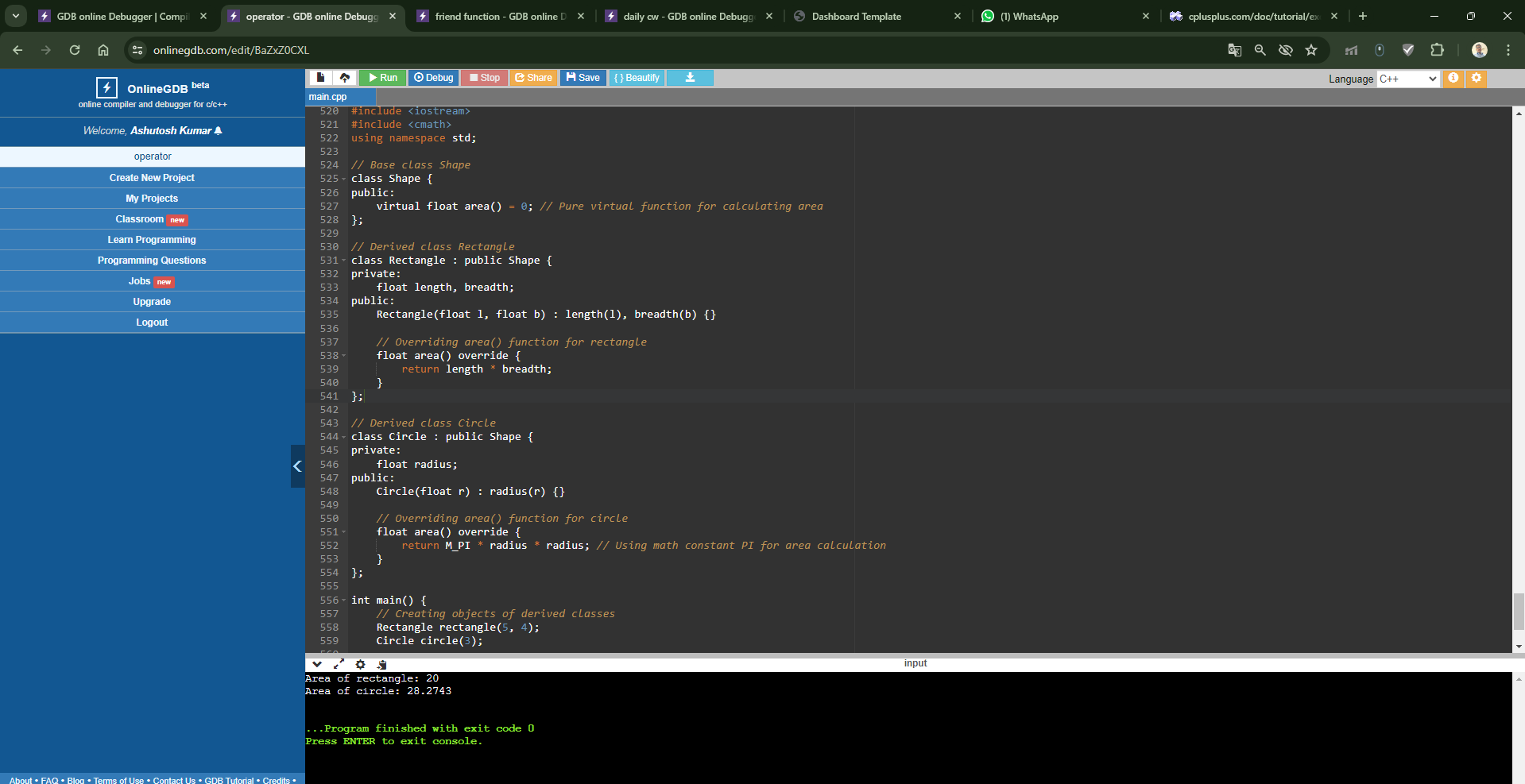
cout << "Area of rectangle: " << rectangle.area() << endl;

// Calculating and displaying area of circle

cout << "Area of circle: " << circle.area() << endl;

return 0;

}



**5.Class student contains roll number, name and course as data member and Input\_student and display\_student as member function. A derived class exam is created from the class student with publicly inherited. The derived class contains mark1, mark2, mark3 as marks of three subjects and input\_marks and display\_result as member function. Create an array of object of the exam class and display the result of 5 students. Try the same program with privately inheritance.**

#include <iostream>

using namespace std;

// Base class Student

class Student {

protected:

int rollNumber;

string name, course;

public:

void inputStudent() {

cout << "Enter Roll Number: ";

cin >> rollNumber;

cout << "Enter Name: ";

cin >> name;

cout << "Enter Course: ";

cin >> course;

}

void displayStudent() {

cout << "Roll Number: " << rollNumber << endl;

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

cout << "Course: " << course << endl;

}

};

// Derived class Exam publicly inherited from Student

class Exam : public Student {

private:

float mark1, mark2, mark3;

public:

void inputMarks() {

cout << "Enter marks for Subject 1: ";

cin >> mark1;

cout << "Enter marks for Subject 2: ";

cin >> mark2;

cout << "Enter marks for Subject 3: ";

cin >> mark3;

}

void displayResult() {

cout << "Marks for Subject 1: " << mark1 << endl;

cout << "Marks for Subject 2: " << mark2 << endl;

cout << "Marks for Subject 3: " << mark3 << endl;

}

};

int main() {

// Create an array of Exam objects

const int numStudents = 5;

Exam students[numStudents];

// Input details and marks for each student

for (int i = 0; i < numStudents; ++i) {

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

students[i].inputStudent();

students[i].inputMarks();

}

// Display details and marks for each student

for (int i = 0; i < numStudents; ++i) {

cout << "\nDetails for Student " << i + 1 << ":" << endl;

students[i].displayStudent();

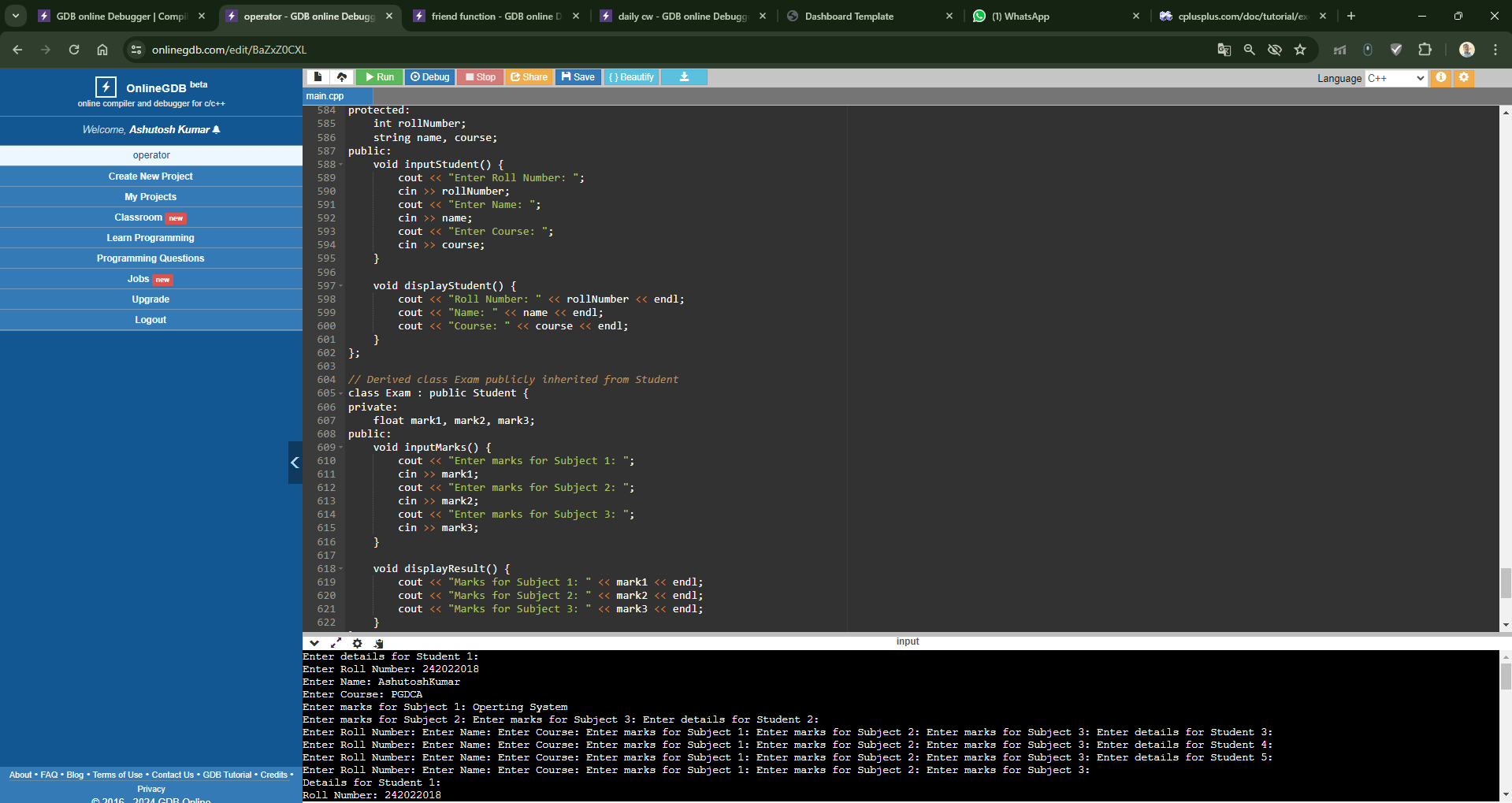
students[i].displayResult();

cout << endl;

}

return 0;

}



**6. A University and a Company have jointly taken a project. Class University contains name of the university, department to which the project is assigned, person to whom the project is assigned. A function display is there to display the information. Class Company contains name of the company, Number of Engineers assigned, amount invested to do the project. A function display is there to display the information. Class Project is inherited from University and Company. It contains type of project, duration of project, amount granted to complete the project. A function display displays the related information. Write a C++ program to implement this and display all information except amount invested by company from Project class**

#include <iostream>

#include <string>

using namespace std;

// University class

class University {

protected:

string universityName;

string department;

string projectPerson;

public:

void inputUniversity() {

cout << "Enter University Name: ";

getline(cin, universityName);

cout << "Enter Department: ";

getline(cin, department);

cout << "Enter Project Person: ";

getline(cin, projectPerson);

}

void displayUniversity() {

cout << "University Name: " << universityName << endl;

cout << "Department: " << department << endl;

cout << "Project Person: " << projectPerson << endl;

}

};

// Company class

class Company {

protected:

string companyName;

int numEngineers;

float amountInvested;

public:

void inputCompany() {

cout << "Enter Company Name: ";

getline(cin, companyName);

cout << "Enter Number of Engineers Assigned: ";

cin >> numEngineers;

cout << "Enter Amount Invested: ";

cin >> amountInvested;

cin.ignore(); // Clear input buffer

}

void displayCompany() {

cout << "Company Name: " << companyName << endl;

cout << "Number of Engineers Assigned: " << numEngineers << endl;

}

};

// Project class inherited from University and Company

class Project : public University, public Company {

private:

string projectType;

float projectDuration;

float projectGrant;

public:

void inputProject() {

cout << "Enter Project Type: ";

getline(cin, projectType);

cout << "Enter Project Duration: ";

cin >> projectDuration;

cout << "Enter Project Grant: ";

cin >> projectGrant;

cin.ignore(); // Clear input buffer

}

void displayProject() {

cout << "\nProject Information:" << endl;

displayUniversity();

displayCompany();

cout << "Project Type: " << projectType << endl;

cout << "Project Duration: " << projectDuration << " months" << endl;

cout << "Project Grant: " << projectGrant << " USD" << endl;

}

};

int main() {

Project project;

cout << "Enter Project Details:" << endl;

project.inputUniversity();

project.inputCompany();

project.inputProject();

cout << "\nProject Information (excluding company investment):" << endl;

project.displayProject();

return 0;

}

