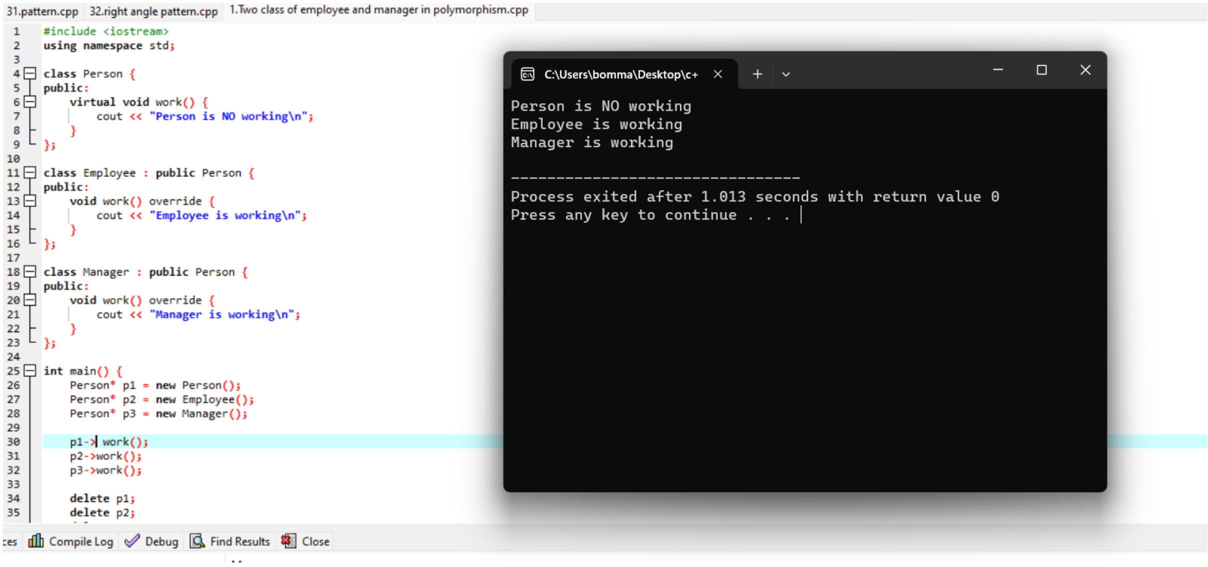
Polymorphism Programs:

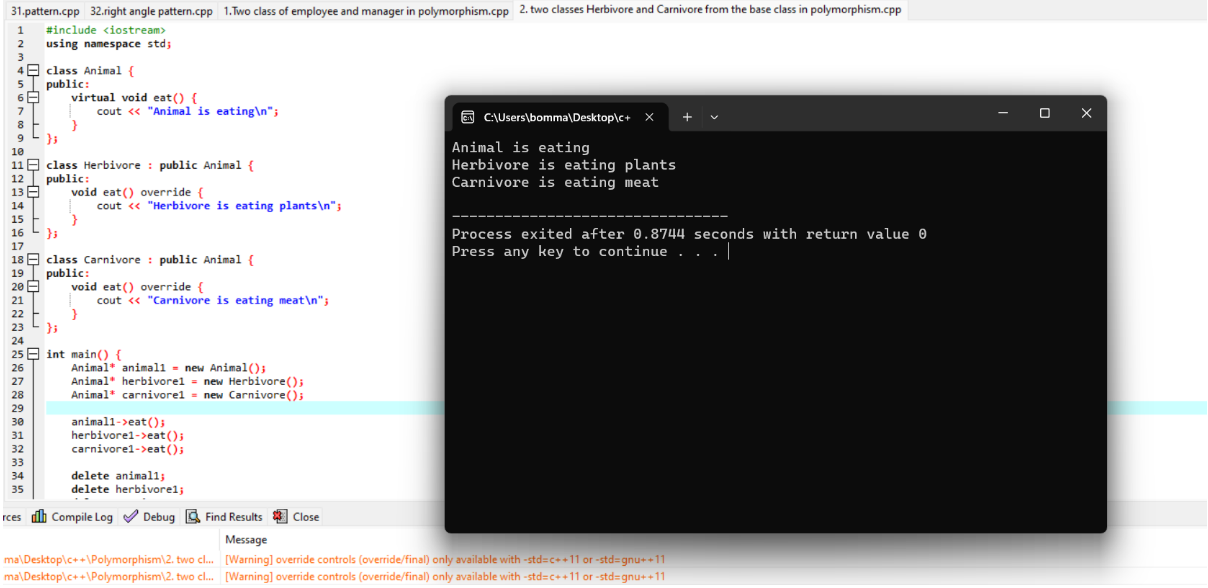
1.Create a base class called Person with a virtual function work (). Derive two classes Employee and Manager from the base class. Implement the work () function for each class


31.pattern.cpp
32.right angle pattern.cpp
1.Two class of employee and manager in polymorphism.cpp
C:\Users\bomma\Desktop\c+
ces
Compile Log
Debug
Find Results
Close

```cpp
#include <iostream>
using namespace std;

class Person {
public:
    virtual void work() {
        cout << "Person is NO working\n";
    }
};

class Employee : public Person {
public:
    void work() override {
        cout << "Employee is working\n";
    }
};

class Manager : public Person {
public:
    void work() override {
        cout << "Manager is working\n";
    }
};

int main() {
    Person* p1 = new Person();
    Person* p2 = new Employee();
    Person* p3 = new Manager();

    p1->work();
    p2->work();
    p3->work();

    delete p1;
    delete p2;
```

```
Person is NO working
Employee is working
Manager is working

--------------------------------
Process exited after 1.013 seconds with return value 0
Press any key to continue . . .
```

2.Create a base class called Animal with a virtual function eat (). Derive two classes Herbivore and Carnivore from the base class. Implement the eat function for each class.

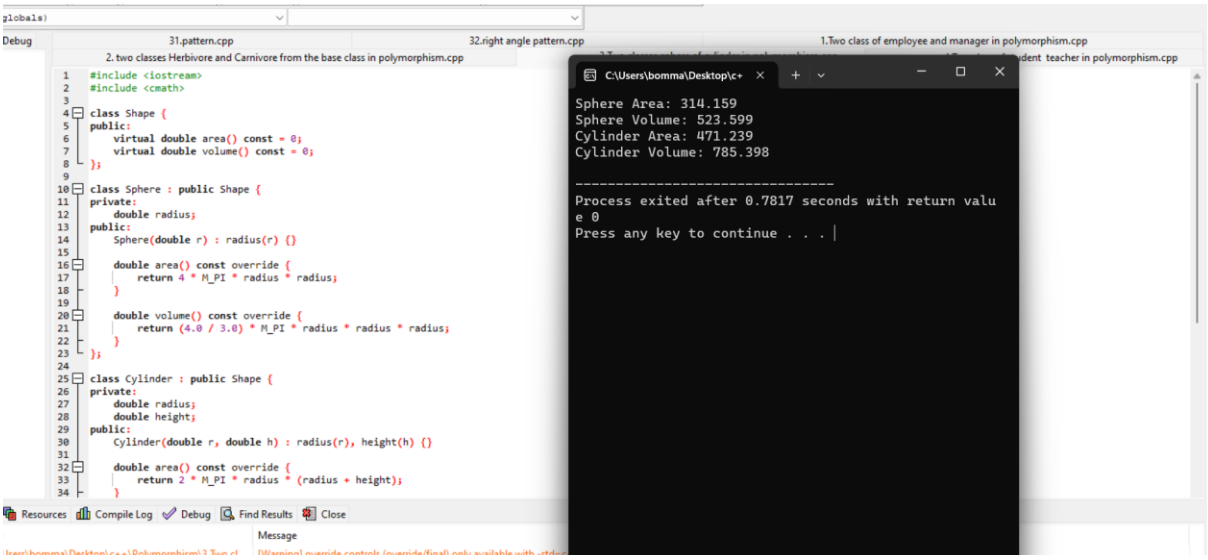

31.pattern.cpp
32.right angle pattern.cpp
1.Two class of employee and manager in polymorphism.cpp
2. two classes Herbivore and Carnivore from the base class in polymorphism.cpp
C:\Users\bomma\Desktop\c+
rces
Compile Log
Debug
Find Results
Close
Message

```cpp
#include <iostream>
using namespace std;

class Animal {
public:
    virtual void eat() {
        cout << "Animal is eating\n";
    }
};

class Herbivore : public Animal {
public:
    void eat() override {
        cout << "Herbivore is eating plants\n";
    }
};

class Carnivore : public Animal {
public:
    void eat() override {
        cout << "Carnivore is eating meat\n";
    }
};

int main() {
    Animal* animal1 = new Animal();
    Animal* herbivore1 = new Herbivore();
    Animal* carnivore1 = new Carnivore();

    animal1->eat();
    herbivore1->eat();
    carnivore1->eat();

    delete animal1;
    delete herbivore1;
```

```
Animal is eating
Herbivore is eating plants
Carnivore is eating meat

--------------------------------
Process exited after 0.8744 seconds with return value 0
Press any key to continue . . .
```

```
ma\Desktop\c++\Polymorphism\2. two cl...   [Warning] override controls (override/final) only available with -std=c++11 or -std=gnu++11
ma\Desktop\c++\Polymorphism\2. two cl...   [Warning] override controls (override/final) only available with -std=c++11 or -std=gnu++11
```

3.Create a base class called Shape with virtual functions area () and volume (). Derive two classes Sphere and Cylinder from the base class. Implement the area and volume () functions for each class

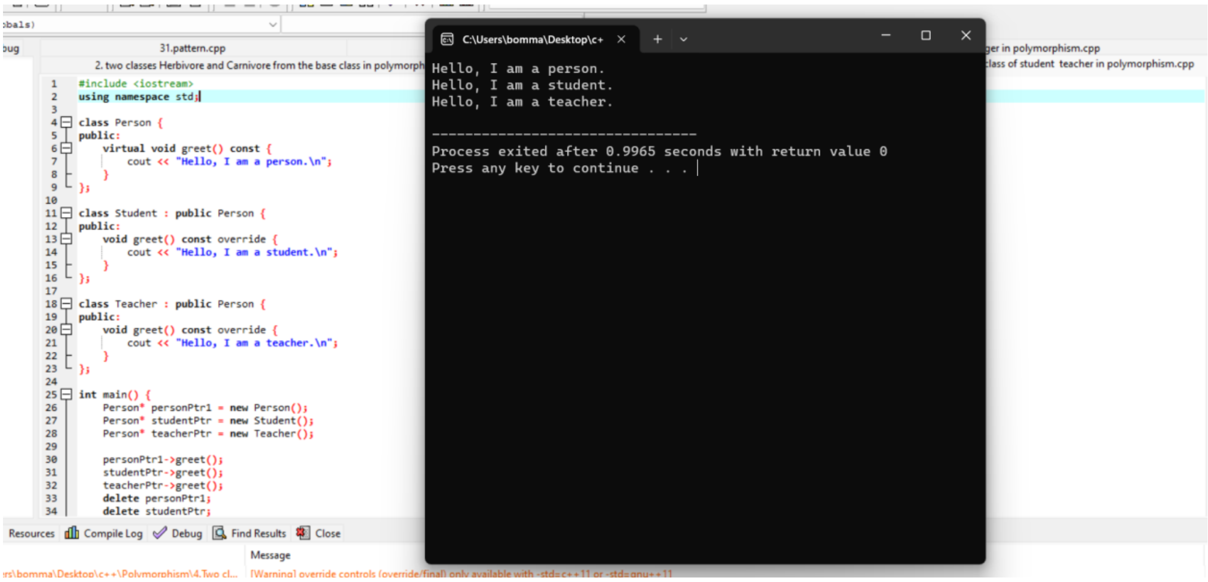


globals)
Debug
31.pattern.cpp
32.right angle pattern.cpp
1.Two class of employee and manager in polymorphism.cpp
2. two classes Herbivore and Carnivore from the base class in polymorphism.cpp
dent teacher in polymorphism.cpp
C:\Users\bomma\Desktop\c+
Resources
Compile Log
Debug
Find Results
Close
Message

```cpp
#include <iostream>
#include <cmath>

class Shape {
public:
    virtual double area() const = 0;
    virtual double volume() const = 0;
};

class Sphere : public Shape {
private:
    double radius;
public:
    Sphere(double r) : radius(r) {}

    double area() const override {
        return 4 * M_PI * radius * radius;
    }

    double volume() const override {
        return (4.0 / 3.0) * M_PI * radius * radius * radius;
    }
};

class Cylinder : public Shape {
private:
    double radius;
    double height;
public:
    Cylinder(double r, double h) : radius(r), height(h) {}

    double area() const override {
        return 2 * M_PI * radius * (radius + height);
    }
```

```
Sphere Area: 314.159
Sphere Volume: 523.599
Cylinder Area: 471.239
Cylinder Volume: 785.398

--------------------------------
Process exited after 0.7817 seconds with return valu
e 0
Press any key to continue . . .
```

4.Create a base class called Person with a virtual function greet). Derive two classes Student and Teacher from the base class. implement the greet) function for each class



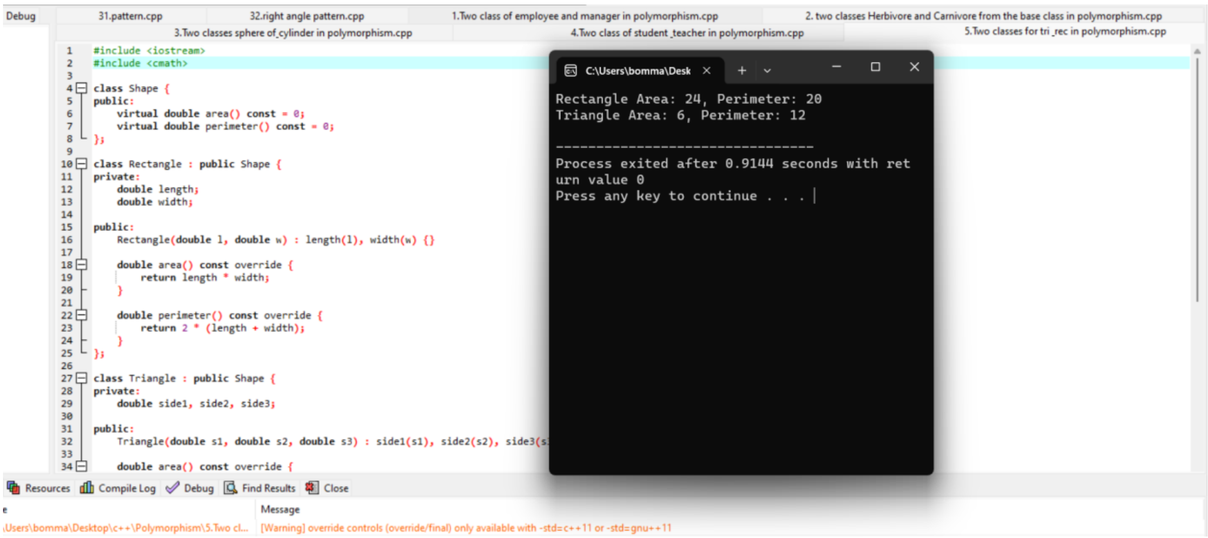
bals)
bug
31.pattern.cpp
2. two classes Herbivore and Carnivore from the base class in polymorph
ger in polymorphism.cpp
lass of student teacher in polymorphism.cpp
C:\Users\bomma\Desktop\c+
Resources
Compile Log
Debug
Find Results
Close
Message

```cpp
#include <iostream>
using namespace std;

class Person {
public:
    virtual void greet() const {
        cout << "Hello, I am a person.\n";
    }
};

class Student : public Person {
public:
    void greet() const override {
        cout << "Hello, I am a student.\n";
    }
};

class Teacher : public Person {
public:
    void greet() const override {
        cout << "Hello, I am a teacher.\n";
    }
};

int main() {
    Person* personPtr1 = new Person();
    Person* studentPtr = new Student();
    Person* teacherPtr = new Teacher();

    personPtr1->greet();
    studentPtr->greet();
    teacherPtr->greet();
    delete personPtr1;
    delete studentPtr;
```

```
Hello, I am a person.
Hello, I am a student.
Hello, I am a teacher.

--------------------------------
Process exited after 0.9965 seconds with return value 0
Press any key to continue . . .
```

```
rs\bomma\Desktop\c++\Polymorphism\4.Two cl...    [Warning] override controls (override/final) only available with -std=c++11 or -std=gnu++11
```

5.Create a base class called Shape with virtual functions area( ) and perimeter(). Derive two classes Rectangle and Triangle from the base class. Implement the area () and perimeter () functions for each class.

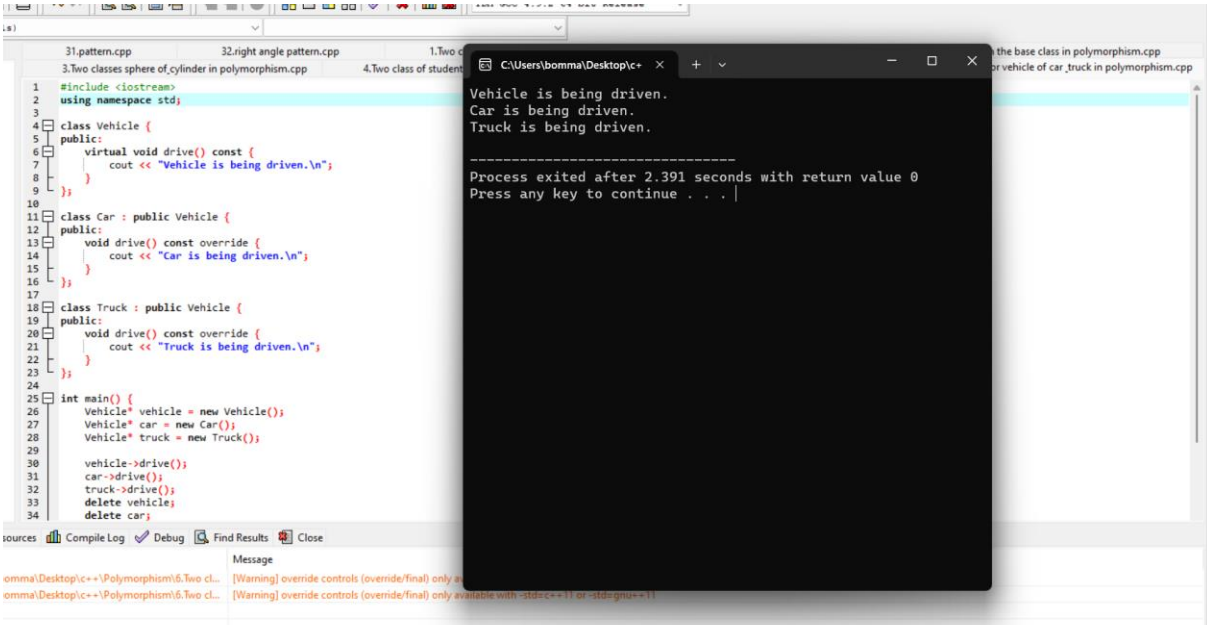

Debug
31.pattern.cpp
32.right angle pattern.cpp
1.Two class of employee and manager in polymorphism.cpp
2. two classes Herbivore and Carnivore from the base class in polymorphism.cpp
3.Two classes sphere of cylinder in polymorphism.cpp
4.Two class of student teacher in polymorphism.cpp
5.Two classes for tri_rec in polymorphism.cpp
C:\Users\bomma\Desk
Resources
Compile Log
Debug
Find Results
Close
Message

```
#include <iostream>
#include <cmath>

class Shape {
public:
    virtual double area() const = 0;
    virtual double perimeter() const = 0;
};

class Rectangle : public Shape {
private:
    double length;
    double width;

public:
    Rectangle(double l, double w) : length(l), width(w) {}

    double area() const override {
        return length * width;
    }

    double perimeter() const override {
        return 2 * (length + width);
    }
};

class Triangle : public Shape {
private:
    double side1, side2, side3;

public:
    Triangle(double s1, double s2, double s3) : side1(s1), side2(s2), side3(s

    double area() const override {
```

```
Rectangle Area: 24, Perimeter: 20
Triangle Area: 6, Perimeter: 12

--------------------------------
Process exited after 0.9144 seconds with ret
urn value 0
Press any key to continue . . .
```

```
\Users\bomma\Desktop\c++\Polymorphism\5.Two cl...   [Warning] override controls (override/final) only available with -std=c++11 or -std=gnu++11
```

6.Create a base class called Vehicle with a virtual function drive(). Derive two classes Car and Truck from the base class. Implement the drive() function for each class.

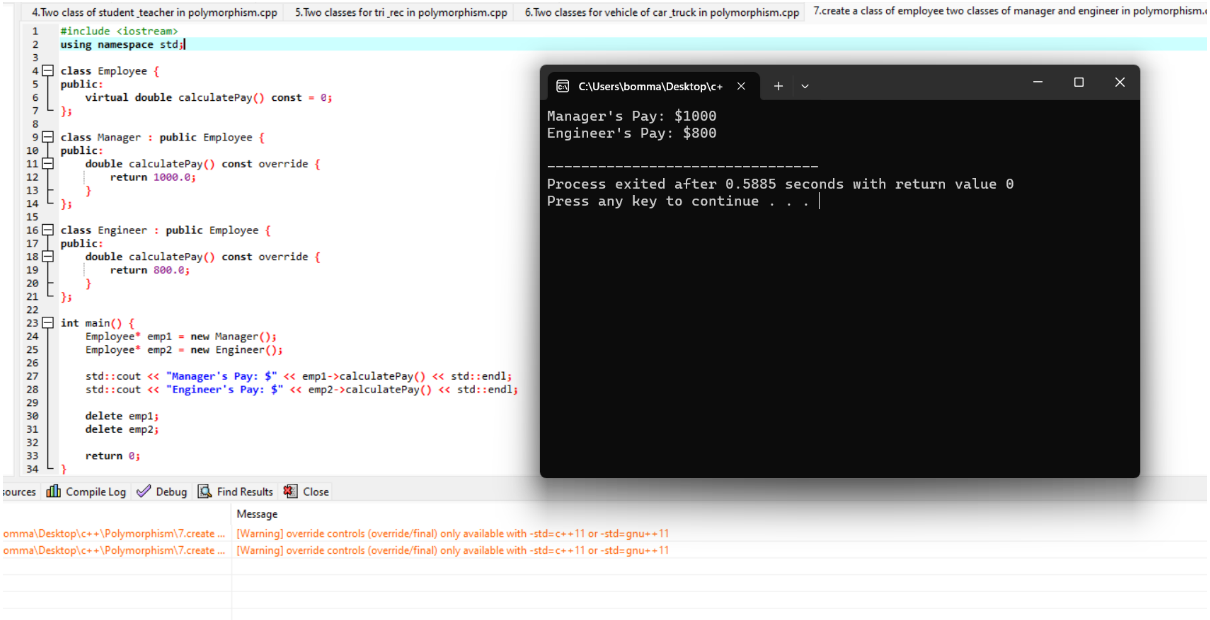

31.pattern.cpp
32.right angle pattern.cpp
1.Two c
the base class in polymorphism.cpp
3.Two classes sphere of cylinder in polymorphism.cpp
4.Two class of student
r vehicle of car_truck in polymorphism.cpp
C:\Users\bomma\Desktop\c+
sources
Compile Log
Debug
Find Results
Close
Message

```
#include <iostream>
using namespace std;

class Vehicle {
public:
    virtual void drive() const {
        cout << "Vehicle is being driven.\n";
    }
};

class Car : public Vehicle {
public:
    void drive() const override {
        cout << "Car is being driven.\n";
    }
};

class Truck : public Vehicle {
public:
    void drive() const override {
        cout << "Truck is being driven.\n";
    }
};

int main() {
    Vehicle* vehicle = new Vehicle();
    Vehicle* car = new Car();
    Vehicle* truck = new Truck();

    vehicle->drive();
    car->drive();
    truck->drive();
    delete vehicle;
    delete car;
```

```
Vehicle is being driven.
Car is being driven.
Truck is being driven.

--------------------------------
Process exited after 2.391 seconds with return value 0
Press any key to continue . . .
```

```
omma\Desktop\c++\Polymorphism\6.Two cl...   [Warning] override controls (override/final) only a
omma\Desktop\c++\Polymorphism\6.Two cl...   [Warning] override controls (override/final) only available with -std=c++11 or -std=gnu++11
```

7.Create a base class called Employee with a virtual function calculate Pay(). Derive two classes Manager and Engineer from the base class. Implement the calculatePay () function for each class.

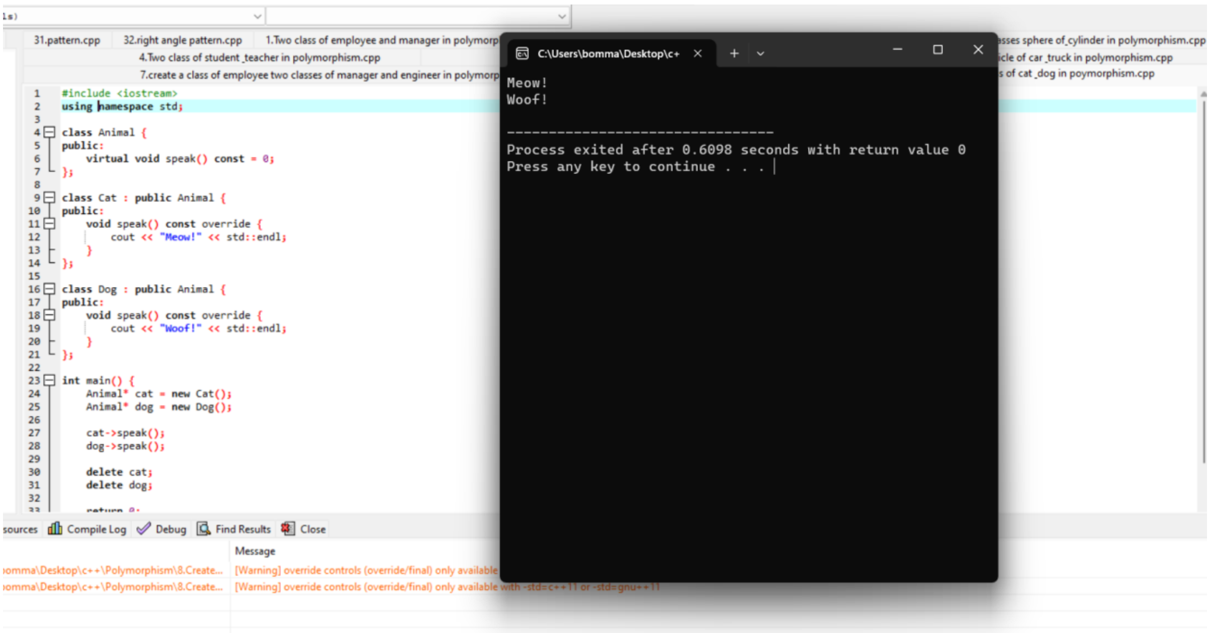

4.Two class of student_teacher in polymorphism.cpp
5.Two classes for tri_rec in polymorphism.cpp
6.Two classes for vehicle of car_truck in polymorphism.cpp
7.create a class of employee two classes of manager and engineer in polymorphism.
C:\Users\bomma\Desktop\c+
sources
Compile Log
Debug
Find Results
Close
Message

```cpp
#include <iostream>
using namespace std;

class Employee {
public:
    virtual double calculatePay() const = 0;
};

class Manager : public Employee {
public:
    double calculatePay() const override {
        return 1000.0;
    }
};

class Engineer : public Employee {
public:
    double calculatePay() const override {
        return 800.0;
    }
};

int main() {
    Employee* emp1 = new Manager();
    Employee* emp2 = new Engineer();

    std::cout << "Manager's Pay: $" << emp1->calculatePay() << std::endl;
    std::cout << "Engineer's Pay: $" << emp2->calculatePay() << std::endl;

    delete emp1;
    delete emp2;

    return 0;
}
```

```
Manager's Pay: $1000
Engineer's Pay: $800

--------------------------------
Process exited after 0.5885 seconds with return value 0
Press any key to continue . . .
```

```
omma\Desktop\c++\Polymorphism\7.create ...   [Warning] override controls (override/final) only available with -std=c++11 or -std=gnu++11
omma\Desktop\c++\Polymorphism\7.create ...   [Warning] override controls (override/final) only available with -std=c++11 or -std=gnu++11
```

8.Create a base class called Animal with a virtual function speak(). Derive two classes Cat and Dog from the base class. Implement the speak() function for each class.

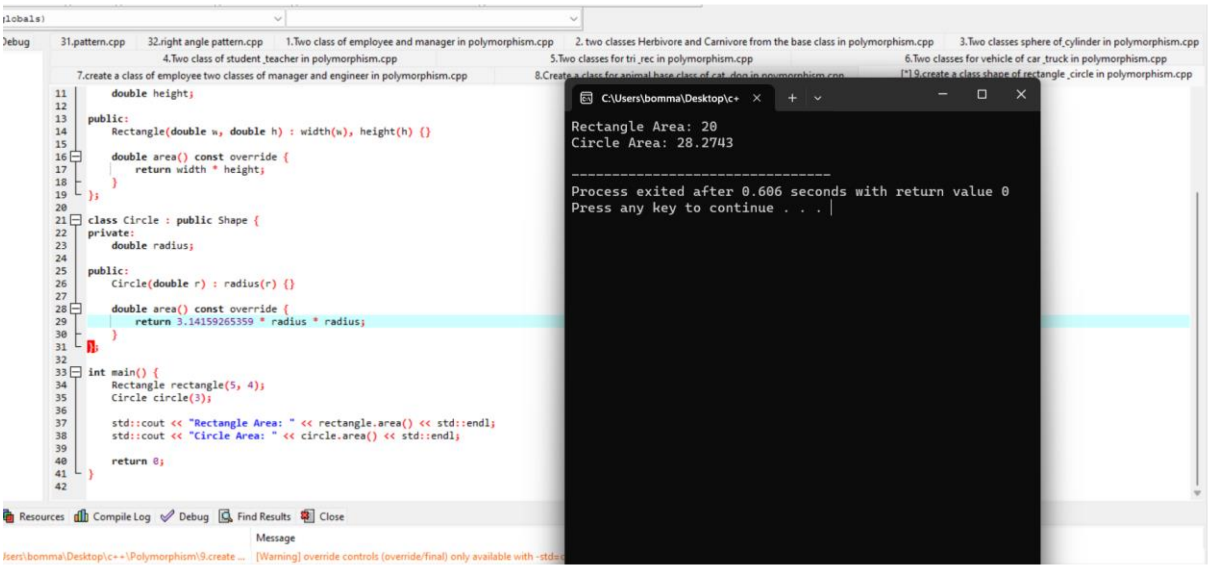

31.pattern.cpp
32.right angle pattern.cpp
1.Two class of employee and manager in polymorp
4.Two class of student_teacher in polymorphism.cpp
7.create a class of employee two classes of manager and engineer in polymorp
sses sphere of cylinder in polymorphism.cpp
cle of car_truck in polymorphism.cpp
s of cat_dog in poymorphism.cpp
C:\Users\bomma\Desktop\c+
sources
Compile Log
Debug
Find Results
Close
Message

```cpp
#include <iostream>
using namespace std;

class Animal {
public:
    virtual void speak() const = 0;
};

class Cat : public Animal {
public:
    void speak() const override {
        cout << "Meow!" << std::endl;
    }
};

class Dog : public Animal {
public:
    void speak() const override {
        cout << "Woof!" << std::endl;
    }
};

int main() {
    Animal* cat = new Cat();
    Animal* dog = new Dog();

    cat->speak();
    dog->speak();

    delete cat;
    delete dog;

    return 0;
```

```
Meow!
Woof!

--------------------------------
Process exited after 0.6098 seconds with return value 0
Press any key to continue . . .
```

```
omma\Desktop\c++\Polymorphism\8.Create...   [Warning] override controls (override/final) only available
omma\Desktop\c++\Polymorphism\8.Create...   [Warning] override controls (override/final) only available with -std=c++11 or -std=gnu++11
```

10.Create a base class called Shape with a virtual function area(). Derive two classes Rectangle and Circle from the base class. Implement the area() function for each class.


lobals)
Debug
31.pattern.cpp
32.right angle pattern.cpp
1.Two class of employee and manager in polymorphism.cpp
2. two classes Herbivore and Carnivore from the base class in polymorphism.cpp
3.Two classes sphere of cylinder in polymorphism.cpp
4.Two class of student _teacher in polymorphism.cpp
5.Two classes for tri _rec in polymorphism.cpp
6.Two classes for vehicle of car _truck in polymorphism.cpp
7.create a class of employee two classes of manager and engineer in polymorphism.cpp
8.Create a class for animal base class of cat_dog in polymorphism.cpp
[*] 9.create a class shape of rectangle _circle in polymorphism.cpp
C:\Users\bomma\Desktop\c+
Resources
Compile Log
Debug
Find Results
Close
Message

```cpp
    double height;

public:
    Rectangle(double w, double h) : width(w), height(h) {}

    double area() const override {
        return width * height;
    }
};

class Circle : public Shape {
private:
    double radius;

public:
    Circle(double r) : radius(r) {}

    double area() const override {
        return 3.14159265359 * radius * radius;
    }
};

int main() {
    Rectangle rectangle(5, 4);
    Circle circle(3);

    std::cout << "Rectangle Area: " << rectangle.area() << std::endl;
    std::cout << "Circle Area: " << circle.area() << std::endl;

    return 0;
}
```

```
Rectangle Area: 20
Circle Area: 28.2743

--------------------------------
Process exited after 0.606 seconds with return value 0
Press any key to continue . . .
```

```
Jsers\bomma\Desktop\c++\Polymorphism\9.create ...   [Warning] override controls (override/final) only available with -std=c
```