1st output

Area of rectangle is 330

Area of figure is calculating... (is coming from super class)

2nd output

Area of rectangle is 330

Area of Circle (PI \* R \* R)113.03999999999999

3rd Output

Area of figure is calculating... (is coming from super class)

Area of Circle (PI \* R \* R)113.03999999999999

4th output

Area of rectangle is 330

Area of Circle (PI \* R \* R)113.03999999999999

// if you need a method which acts as a contract

// or the rule then make that method as abstract

// if the method is abstract in the class then

// the class should also be made abstract

// if the class is abstract then creating an object

// of the class is not allowed directly

// in a abstract class it is possible to have

// concrete method/s, and this can be access only

// through child class/es

encapsulation

inheritance

polymorphism

abstraction

class Vehicle{

void abstract move();

}

class Car extends Vehicle{

void move(){

Sop("car moves");

}

}

class Truck extends Vehicle{

void move(){

Sop("truck moves");

}

}

class First extends Second{}

class First extends Second extends Thrid {} (invalid)

class First extends Second, Third {} (invalid)

final class myclass{ } // then this class cannot be inherited

class mymclass{

method1(){}

final method2(){}

}

class Yourclass extends myclass{

method1(){}

method2(){}(this line becomes invalid )

}

to declare constants in the program

final int x =33;

\*) class extends class

\*) interface extends interface or interfaces

interface I1 extends I2, I3, I4{}

\*) class implements interface or interfaces

\*) interface neither extends not implements the class

Interfaces are used to show working of multiple inheritance