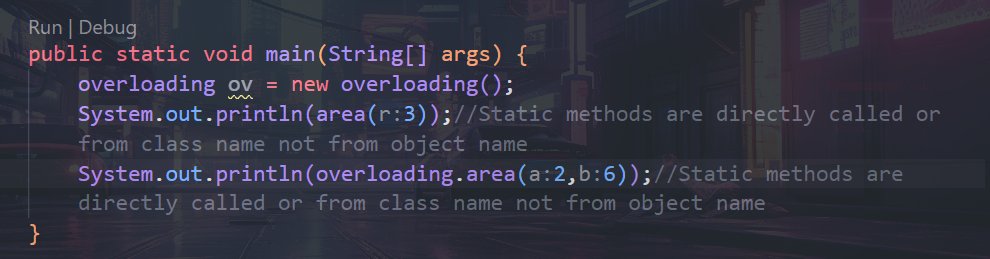
* We cant use super keyword inside static
* Inheritance
* Abstraction

Polymorphism

* Important feature of Object Oriented Programming
* Poly 🡪 Many
* Morphism 🡪 Form
* Hence Many form
* Types:-
  1. Compile time polymorphism
     + Method Overloading
       - Used again and again
       - Method name should be same
       - Method return type, No of Arguments and types may or may not be different
* Static methods are directly called or from class name not from object name



* + - Method Overriding
  1. Run Time Polymorphism

Method Overloading:-

class overloading {

    static float area(int r){

        return 3.14f\*r\*r;

    }

    static int area(int a,int b){

        return a\*b;

    }

    public static void main(String[] args) {

        overloading ov = new overloading();

        System.out.println(area(3));//Static methods are directly called or from class name not from object name

        System.out.println(overloading.area(2,6));//Static methods are directly called or from class name not from object name

    }

}

Method Overriding

* Method name should be same.
* Method return type, No or Arguments and types of arguments should be same
* It works in Inheritance.
* Jiska object uske methods ko call.
* class overridedemo{
* void draw(int m){ //no of arguments, return type of method and type of arguments should be same
* System.out.println("Base Class draw method");
* }
* }
* class overridding extends overridedemo {//no of arguments, return type of method and type of arguments should be same
* void draw(int m){
* System.out.println("Child Class draw method");
* }


* public static void main(String[] args) {
* overridding ov = new overridding();
* ov.draw(0);
* }
* }

Final Keyword

* Final is keyword in java. When final is used within variable, method and class.
* When final is used within variable then it cannot be modified.
* When final is used within method then method cannot be overridden
* When final is used with class then class cannot be inherited.