Homework Hustlers: https://discord.gg/aJ55rZBV

- Wizard.

How is code reuse achieved through inheritance?

It is because the parent properties are inherited by the child's so that code is resued.

What are the types of inheritance in Java? Why does Java not support multiple inheritance?

- Single inheritance
- Multiple inheritance
- Hierarchical inheritance
- Multi Level inheritance with interfaces
- Hybrid aint supported

Define a class Person with attributes name and age.

```
~

class Person {
    public String name;
    public int age;

}
class tuto {
    public static void main(String[] args) {
        Person ob1 = new Person();
    }
}
```

Create a subclass Employee that inherits from Person and adds an attribute employeeId.

```
~

class Person {
    public String name;
    public int age;

}

class Employee extends Person {
    public String employeeid;

}
class tuto {
    public static void main(String[] args) {
        Person ob1 = new Person();
    }
}
```

Define a class Animal with a method makeSound().

```
~

class Animal {
    public void makeSound(){
        System.out.println("Sound made.");
    }
}

class tuto {
    public static void main(String[] args) {
        Animal an1 = new Animal();
        an1.makeSound();
    }
}
```

Create subclasses Dog and Cat that inherit from Animal and override the makeSound() method with barking and meowing, respectively

```
~

class Animal {
    public void makeSound(){
        System.out.println("Sound made.");
    }
}

class Dog {
    public void makeSound(){
        System.out.println("woof");
    }

}

class Cat {
    public void makeSound(){
        System.out.println("Meow");
    }

}
class tuto {
    public static void main(String[] args) {
        Animal an1 = new Animal();
        an1.makeSound();
        Dog n1 = new Dog();
        n1.makeSound();
        Cat n2 = new Cat();
        n2.makeSound();
    }
}
```

```
~

[wizard@archlinux 2]$ java tuto
Sound made.
woof
Meow
```

Create a class A with a method display() that prints "Class A"

```
~

class A {
    public void display(){
        System.out.println("Class A");
    }


}

class tuto {
    public static void main(String[] args) {
        A newA = new A();
    }
}
```

Create a subclass B that inherits from A and overrides display() to print "Class B".

```
~

class A {
    public void display(){
        System.out.println("Class A");
    }


}

class B extends A{
    public void display(){
        System.out.println("Class B");
    }


}
class tuto {
    public static void main(String[] args) {
        B newB = new B();
        newB.display();
    }
}
```

```
~

[wizard@archlinux 2]$ java tuto
Class B
[wizard@archlinux 2]$
```

Create another subclass C that inherits from B and overrides display() to print "Class C"

```
~

class A {
    public void display(){
        System.out.println("Class A");
    }

}

class B extends A {
    public void display(){
        System.out.println("Class B");
    }

}
class C extends B {
    public void display(){
        System.out.println("Class C");
    }

}
class tuto {
    public static void main(String[] args) {
        C newC = new C();
        newC.display();
    }
}
```

```
~

[wizard@archlinux 2]$ java tuto
Class C
[wizard@archlinux 2]$
```

Create an object of class C and call its display() method

```
~

class A {
    public void display(){
        System.out.println("Class A");
    }

}

class B extends A {
    public void display(){
        System.out.println("Class B");
    }

}
class C extends B {
    public void display(){
        System.out.println("Class C");
    }

}

class tuto {
    public static void main(String[] args) {
        C newC = new C();
        newC.display();
    }
}
```

```
~

[wizard@archlinux 2]$ java tuto
Class C
[wizard@archlinux 2]$
```

Implement a class Shape with attributes color and method area().

```
~



class Shape {
    public String color;
    public void Area(){
        System.out.println("This Prints area");
    }
}
class tuto {
    public static void main(String[] args) {

    }
}
```

Create a subclass Rectangle that inherits from Shape and adds attributes length and width.

```
~

class Shape {
    public String color;
    public void Area(){
        System.out.println("This Prints area");
    }
}

class Rectangle extends Shape{
    public int length, width;
}

class tuto {
    public static void main(String[] args) {

    }
}
```

Use super() to initialize the color attribute in the Rectangle constructor.

```
~

class Shape {
    public String color;
    public void Area(){
        System.out.println("This Prints area");
    }

    public Shape(String color){
        this.color = color;
    }
}

class Rectangle extends Shape{
    public int length, width;

    public Rectangle(String color) {
        super(color);
    }
}

class tuto {
    public static void main(String[] args) {
        Rectangle obj1 = new Rectangle("red");
        System.out.println(obj1.color);

    }
}
```

```
~

[wizard@archlinux 2]$ javac tuto.java
[wizard@archlinux 2]$ java tuto
red
[wizard@archlinux 2]$
```

Create an overloaded method named calculateArea to compute and display the area of a rectangle using length and width.

```
~

class AreaCalculator {
    public AreaCalculator() {
        System.out.println("This prints area with no param");
    }
    public AreaCalculator(int length , int breadth) {
        System.out.println(String.format("Area of rectangle with %d length %d breadth is %d",length,breadth,length*breadth ));
    }
}
class tuto {
    public static void main(String[] args) {
        AreaCalculator obj1 = new AreaCalculator();
        AreaCalculator obj2 = new AreaCalculator(1,2);


    }
}
```

```
~

[wizard@archlinux 2]$ java tuto
This prints area with no param
Area of rectangle with 1 length 2 breadth is 2
[wizard@archlinux 2]$
```

Overload the calculateArea method to calculate and display the area of a square using the side length

```
~

class AreaCalculator {
    public AreaCalculator() {
        System.out.println("This prints area with no param");
    }

    public AreaCalculator(int length , int breadth) {
        System.out.println(String.format("Area of rectangle with %d length %d breadth is %d",length,breadth,length*breadth ));
    }

    public AreaCalculator(int length ) {
        System.out.println(String.format("Area of square with %d length is %d",length,length*length));
    }
}
class tuto {
    public static void main(String[] args) {
        AreaCalculator obj1 = new AreaCalculator();
        AreaCalculator obj2 = new AreaCalculator(1,2);
        AreaCalculator obj3 = new AreaCalculator(2);


    }
}
```

```
~

[wizard@archlinux 2]$ java tuto
This prints area with no param
Area of rectangle with 1 length 2 breadth is 2
Area of square with 2 length is 4
[wizard@archlinux 2]$
```

Overload the calculateArea method again to calculate and display the area of a triangle using base and height

```
~

class AreaCalculator {
    public AreaCalculator() {
        System.out.println("This prints area with no param");
    }

    public AreaCalculator(int length , int breadth) {
        System.out.println(String.format("Area of rectangle with %d length %d breadth is 
%d",length,breadth,length*breadth ));
    }

    public AreaCalculator(int length ) {
        System.out.println(String.format("Area of square with %d length is %d",length,length*length));
    }

    public AreaCalculator(double base, double height) {
        System.out.println(String.format("Area of triangle with %f base , %f height is 
%f",base,height,0.5*base*height));
    }
}
class tuto {
    public static void main(String[] args) {
        AreaCalculator obj1 = new AreaCalculator();
        AreaCalculator obj2 = new AreaCalculator(1,2);
        AreaCalculator obj3 = new AreaCalculator(2);
        AreaCalculator obj4 = new AreaCalculator(2.0,3.0);



    }
}
```

```
~

[wizard@archlinux 2]$ java tuto
This prints area with no param
Area of rectangle with 1 length 2 breadth is 2
Area of square with 2 length is 4
Area of triangle with 2.000000 base , 3.000000 height is 3.000000
[wizard@archlinux 2]$
```

Define a class Animal as a final class. Try creating a subclass Dog that inherits from class Animal and observe the result.

```
~

final class Animal {
}

class Dog extends Animal{}
class tuto {
    public static void main(String[] args) {
    }
}
```

```
~

[wizard@archlinux 2]$ javac tuto.java
tuto.java:4: error: cannot inherit from final Animal
class Dog extends Animal{}
                  ^
1 error
[wizard@archlinux 2]$
```

Define a class Vehicle as final with a method start(). Attempt to create a subclass Car that inherits from Vehicle and observe the result.

```
~

final class Vehicle {
}

class Car extends Vehicle{}
class tuto {
    public static void main(String[] args) {
    }
}
```

```
~

[wizard@archlinux 2]$ javac tuto.java
tuto.java:4: error: cannot inherit from final Vehicle
class Car extends Vehicle{}
                  ^
1 error
[wizard@archlinux 2]$
```