**Course 3 – Sprint 5 – Assessment 1 – Predict the output**

# Predict the output of the following code: \*



Value of number: 50

Child class number 50

Child class second number 100

Value of number: 20

Child class number 50

1. **Which of the following statements are false for method overriding in java?** \*

*Mark only one oval.*

### This is also called dynamic polymorphism or runtime polymorphism. Overriding occurs between the methods in the same class.

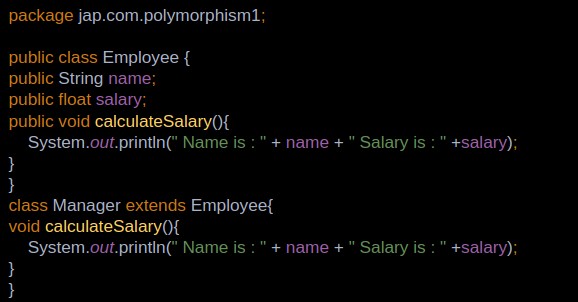
Overriding occurs between the superclass and subclass.

### Overriding methods have the same signature, i.e., the same name and method arguments.

In method overriding, the method to call is determined at runtime based on the object type.

# Fix the compile time error in the code and also explain why the error is \*

**coming?**



class Manager extends Employee

{

**public** void calculateSalary() // Cannot reduce the visibility of the inherited method from base class

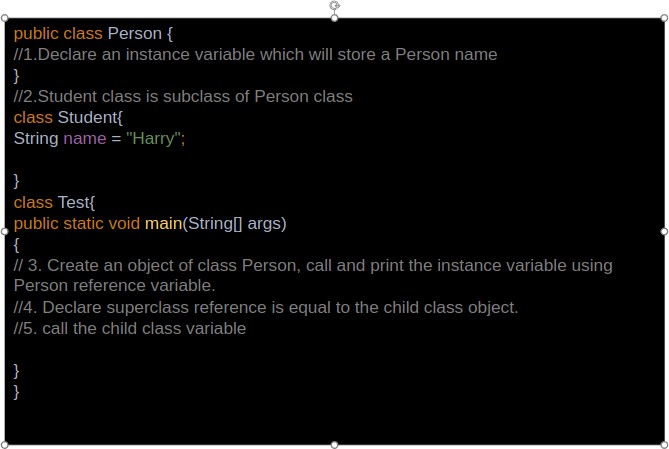
{

System.out.println(" Name is : " + name + " Salary is : ");

}

}

# Write the code for the comments provided and predict the output as well \*



public class Person

{

String name = "John";

}

class Student extends Person

{

String name = "Harry";

}

class Test

{

public static void main(String[] args)

{

Person obj1 = new Person();

System.out.println(obj1.name);

Person obj2 = new Student();

System.out.println(obj2.name);

}

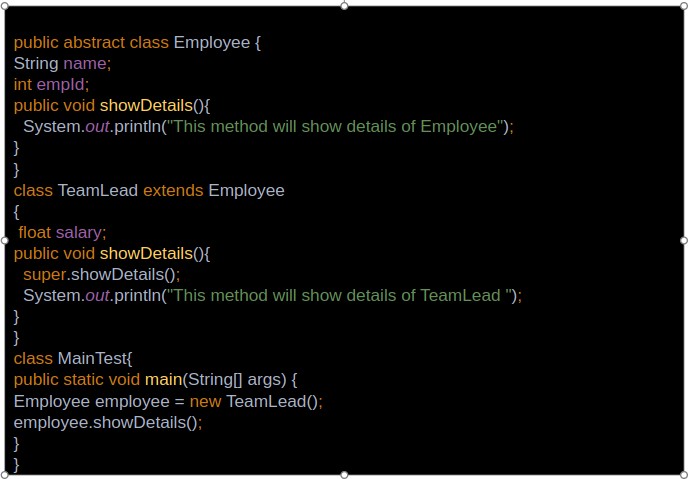
}

**Output:**

John

John

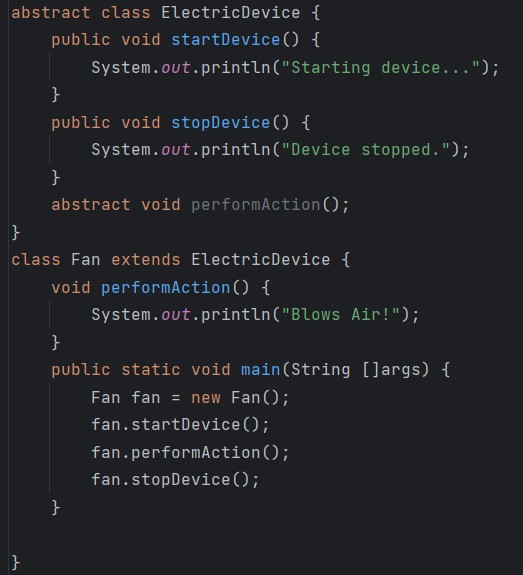
1. **Predict the output for the following code** \*



This method will show details of Employee

This method will show details of TeamLead

# Predict the output of the below code.

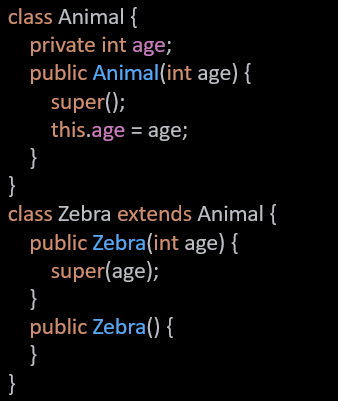


Starting device…

Blows Air!

Device stopped.

1. **Fix the compile time errors in the below code**



class Animal {

private int age;

public Animal(int age) {

super();

this.age = age;

}

}

class Zebra extends Animal {

public Zebra(int age) {

super(age);

}

*// Zebra cannot have a default constructor as the parent class doesn’t have a default constructor*

*//* ***Or*** *it needs to pass in a constant literal value for age to the super class constructor in its default constructor*

*//* ***Or*** *Create a default constructor in the base class.*

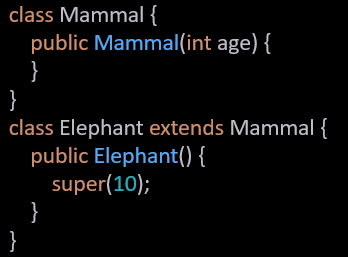
public Zebra() {

super(15);

}

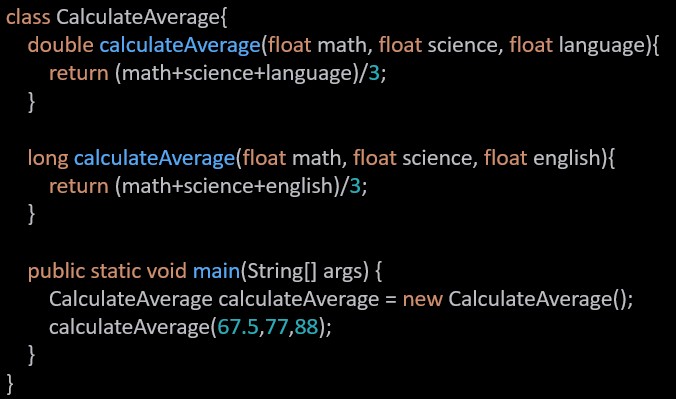
}

# Will the below code compile without errors ? Give reasons



Yes, this will compile as we are passing in a value for the age parameter to the parameterized constructor of the super class Mammal.

1. **Predict the output of the below code.**



The above code will not compile due to the following reasons:

1. calculateAverage(float, float, float) defined twice in the class CalculateAverage. Return type doesn’t lead to overloading.
2. Second method long calculateAverage doesn’t return long
3. Call in main calculateAverage(67.5, 77, 88) will not be resolved to any of the two methods due to the parameters being of type double, int , int

# Predict the output of the below code. Give reasons



Voter{name=’Jane’,age=0}

Voter{name=’David’,age=35}

1. **Following code is giving compile time error, fix the code and also explain why it is giving compile time error**



public class Person {

public String name;

private int age;

public Person(String name, int age) {

this.name = name;

this.age = age;

}

public void setName(String name) {

this.name = name;

}

public void setAge(int age) *{// private function needs to be public*

this.age = age;

}

public void printDetails() {

System.out.println("Name:" + name);

System.out.println("Age: " + age);

}

}

public class Main {

public static void main(String[] args) {

Person person = new Person("John", 25);

person.setName("Sarah");

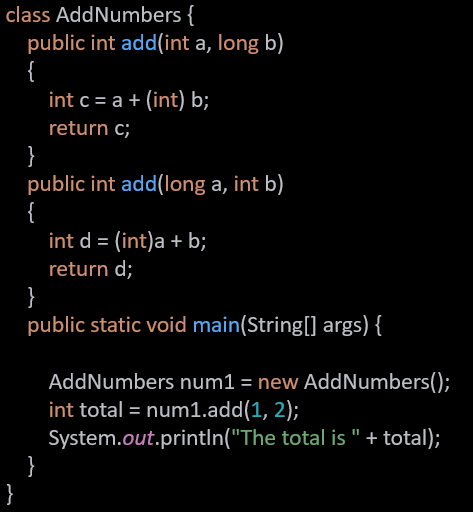
person.setAge(30);

person.printDetails();

}

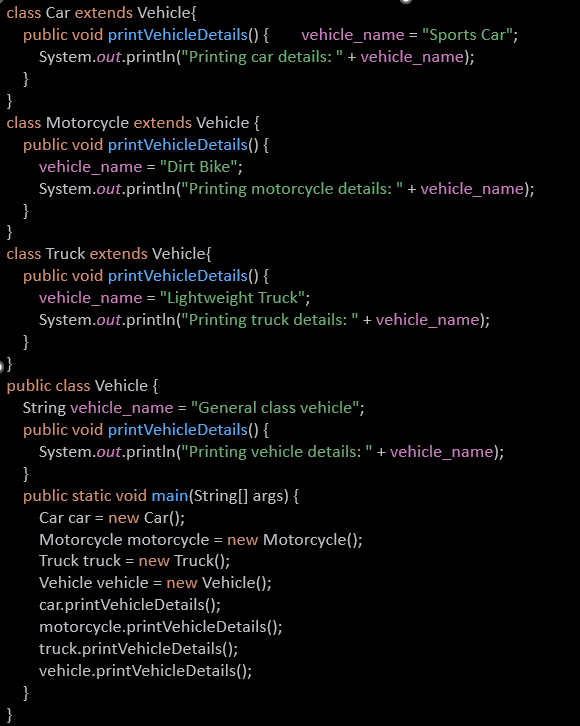
}

1. **Predict the output of the below code. Give reasons**



It will not compile because of an ambiguous method call in num1.add(1, 2);

1. **Predict the output of the below code.**



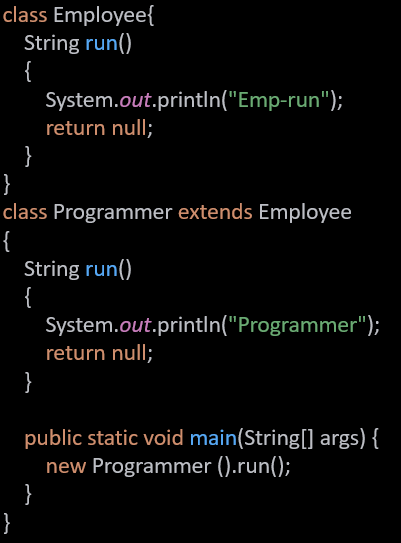
Printing car details: Sports Car

Printing motorcycle details: Dirt Bike

Printing truck details: Lightweight Truck

Printing vehicle details: General class vehicle

1. **Predict the output of the below code. Give reasons**



Programmer

The function of class Programmer will be invoked in the call new Programmer().run()

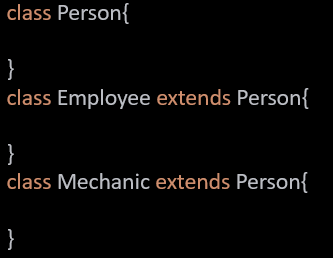
1. **Choose the correct option and give reasons**

## The code exhibits polymorphism with classes

* 1. The code exhibits polymorphism with methods

## The code does not exhibit polymorphism

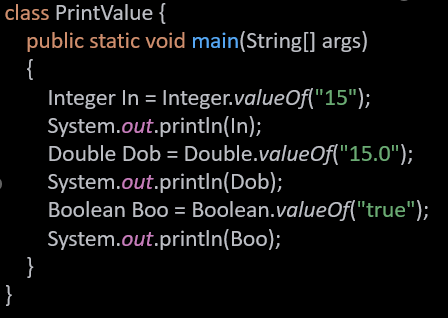
* 1. The code exhibits polymorphism with classes and interfaces



c. The code does not exhibit polymorphism

The code exhibits inheritance but not polymorphism.

1. **Predict the output of the below code.**

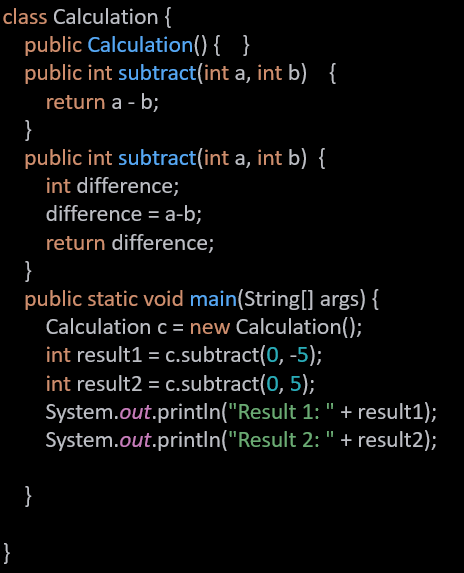


15

15.0

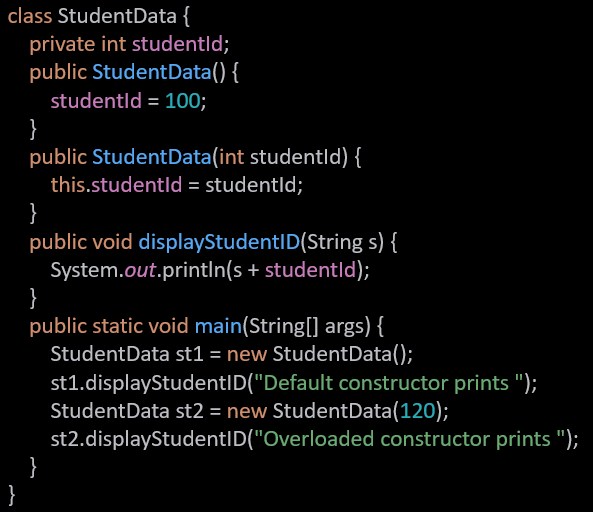
true

1. **Predict the output of the below code.**



Compilation error due to subtract(int,int) being defined twice in the class Calculation.

1. **Predict the output of the below code**

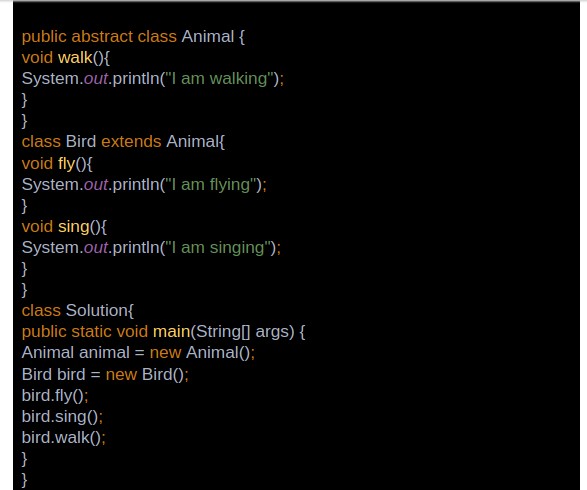


Default constructor prints 100

Overloaded constructor prints 120

# Following code is giving compile time error, fix the code and also explain \*

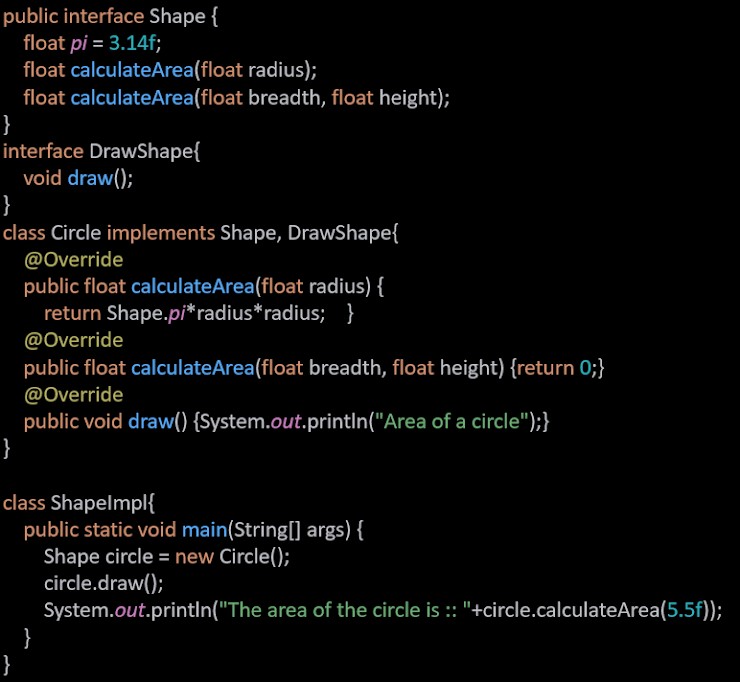
**why it is giving compile time error**



Line *Animal animal = new Animal();* will not compile as we are trying to instantiate an object of an abstract class.

Removing the line will fix the code

# Identify and fix the error in the below code



There are 3 solutions possible  
1. Either We need to create the circle object in main of type Circle to be able to call both draw and calculateArea methods

2. Or we should combine the 2 interfaces into one Shape interface and the class Circle implements only the Shape interface

**3.(Best)**cast circle object to type circle before calling draw().