Question 1 : Functional Interface with default, static, non-static & abstract methods

@FunctionalInterface

interface FI {

    abstract int cube(int n);

    default int square(int n) {

        return n \* n;

    }

    default int add(int a, int b) {

        return a + b;

    }

    static int sub(int a, int b) {

        return a - b;

    }

    static int mul(int a, int b) {

        return a \* b;

    }

    static double div(int a, int b) {

        return a / b;

    }

}

public class FuncInterAssignment {

    public static void main(String[] args) {

        FI obj = (int n) -> n \* n \* n;

        System.out.println("cube = " + obj.cube(5));

        System.out.println("square = " + obj.square(6));

        System.out.println("add = " + obj.add(2, 4));

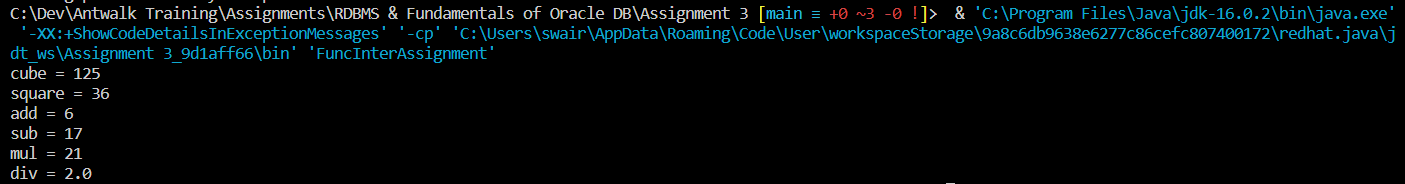
        System.out.println("sub = " + FI.sub(23, 6));

        System.out.println("mul = " + FI.mul(3, 7));

        System.out.println("div = " + FI.div(4, 2));

    }

}

Output :

Question 2 : Constructor Method Reference

Person Class:

public class Person {

    private String name;

    private int age;

    public Person(String name, int age) {

        super();

        this.name = name;

        this.age = age;

    }

    public String getName() {

        return name;

    }

    public void setName(String name) {

        this.name = name;

    }

    public Integer getAge() {

        return age;

    }

    public void setAge(int age) {

        this.age = age;

    }

    @Override

    public String toString() {

        return "Person [name=" + name + ", age=" + age + "]";

    }

}

Tester Class:

@FunctionalInterface

interface MethodRefInterface {

    Person getPerson(String name, int age);

}

public class TesterMethodRefAssignment {

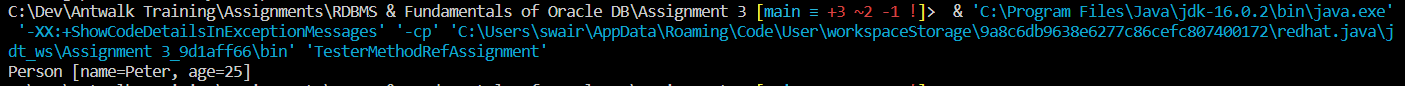
    public static void main(String[] args) {

        MethodRefInterface obj = Person::new;

        System.out.println(obj.getPerson("Peter", 25));

    }

}

Output :