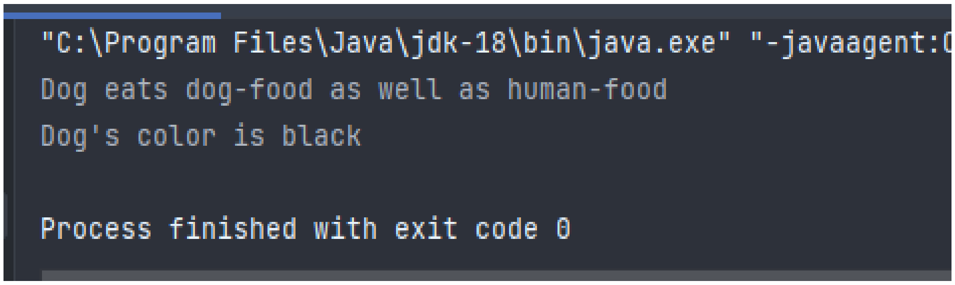
Name : Hardik Patel
Std Id : 202103032

1.WAP in Java to create a class Animal with some method, ex - eat(). Extend the Animal class to more specialized class say for example Dog. Demonstrate the concept of method overriding with the help of classes and functions defined by you.

Code :

```
//Function overriding means when the Dog class contains the
method which is already present
// in the Animal class. Hence, the Dog class overrides the
method of the Animal class.
// In overriding : Animal and Dog classes both contain the same
function
// with a different definition.
class Animal{
    String colour;
    public void eat(){
        System.out.println("This is an Animal");
    }
}
class Dog extends Animal{
    @Override
    public void eat() {
        System.out.println("Dog eats dog-food as well as human-
food");
        System.out.println("Dog's color is " + this.colour);
    }
}
public class Exercise1 {
    public static void main(String[] args) {
        Dog d1 = new Dog();
        d1.colour = "black";
        d1.eat();
    }
}
```

Output :



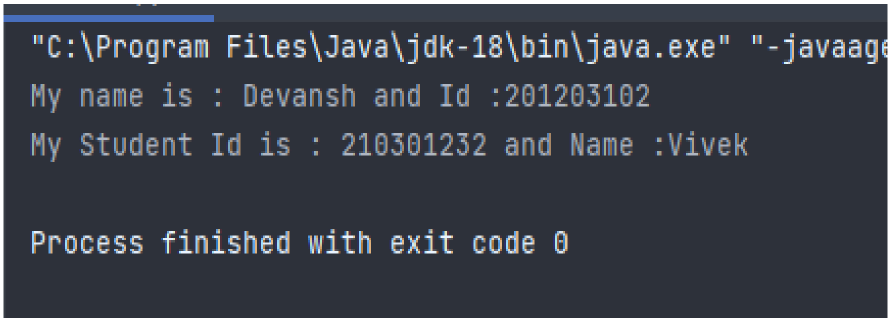
```
"C:\Program Files\Java\jdk-18\bin\java.exe" "-javaagent:C
Dog eats dog-food as well as human-food
Dog's color is black

Process finished with exit code 0
```

2. WAP in Java that overloads a method, for example, display() for the same number of arguments.

Code :

```
// Method overloading is allows us to use
// same function with different functionality.
//implement : The return type of the overloaded function is
different.
// The type of the parameters passed to the function are
different.
//The number of parameters passed to the function are different
class Student{
    String name;
    int StdId;
    void Display(String name, int StdId){
        System.out.println("My name is : " + name + " and Id :"
+ StdId);
    }
    void Display(int StdId, String name){
        System.out.println("My Student Id is : " + StdId + "
and Name :" + name);
    }
}
public class Exercise2 {
    public static void main(String[] args) {
        Student s1 = new Student();
        s1.name = "Devansh";
        s1.StdId = 201203102;
        s1.Display(s1.name, s1.StdId);
        Student s2 = new Student();
        s2.name = "Vivek";
        s2.StdId = 210301232;
        s2.Display(s2.StdId, s2.name);
    }
}
```

Output :



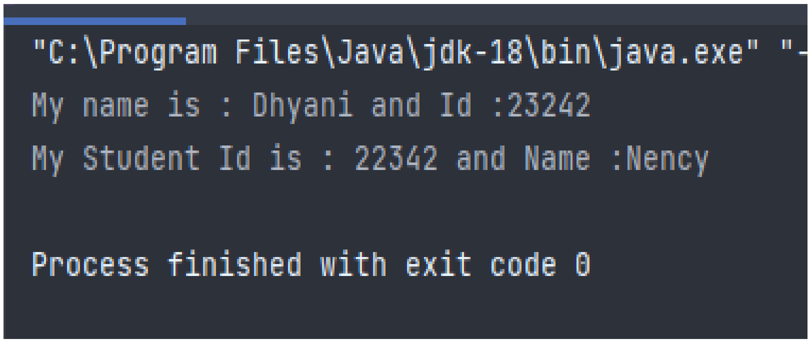
```
"C:\Program Files\Java\jdk-18\bin\java.exe" "-javaage
My name is : Devansh and Id :201203102
My Student Id is : 210301232 and Name :Vivek

Process finished with exit code 0
```

3. WAP in Java that overloads a method, for example, display() for the different number of arguments.

Code :

```
class Data{
    String name;
    int StdId;
    void Display(){
        System.out.println("My name is : " + this.name + " and Id :" + this.StdId);
    }
    void Display(String name){
        System.out.println("My Student Id is : " + this.StdId + " and Name :" + name);
    }
}
public class Exercise3 {
    public static void main(String[] args) {
        Data s1 = new Data();
        s1.name = "Dhyani";
        s1.StdId = 23242;
        s1.Display();
        Data s2 = new Data();
        s2.name = "Nency";
        s2.StdId = 22342;
        s2.Display(s2.name);
    }
}
```

Output :



```
"C:\Program Files\Java\jdk-18\bin\java.exe" "-
My name is : Dhyani and Id :23242
My Student Id is : 22342 and Name :Nency

Process finished with exit code 0
```

4. which is a valid method overloading code snippet:

a.
```
public double area (double length, double width)
{
return length*width;

}
public double area_cir (double radius)
{
return 2*3.14*radius*radius;
}
```

b.
```
public double circumference (double l, double b)
{
return 2*(l+b);
}
public double circumference (double r)
{
return 2*3.14*r;
}
```

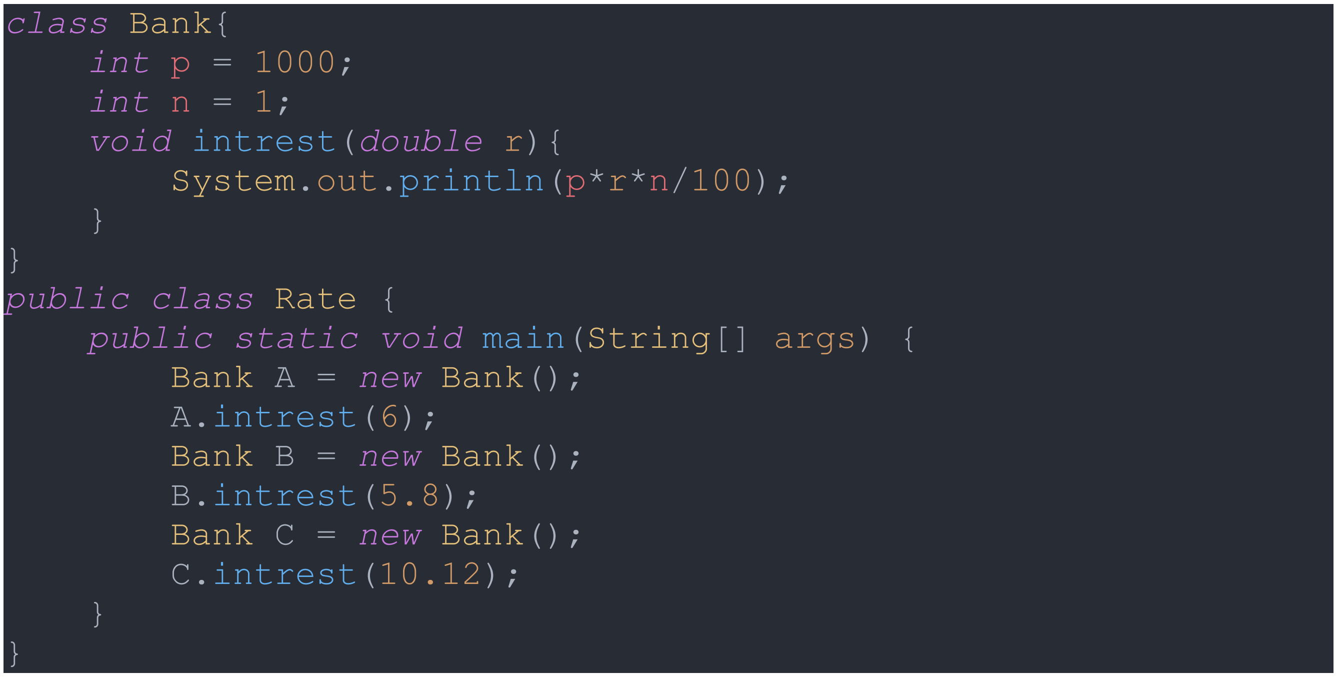
Ans : Option (b) is the valid method overloading code snippet because in option (a) name of the functions are different So,it's not overloding

5. Are the given snippets of code represent valid constructor overloading?
Justify.

a.
```
class A {
A() {};
A(int a) {
System.out.println(a);
}
A(String b) {
System.out.println(b);
}
}
```
b.
```
class JT {
JT() {};
JT (double a) {
System.out.println(a);
}
JT (double b, double c) {
System.out.println(b+c);
}
}
```
Ans : Yes, Both are valid constructor overloading because in first constructor there is no argument ,in second and third constructor argument numbers are same but there data types are different.

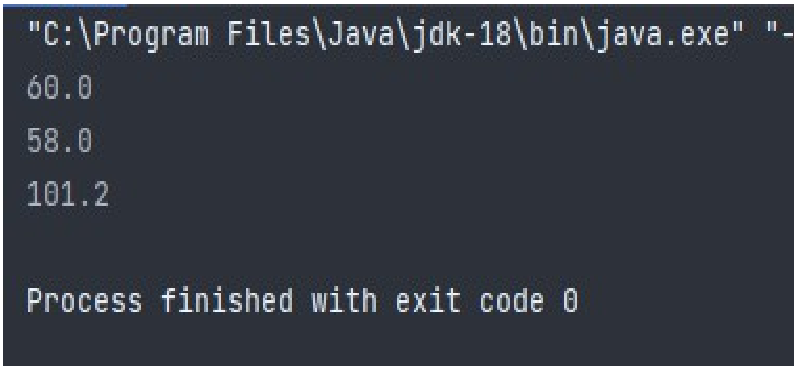
6. WAP in JAVA where consider a scenario where a Bank is a class that provides a method to get the rate of interest. However, the rate of interest may differ according to banks. For example, Bank A, Bank B, and Bank C are providing 6%, 5.8%, and 10.12% rate of interest respectively.

Code :

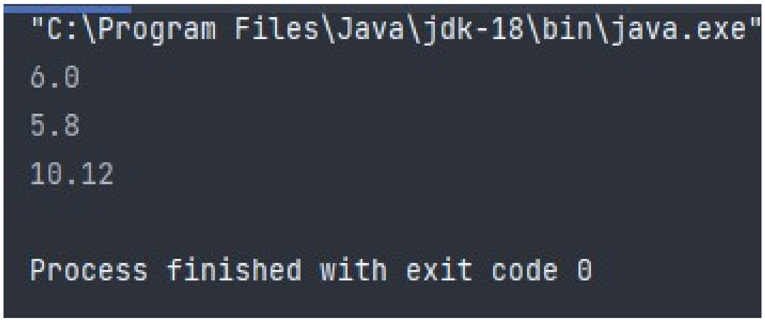


```java
class Bank{
    int p = 1000;
    int n = 1;
    void intrest(double r){
        System.out.println(p*r*n/100);
    }
}
public class Rate {
    public static void main(String[] args) {
        Bank A = new Bank();
        A.intrest(6);
        Bank B = new Bank();
        B.intrest(5.8);
        Bank C = new Bank();
        C.intrest(10.12);
    }
}
```

Output :



```
"C:\Program Files\Java\jdk-18\bin\java.exe" "-
60.0
58.0
101.2

Process finished with exit code 0
```



```
"C:\Program Files\Java\jdk-18\bin\java.exe"
6.0
5.8
10.12

Process finished with exit code 0
```

for p = 100 , N = 1