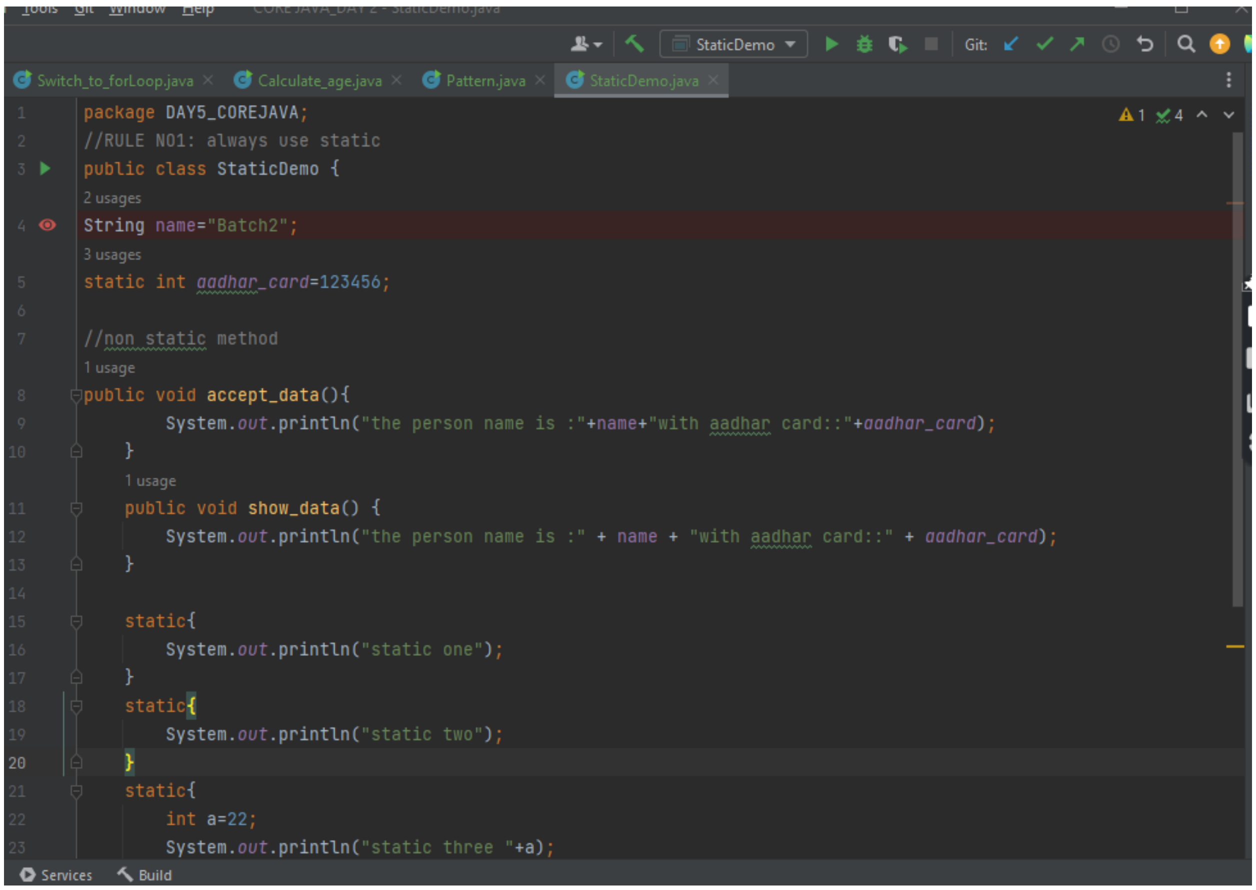
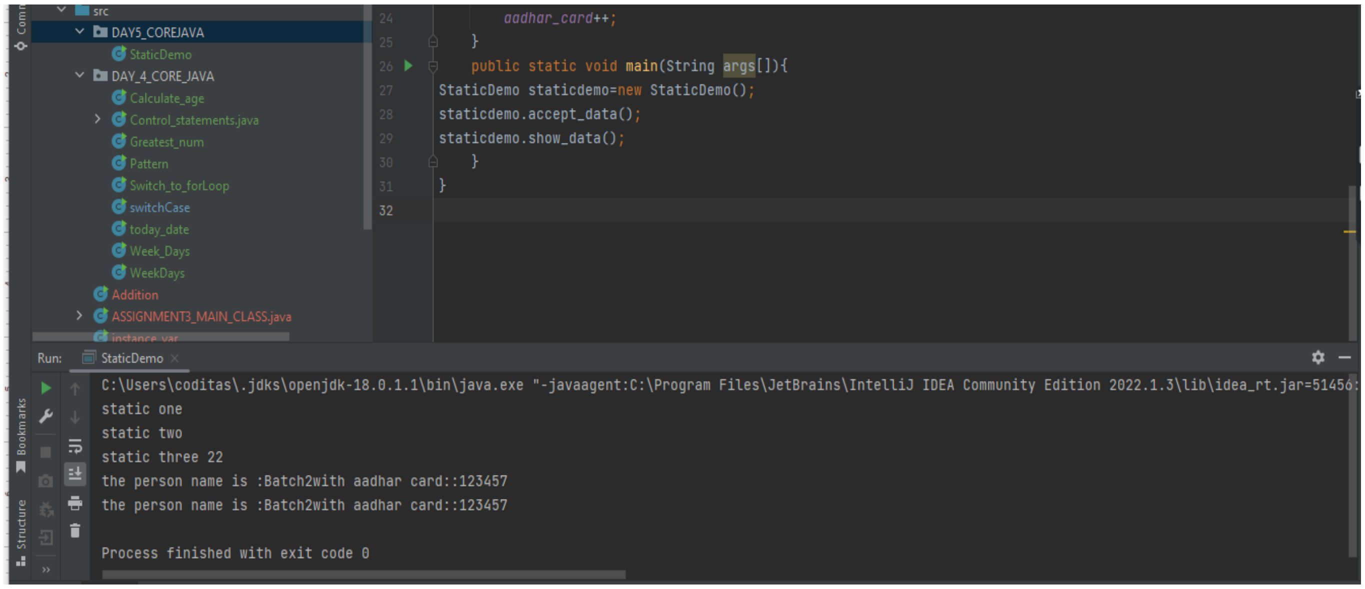
CORE JAVA DAY 5 ASSIGNMENT: DT:25/07/2022:-

Q:1)WAP of static keyword with all possible ways. (variable,method,class, block).

Ans:1)



```java
package DAY5_COREJAVA;
//RULE NO1: always use static
public class StaticDemo {
String name="Batch2";
static int aadhar_card=123456;

//non static method
public void accept_data(){
        System.out.println("the person name is :"+name+"with aadhar card::"+aadhar_card);
    }
    public void show_data() {
        System.out.println("the person name is :" + name + "with aadhar card::" + aadhar_card);
    }

    static{
        System.out.println("static one");
    }
    static{
        System.out.println("static two");
    }
    static{
        int a=22;
        System.out.println("static three "+a);
```



```java
        aadhar_card++;
    }
    public static void main(String args[]){
StaticDemo staticdemo=new StaticDemo();
staticdemo.accept_data();
staticdemo.show_data();
    }
}
```

```
C:\Users\coditas\.jdks\openjdk-18.0.1.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.1.3\lib\idea_rt.jar=51456:
static one
static two
static three 22
the person name is :Batch2with aadhar card::123457
the person name is :Batch2with aadhar card::123457

Process finished with exit code 0
```

Q:2)

ANS:2)

Static methods (in fact all methods) as well as static variables are stored in the PermGen section of the heap, since they are part of the reflection data (class related data, not instance related).

Note that only the variables and their technical values (primitives or references) are stored in PermGen space.

If your static variable is a reference to an object that object itself is stored in the normal sections of the heap (young/old generation or survivor space). Those objects (unless they are internal objects like classes etc.) are *not* stored in PermGen space.

Prior to Java 8:

The static variables were stored in the permgen space(also called the method area).

PermGen Space is also known as Method Area

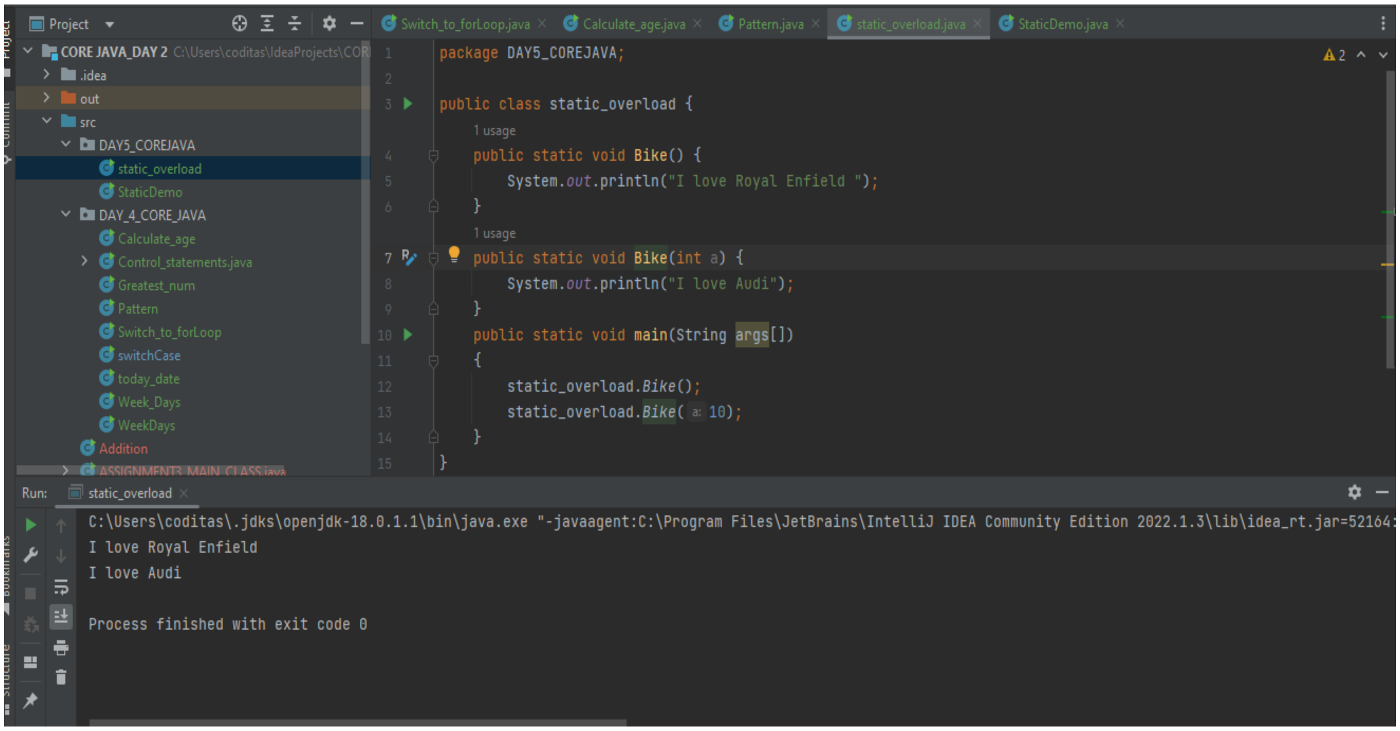
PermGen Space used to store 3 things

1. **Class level data (meta-data)**
2. **interned strings**
3. **static variables**

The static variables are stored in the Heap itself.From Java 8 onwards the PermGen Space have been removed and new space named as MetaSpace is introduced which is not the part of Heap any more unlike the previous Permgen Space. Meta-Space is present on the native memory (memory provided by the OS to a particular Application for its own usage)and now it only stores the class meta data.

Q:3.1)Prove practically=>Can we Overload static methods in Java?

Ans:3.1)Yes, we can overload the static method. A class can have more than one static method with the same name, but different in input parameters. You can overload the static method by changing the number of arguments or by changing the type of arguments.



```java
package DAY5_COREJAVA;

public class static_overload {
    1 usage
    public static void Bike() {
        System.out.println("I love Royal Enfield ");
    }
    1 usage
    public static void Bike(int a) {
        System.out.println("I love Audi");
    }
    public static void main(String args[])
    {
        static_overload.Bike();
        static_overload.Bike( a: 10);
    }
}
```

```
C:\Users\coditas\.jdks\openjdk-18.0.1.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.1.3\lib\idea_rt.jar=52164
I love Royal Enfield
I love Audi

Process finished with exit code 0
```

Q:3.2))Can we Override static methods in Java?

Ans:3.2)**No, we cannot override static methods because method overriding is based on dynamic binding at runtime and the static methods are bonded using static binding at compile time. So, we cannot override static methods.**

The calling of the method depends upon the type of object that calls the static method. It means:

- **If we call a static method by using the parent class object, the original static method will be called from the parent class.**
- **If we call a static method by using the child class object, the static method of the child class will be called.**

Q:3.3)Why is the main() method declared as static?

Ans:3.3)The main method is static in Java, so the JVM can directly invoke it without instantiating the class's object.

If the main method is non-static, then JVM needs to create an instance of the class, and there would be ambiguity if the constructor of that class takes an argument – which constructor should be called by JVM and what parameters should be passed? We know that JVM can't instantiate a Java class without calling a constructor method. We can agree upon a default constructor, but that's extra overhead. Also, the class must not be abstract; otherwise, the JVM could not instantiate it. To make Java a little less verbose than it already is, the main method is static in Java.

Q:4)Is there any error in the below code snippet? If yes, identify the error and give the reason behind it.

```
public class Demo

{

void m1(Demo demo) {

System.out.println("Instance method");

}
```

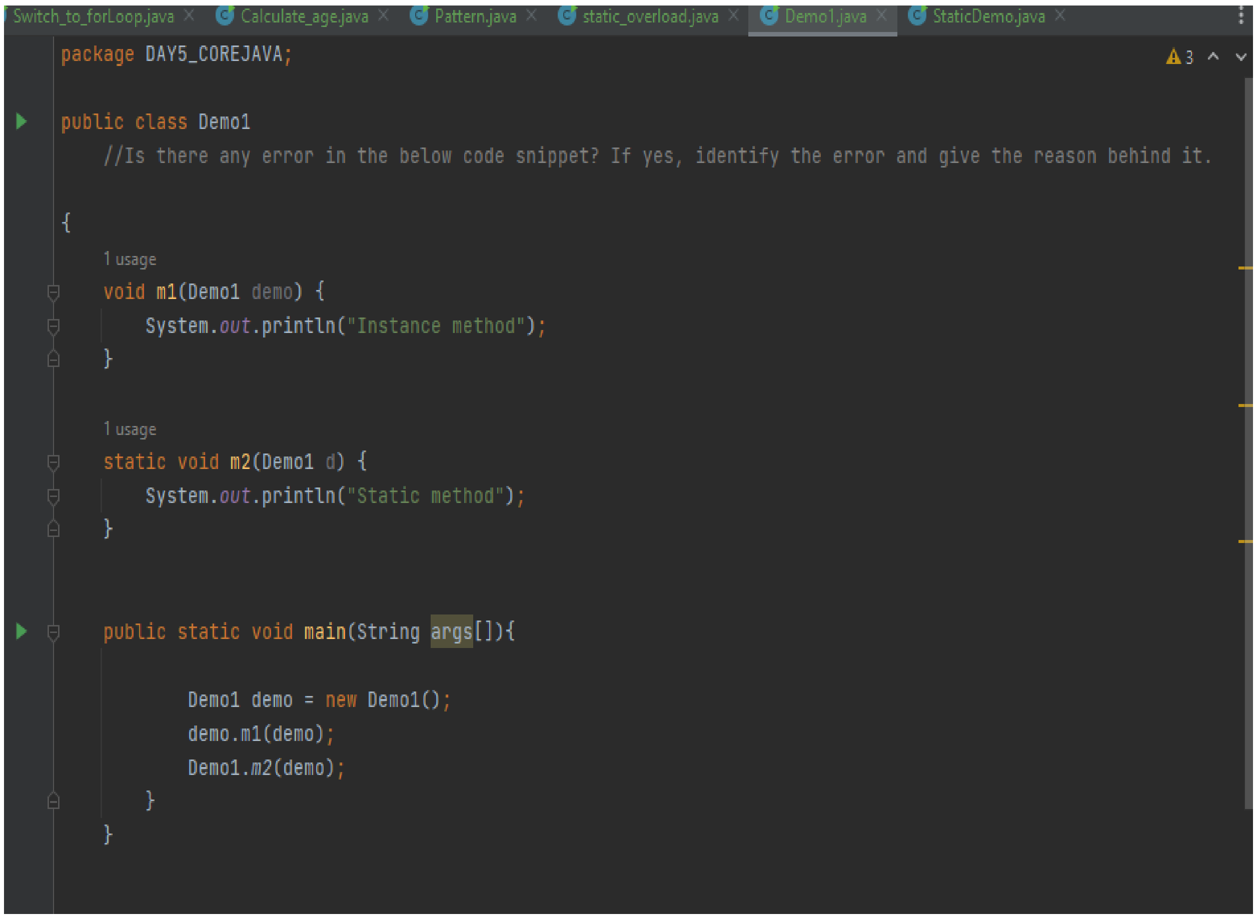
static void m1(Demo d) {

System.out.println("Static method");

}

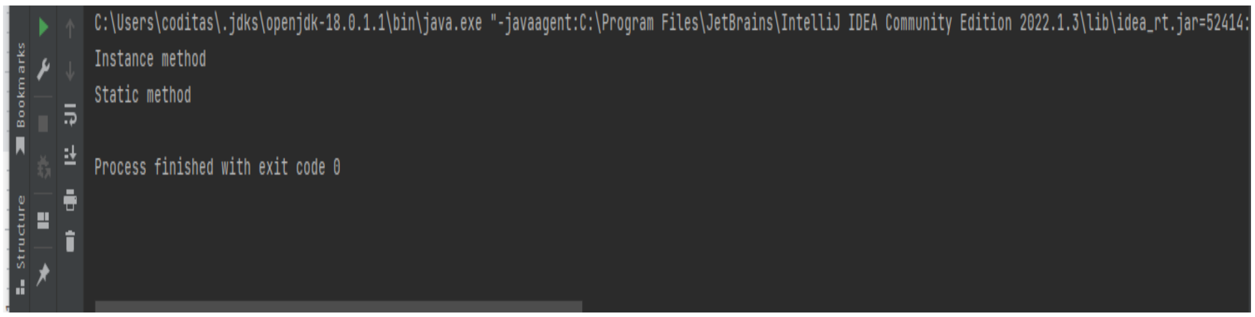
}

Ans:4)



```
package DAY5_COREJAVA;

public class Demo1
    //Is there any error in the below code snippet? If yes, identify the error and give the reason behind it.

{
    1 usage
    void m1(Demo1 demo) {
        System.out.println("Instance method");
    }

    1 usage
    static void m2(Demo1 d) {
        System.out.println("Static method");
    }


    public static void main(String args[]){

            Demo1 demo = new Demo1();
            demo.m1(demo);
            Demo1.m2(demo);
        }
    }
```

WE cannot overload two methods if they are only different by static keyword.



```
C:\Users\coditas\.jdks\openjdk-18.0.1.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.1.3\lib\idea_rt.jar=52414:
Instance method
Static method

Process finished with exit code 0
```

Q:5)Will the following code snippet compile fine? If yes, what will be output

public class Myclass{

private static int x=10;

static {

X++;

static {

++x;

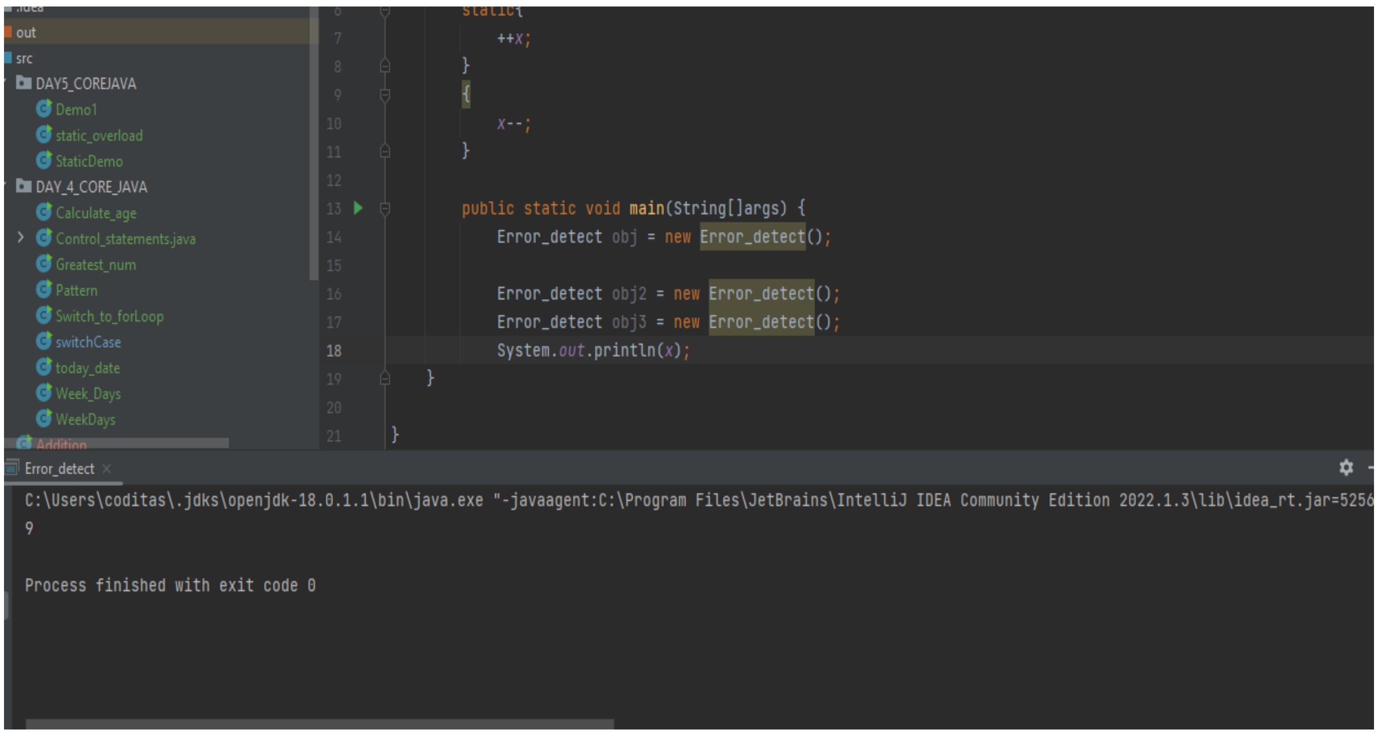
}

{

X--

```
}

public static void main(String[] args) {

Myclass obj = new Myclass();

Myclass obj2 = new Myclass();

}

}
```

Ans:5)



```
        static{
            ++x;
        }
        {
            x--;
        }

        public static void main(String[]args) {
            Error_detect obj = new Error_detect();

            Error_detect obj2 = new Error_detect();
            Error_detect obj3 = new Error_detect();
            System.out.println(x);
        }

    }
```

```
C:\Users\coditas\.jdks\openjdk-18.0.1.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.1.3\lib\idea_rt.jar=5256
9

Process finished with exit code 0
```

The output of the above code is :9