"static" Keyword

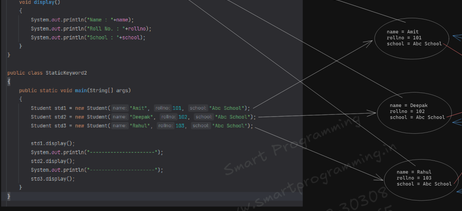
⇒ "static" Keyword is the non-access modifier
⇒ It can be used with
  - variables
  - methods
  - blocks
  - inner class
⇒ Use :-
  1. It is used for memory management
⇒ NOTE : static belongs to class, not objects

"static" variables

⇒ If we use static keyword with variable then it will be static variable
⇒ "static" variable is shared among all the objects of the class
⇒ Program

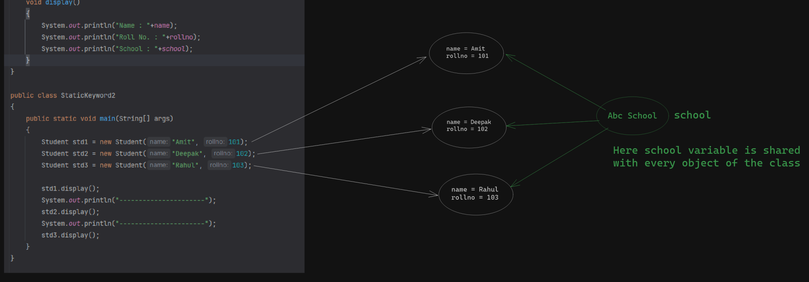
```java
class Student
{
    String name;
    int rollno;
    String school;

    Student(String name, int rollno, String school)
    {
        this.name = name;
        this.rollno = rollno;
        this.school = school;
    }

    void display()
    {
        System.out.println("Name : "+name);
        System.out.println("Roll No. : "+rollno);
        System.out.println("School : "+school);
    }
}

public class StaticKeyword2
{
    public static void main(String[] args)
    {
        Student std1 = new Student( name: "Amit",  rollno: 101, school: "Abc School");
        Student std2 = new Student( name: "Deepak",  rollno: 102,  school: "Abc School");
        Student std3 = new Student( name: "Rahul",  rollno: 103,  school: "Abc School");

        std1.display();
        System.out.println("----------------------");
        std2.display();
        System.out.println("----------------------");
        std3.display();
    }
}
```

Wrong Approach




```java
class Student
{
    String name;
    int rollno;
    String school = "Abc School";

    Student(String name, int rollno)
    {
        this.name = name;
        this.rollno = rollno;
    }

    void display()
    {
        System.out.println("Name : "+name);
        System.out.println("Roll No. : "+rollno);
        System.out.println("School : "+school);
    }
}

public class StaticKeyword2
{
    public static void main(String[] args)
    {
        Student std1 = new Student( name: "Amit",  rollno: 101);
        Student std2 = new Student( name: "Deepak",  rollno: 102);
        Student std3 = new Student( name: "Rahul",  rollno: 103);

        std1.display();
        System.out.println("----------------------");
        std2.display();
        System.out.println("----------------------");
        std3.display();
    }
}
```

Right Approach



⇒ Point to remember :
  1. static variables occupy memory in Method Area or Class Area
  2. static variable can be used in static method only, not in instance method
     but instance variable can be used in instance method and static method both.

"static" method

⇒ If we provide static keyword with any method, then it is known as static method
⇒ A static method belongs to class, not object
⇒ A static method is called using class name, not using object
⇒ Program

```java
public class staticKeyword3
{
    void m1()
    {
        System.out.println("m1 instance method");
    }

    static void m2()
    {
        System.out.println("m2 static method");
    }

    public static void main(String[] args)
    {
        staticKeyword3 obj = new staticKeyword3();
        obj.m1();

        staticKeyword3.m2();
    }
}
```

⇒ Points to remember:
  1. static method can use only static variables, not instance variables

"static" Block

⇒ It is a block which is loaded in the memory at class loading time
⇒ It is used to initialize the static variables
⇒ Syntax :

```
static
{
    //body
}
```

"static" inner class

⇒ static keyword can be used only with inner class, not outer class

```java
public class StaticKeyword6
{
    static class Test
    {

    }
}
```

→ If we use static keyword here, then it will provide an error

"enum" Keyword

⇒ enum is the keyword in Java
⇒ It is used to define a collection of constants (fixed set of values)
⇒ For eg.

```
enum Days
{
    MONDAY, TUESDAY, WEDNESDAY, THRUSDAY, FRIDAY, SATURDAY, SUNDAY;
}
```

⇒ NOTE : enum can have constructors, methods and variables also

⇒ Use in Real World Projects :-

```
enum Status
{
    SUCCESS, FAIL, PENDING;
}
-------------------------------------------------
enum OrderStatus
{
    PLACED, SHIPPED, DELIVERED, CANCELLED;
}
```

instanceof Operator

⇒ instanceof is an operator which is used to test whether an object is an instance of class or subclass
⇒ It returns the boolean value

```java
class A
{

}
class B extends A
{

}
public class InstanceofOperator1
{
    public static void main(String[] args)
    {
        A obj1 = new A();
        B obj2 = new B();

        System.out.println(obj1 instanceof A);
        System.out.println(obj1 instanceof B);

        System.out.println(obj2 instanceof A);
        System.out.println(obj2 instanceof B);
    }
}
```

Interview Questions

⇒ Static Keyword in Java
1. What is the purpose of the static keyword in Java?
2. What are the different usages of the static keyword? (variable, method, block, nested class)
3. Can we override a static method in Java? Why or why not?
4. What happens if a static method tries to use this or super?
5. Can we declare a constructor as static? Why or why not?
6. Can a static method be abstract? Why or why not?
7. Can we make a class static in Java? In which cases is it allowed?

⇒ Static Variables in Java
8. What is a static variable in Java?
9. What is the difference between a static variable and an instance variable?
10. Where are static variables stored in memory?
11. When is a static variable initialized?
12. Can static variables be private in Java?
13. What is the default value of a static variable?
14. How are static variables shared among objects?
15. Can we access static variables using an object reference?

⇒ Static Methods in Java
16. What is a static method in Java?
17. What is the difference between a static method and a non-static method?
18. Why can't static methods use this or super?
19. Can we call a static method using an object?
20. How can we call a static method from another class?
21. What happens if we declare the main method as non-static?

⇒ Static Blocks in Java
22. What is a static block in Java?
23. When is a static block executed?
24. Can a class have multiple static blocks? If yes, in which order are they executed?
25. What is the purpose of a static block? Give real-life use cases.
26. Can we use return statements in a static block? Why or why not?
27. What happens if we define a static block in a class without a main method?

⇒ Enum in Java
28. What is an enum in Java? Why do we use it?
29. How is enum different from a class in Java?
30. Can we extend an enum in Java? Why or why not?
31. Can an enum implement an interface?
32. Can we create a constructor inside an enum?
33. What are the values() and valueOf() methods in enums?
34. Can we use enums in switch statements? Explain with an example.
35. Can enums have methods and fields like normal classes?
36. What is the difference between enum in Java and final static constants?

⇒ instanceof Operator in Java
37. What is the instanceof operator in Java?
38. What is the return type of the instanceof operator?
39. What happens if we use instanceof with null?
40. Can instanceof be used with interfaces?
41. Can instanceof be used with generic types?
42. How does instanceof differ from getClass()?
43. What happens if we compare unrelated classes with instanceof?