DAY-26

19 July 2023 15:06

STATIC METHODS AND STATIC VARIABLES

1. HOW TO ACCESS STATIC MEMBER OF ONE CLASS WITHIN SAME CLASS

```
class Test1
{
	static int a = 10;
	static double b = 3.14;
	public static void main(String[] args)
	{
		System.out.println(a);
		m1();
		System.out.println(b);
	}

	public static void m1()
	{
		System.out.println("inside m1 method");
	}

}
```

2. HOW TO ACCESS STATIC MEMBER OF ONE CLASS IN OTHER CLASS WITHIN SAME SOURCE FILE

```
class Test1
{
	static int a = 10;
	static double b = 3.14;
	public static void main(String[] args)
	{
		int x = 5;
		System.out.println(x);

	}

	public static void m1()
	{
		System.out.println("inside m1 method");
	}

}

class Test2
{
	public static void main(String[] args)
	{
		System.out.println(Test1.a);
```

```
        Test1.m1();
        System.out.println(Test1.b);
    }
}
```

- REINTIALIZATION OF A SATIC VARIABLE OF ONE CLASS IN OTHER CLASS

```
class Test1
{
    static int x = 1;
    static double y = 2;
    public static void main(String[] args)
    {

        System.out.println(x);
        System.out.println(y);

    }


}

class Test2
{
    public static void main(String[] args)
    {
        System.out.println(Test1.x);
        System.out.println(Test1.y);

        Test1.x = 10;
        Test1.y = 2.2;

        System.out.println(Test1.x);
        System.out.println(Test1.y);


    }
}
```

NOTE :

- WE CAN MAKE STATIC VARIABLE AS FINAL
- IF WE MAKE STATIC VARIABLE AS FINAL THEN WE CAN NOT REINTIALISE IT

➢ PROGRAM TO PRINT THE DEFAULT VALUES OF STATIC VARIABLES

```
    class Test1
    {
        static int a;
        static double b;
        static float f;
        static String s;
        static char c;

        public static void main(String[] args)
        {

                System.out.println(a);
                System.out.println(b);
                System.out.println(f);
                System.out.println(s);
                System.out.println(c);

        }


}
```

IMPORTANT CONCLUSIONS

- WE CAN ACCESS STATIC MEMBERS OF ONE CLASS WITHIN THE SAME CLASS DIRECTLY
- WE CAN ACCESS STATIC MEMBERS OF ONE CLASS IN ANOTHER CLASS BY USING CLASS NAME
  - ◻ SYNTAX :
    - ◆ CLASS_NAME.STATIC_METHOD()
    - ◆ CLASS_NAME.STATIC_VARIABLE_NAME;
- WE CAN REINITIALIZE STATIC MEMBERS OF ONE CLASS IN ANOTHER CLASS USING CLASS NAME
  - ◻ SYNTAX : CLASS_NAME.STATIC_VARIABLE_NAME = NEW_VALUE
- WE CAN USE THE FINAL KEYWORD WITH STATIC VARIABLE ALSO, IF WE USE FINAL KEYOWRD WE CAN NOT REINITIALIZE THE VARIABLE

TASK - 1

1. IN FIRST CLASS THERE MUST TWO METHODS
   - **a.** 1ST SHOULD READ THE NUMBER
   - **b.** 2ND METHOD SHOULD FIND THE PALINDROME OF THAT NUMBER
2. IN THE SECOND CLASS THERE MUST BE A MAIN METHOD , IN WHICH METHOD CALLING SHOULD HAPPEN FOR BOTH THE STATIC METHODS

STATIC MEMBERS LOADING

- WE CAN ACCESSS STATIC MEMBERS OF ONE CLASS BY USING CLASS NAME BECAUSE THE STATIC MEMBERS WILL LOADED INTO CLASS MEMORY WHILE WE LOAD THE CLASS TO JVM FOR EXECUTION AND THIS CLASS MEMORY WILL HAVE SAME AS THAT OF CLASS NAME

```
class Test1
{

        static int r = 7;
        static String s = "BitsQ";
```

```
          public static void m1()
          {
                    System.out.println("Hi");
          }


}

class Test2
{
          public static void main(String[] args)
          {

                    Test1.m1();
                    System.out.println(Test1.r);

          }
}
```