Object Oriented Programming with Java (OOPJ)

Session 3: Operators & Basics

Kiran Waghmare

Reading Different Types of Input

Reading Different Types of Input Method	Reads	Example Input
nextInt()	Integer	10
nextDouble()	Double (decimal)	3.14
nextFloat()	Float (decimal)	5.75
nextLong()	Long Integer	123456789
nextBoolean()	Boolean	true / false
next()	Single word	"Hello"
nextLine()	Full line (including spaces)	"Hello World"

Arithmetic Operators

Operator	Description	Example	Output
+	Addition	10 + 5	15
-	Subtraction	10 - 5	5
*	Multiplication	10 * 5	50
/	Division	10 / 5	2
%	Modulus (Remainder)	10 % 3	1

Relational Operators

Operator	Description	Example	Output
==	Equal to	10 == 5	false
!=	Not equal to	10 != 5	true
>	Greater than	10 > 5	true
<	Less than	10 < 5	false
>=	Greater than or equal to	10 >= 5	true
<=	Less than or equal to	10 <= 5	false

Logical Operator

Operator	Description	Example	Output
&&	Logical AND	(10 > 5) && (5 < 10)	true
||	Logical OR	(10 > 5) || (5 < 10)	true
!	Logical NOT	!(10 > 5)	false

Bitwise Operator

Operator	Description	Example	Output
&	Bitwise AND	5 & 3 (0101 & 0011)	1
|	|	Bitwise OR	`5
^	Bitwise XOR	5 ^ 3 (0101 ^ 0011)	6
~	Bitwise NOT	~5 (~0101)	-6
<<	Left Shift	5 << 1	10
>>	Right Shift	5 >> 1	2

Assignment Operator

Operator	Description	Example	Equivalent
=	Assign	x = 5	x = 5
+=	Add and assign	x += 5	x = x + 5
-=	Subtract and assign	x -= 5	x = x - 5
*=	Multiply and assign	x *= 5	x = x * 5
/=	Divide and assign	x /= 5	x = x / 5
%=	Modulus and assign	x %= 5	x = x % 5

Bitwise Operator

Operator	Description	Example	Binary Representation	Output
&	Bitwise AND	5 & 3 (0101 & 0011)	0101 & 0011 = 0001	1
`	`	Bitwise OR	`5	3 (0101
^	Bitwise XOR	5 ^ 3 (0101 ^ 0011)	0101 ^ 0011 = 0110	6
~	Bitwise NOT	~5 (~0101)	~0101 = 1010 (2's complement)	-6
<<	Left Shift	5 << 1	0101 << 1 = 1010	10
>>	Right Shift	5 >> 1	0101 >> 1 = 0010	2

Bitwise Shift Operator

Operator	Operation	Zero Fill?	Used For
x << n	Left Shift	Yes (right-side)	Multiplication by 2^n
x >> n	Right Shift	No (preserves sign bit)	Division by 2^n
>>>	Unsigned Right Shift	Yes (left-side)	Handling unsigned data

What will be the output of the following code?

```
int a = 5;
int b = 3;
System.out.println(a++ * --b);
```

A) 15
B) 12
C) 14
D) 10

What will be the result of the expression 10 / 3 * 3 in Java?

A) 3
B) 10
C) 9
D) 0

What is the output of the following code?

```
int x = 10;
int y = 20;
System.out.println(x == 10 & y == 20);
```

A) true
B) false
C) Compilation error
D) Runtime error

Which of the following expressions will not compile?

A) int x = 10.5;
B) int x = (int)10.5;
C) double x = 10.5;
D) double x = (double)10;

What will be the output of the following code?

```
boolean b = true;
System.out.println(b || false && true);
```

A) true
B) false
C) Compilation error
D) Runtime error

What will be the output of the following code?

```
int x = 5;
if (x > 3) {
    if (x < 7) {
        System.out.println("Hello");
    }
}
```

A) No output
B) "Hello"
C) Compilation error
D) Runtime exception

What will be the output of the following code?

```
int x = 0;
for (int i = 0; i < 5; i++) {
    x++;
    if (x == 3) {
        break;
    }
}
System.out.println(x);
```

A) 3

B) 5

C) 4

D) Infinite loop

What is the result of the following code?

```
int i = 0;
while (i < 5) {
    if (i == 3) {
        continue;
    }
    System.out.println(i);
    i++;
}
```

A) 0 1 2 3 4
B) 0 1 2 4
C) 1 2 3 4
D) Infinite loop

Consider the following code:

```
int x = 0;
for (int i = 1; i < 5; i++) {
    x += i;
    if (x > 6) break;
}
System.out.println(x);
```

What is the output of this code?

A) 6
B) 7
C) 10
D) 4

```
class Equals {
    public static void main(String[] args) {
        int x = 100;
        double y = 100.1;
        boolean b = (x = y);
        System.out.println(b);
    }
}
```

```
class Test{
  public static void main(String args[]){
    int i = (int)(char)(byte)-2;
    System.out.println("Output : " + i);
  }
}
```

OOP:Object Oriented Programming

-OOP is a programming paradigm based on [illegible] You are screen sharing HI Stop share [illegible]ch contain data(attributes) and methods(behaviour).

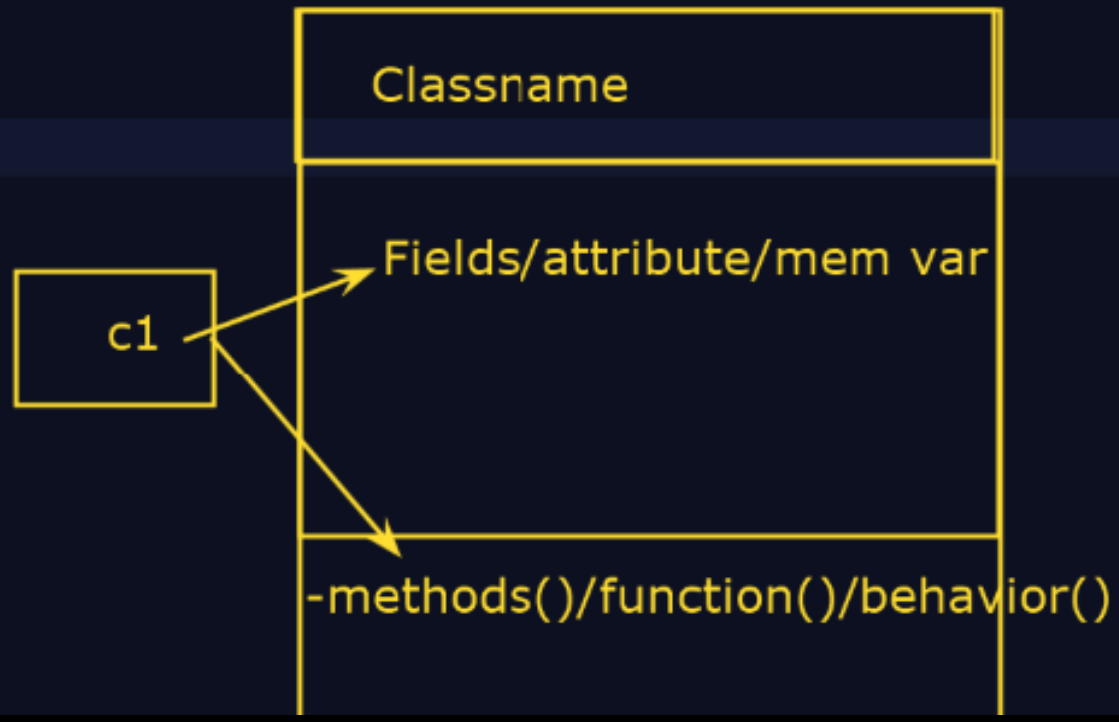
-Java follows the OOP paradigm, making it easy to organize and mange code e[illegible]

Class:

-A class is a blueprint or template that defines the properties(attributes/f[illegible] behaviour (defined by methods)

Object:

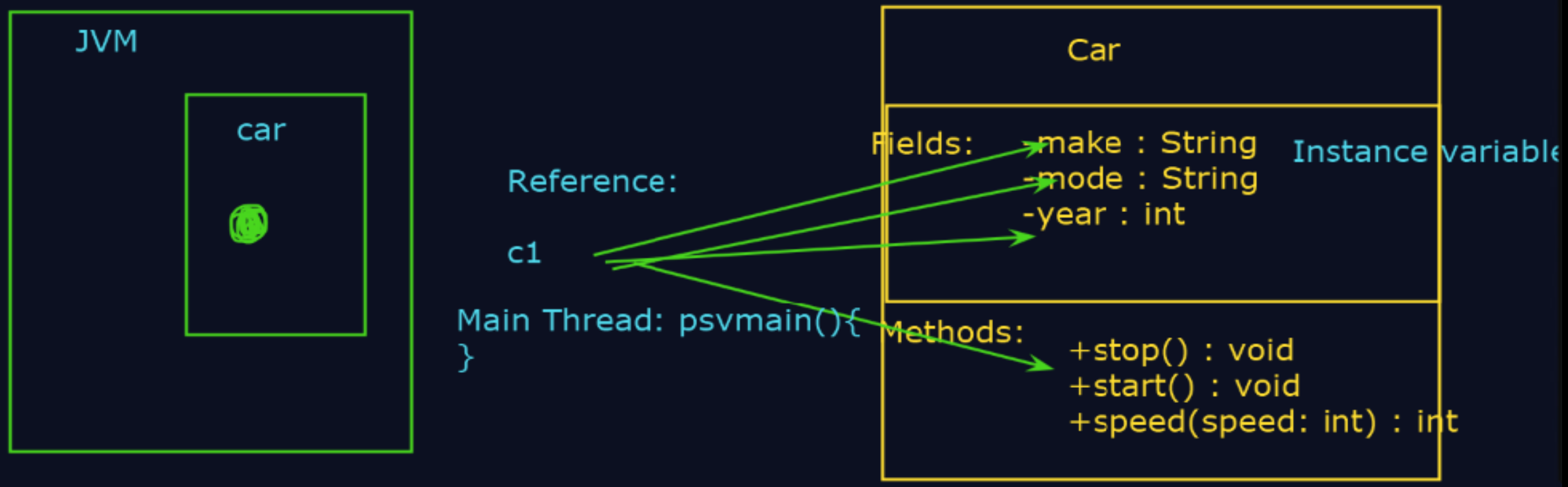
-An object is an instance of a class.



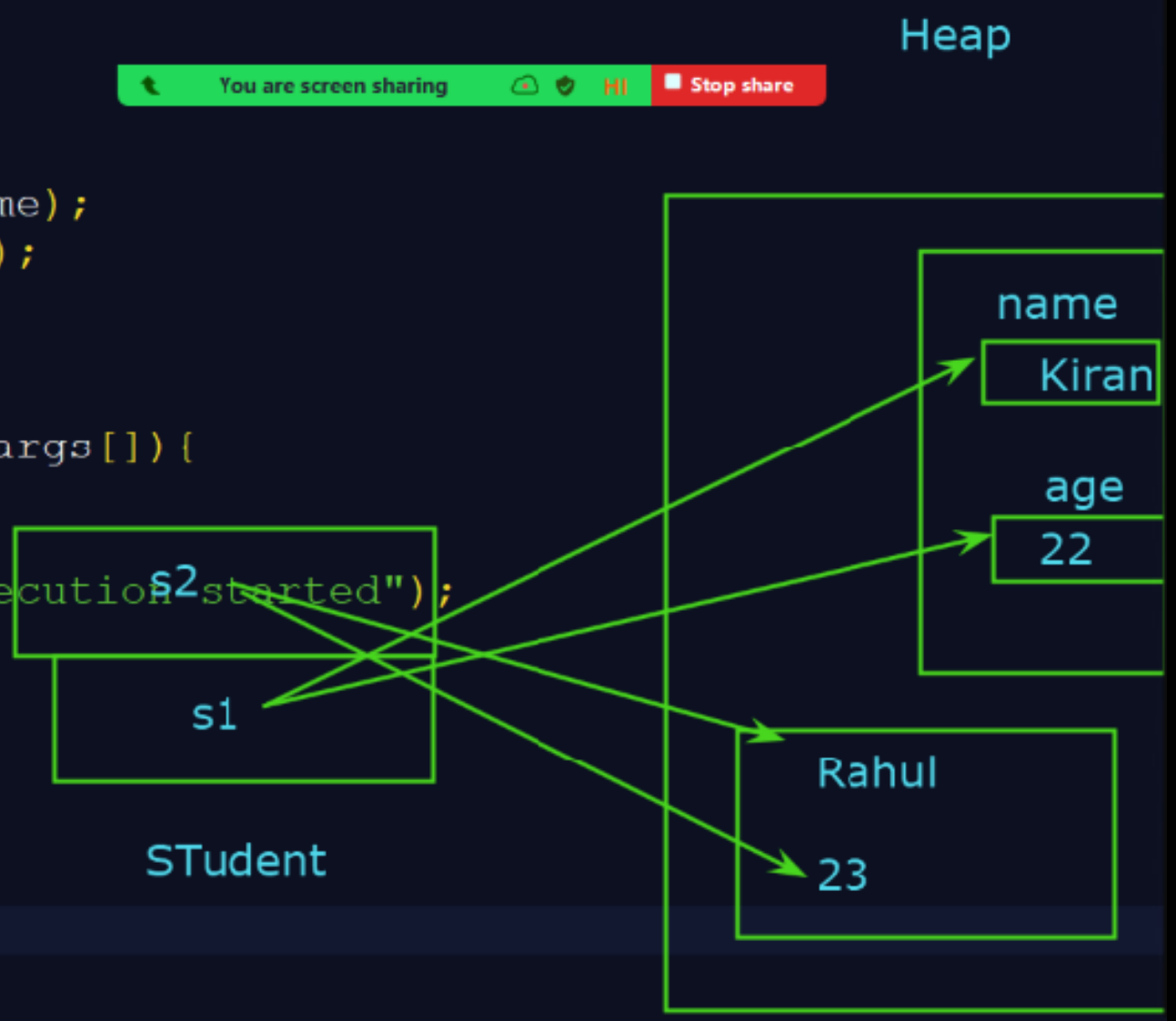
Features of OOP:

1.Object and Classes : Object are real world entities and classes are their
2.Encapsulation : Data Hiding implementation details and expressing only ne
functionality.

3.Abstraction : Simplifying complex system by focusing on essential details
4.Inheritance: Enabling code reuse by creating new classes from existing on
5.Polymorphism : Allowing one interface to be used for different implementa



```java
class Student{
    String name;
    int age;

    void display(){
        System.out.println("Name = "+name);
        System.out.println("Age = "+age);
    }


    public static void main(String args[]){


        System.out.println("Main Execution started");
        Student s1 = new Student();
        s1.name= "Kiran";
        s1.age=22;

        Student s2 = new Student();
        s1.name= "Rahul";
        s1.age=23;
```

```java
String name;
int age;

void display(){
    System.out.println("Name = "+name);
    System.out.println("Age = "+age);
}


    public static void main(String args[]){


        System.out.println("Main Execution st
        Student s1 = new Student();
        s1.name= "Kiran";
        s1.age=22;
        s1.display();

        Student s2 = new Student();
        s2.name= "Rahul";
        s2.age=23;
        s2.display();


}
```

You are screen sharing Stop share

C:\WINDOWS\syster

```
                    age=20;
                    ^
2 errors

C:\Test>javac Student.java

C:\Test>java Student
Main Execution started

C:\Test>javac Student.java

C:\Test>java Student
Main Execution started
Name = Kiran
Age = 22
Name = Rahul
Age = 23

C:\Test>
```

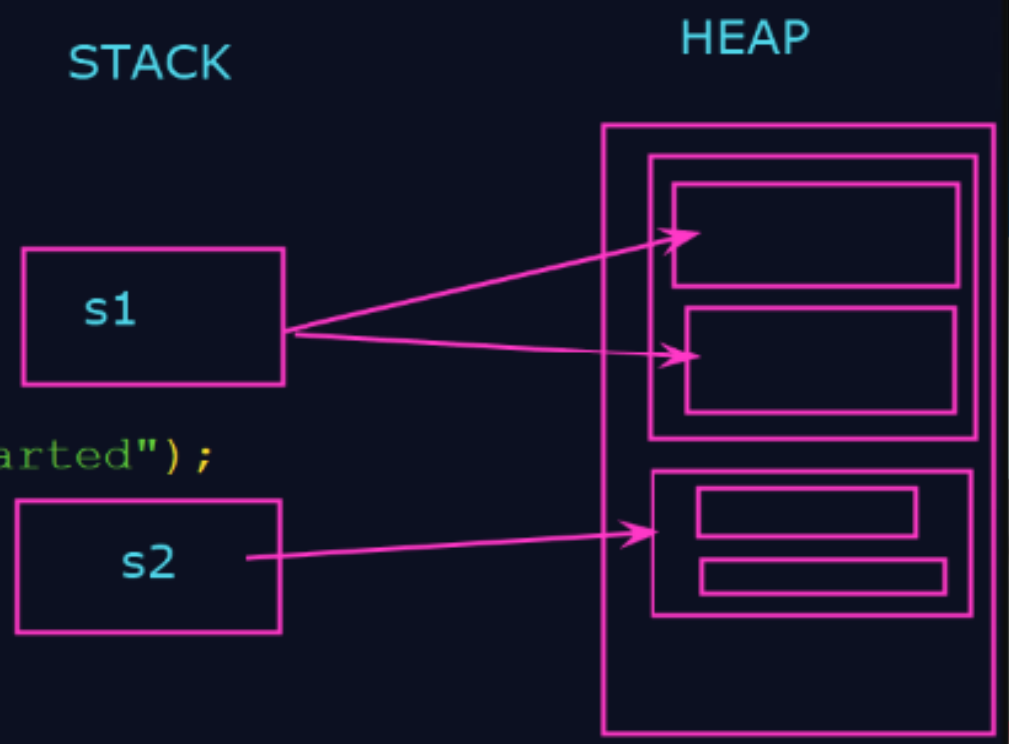
```java
class Student1{
    String name;
    int age;

    void display(){
        System.out.println("Name = "+name);
        System.out.println("Age = "+age);
    }


    public static void main(String args[]){


        System.out.println("Main Execution st
        Student s1 = new Student();
        s1.name= "Kiran";
        s1.age=22;
        s1.display();

        Student s2 = new Student();
        s2.name= "Rahul";
        s2.age=23;
        s1.display();
```

You are screen sharing HI Stop share

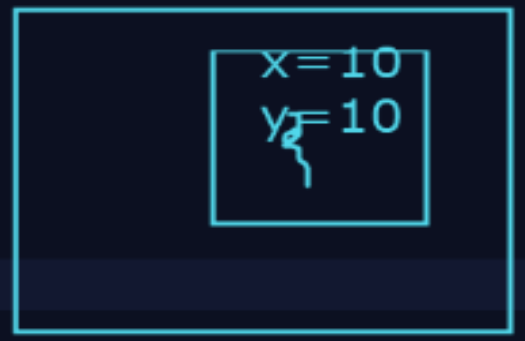
C:\WINDOWS\syster

```
C:\Test>javac Student.java

C:\Test>java Student
Main Execution started
Name = Kiran
Age = 22
Name = Rahul
Age = 23

C:\Test>javac Student1.java

C:\Test>java Student1
Main Execution started
Name = Kiran
Age = 22
Name = Kiran
Age = 22

C:\Test>
```

```java
String name;
int age;

void display(){
    System.out.println("Name = "+name);
    System.out.println("Age = "+age);
}


    public static void main(String args[]){

        Student1 s4 = null;
        Student1 s1 = new Student1();

        System.out.println("Main Execution started");
        Student1 s1 = new Student1();
        s1.name= "Kiran";
        s1.age=22;
        s1.display();


        s1.name= "Rahul";
        s1.age=23;
        s1.display();

        Student1 s2 = new Student1();
```

```
class MathOperation{

}

class MathOperationsDemo{
    static int x = 10;
    static int y = 20;
    int z =30;

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

            MathOperationsDemo m1 = new MathOperationsDemo();
            System.out.println(m1.z);


    }


}
```



```java
public static void main(String args[]){
    TestDemo t1 = new TestDemo();
    t1.a = 100;
    t1.name = "Prajwal";
    t1.display();

    //System.out.println("Local variable=
    System.out.println(Test.counter);
    //Test.show();//Error: non-static meth

    Test t2 = new Test();
    t2.show();

    t1.setnum(1000);
    t1.display();

}
```

C:\WINDOWS\syster

```
C:\Test>javac TestDemo.java

C:\Test>java TestDemo
Local variable=100
a = 100
Name = Prajwal
0
Additng counter:0
a = 1000
Local variable=100
a = 1000
Name = Prajwal

C:\Test>
```

```java
class StaticDemo{

    int x=10;//instance variable
    static int y =20;

    static{
        y = 2000;
        System.out.println("Static block: "+y);
    }

    void display(){
        System.out.println("Display(): Instance method()");
    }

    static void show(){
        System.out.println("Show(): static method()");
    }

    public static void main(String args[]){

        //System.out.println(x);//Error: access nahi kar sa
        System.out.println(y);
        System.out.println(Demo.m);

        StaticDemo d1 = new StaticDemo();//Instance created
        d1.display();//Instance method call

        show();//static method call
```

You are screen sharing Stop share

```
Show(): static method()
Show(): Demo :static method()

C:\Test>javac StaticDemo.java

C:\Test>java StaticDemo
Static block: 2000
2000
100
Display(): Instance method()
Show(): static method()
Show(): Demo :static method()

C:\Test>
```

```java
void display(){
    System.out.println("Display(): Instance method()");
}

static void show(){
    System.out.println("Show(): static method()");
}

public static void main(String args[]){

    //System.out.println(x);//Error: access nahi kar sa
    System.out.println(y);
    System.out.println(Demo.m);

    StaticDemo d1 = new StaticDemo();//Instance create
    d1.display();//Instance method call

    show();//static method call
    Demo.show1();//static method call

}

static{
    y1 = 2000;
    System.out.println("Static block y: "+y);
    System.out.println("Static block y1: "+y1);
}
```

```
Static block y1: 2000
20
100
Display(): Instance method()
Show(): static method()
Show(): Demo :static method()

C:\Test>javac StaticDemo.java

C:\Test>java StaticDemo
Static block y: 20
Static block y1: 2000
20
100
Display(): Instance method()
Show(): static method()
Show(): Demo :static method()

C:\Test>
```