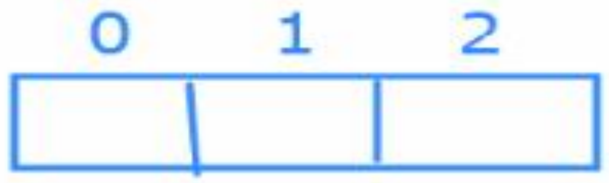
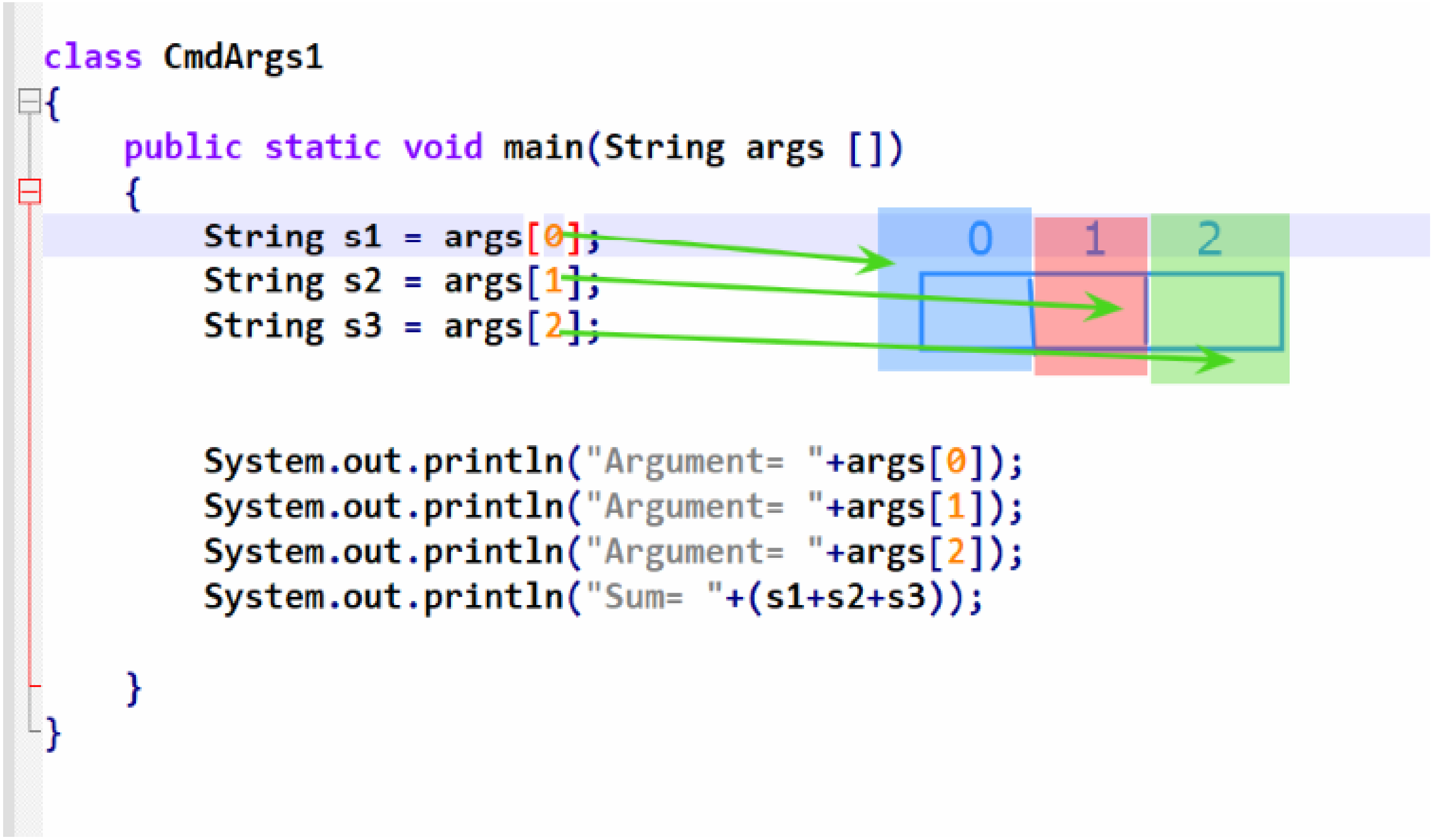
Logic Building Session Day 3: March 2022

Kiran Waghmare

```java
class CmdArgs1
{
    public static void main(String args [])
    {
        String s1 = args[0];
        String s2 = args[1];
        String s3 = args[2];


        System.out.println("Argument= "+args[0]);
        System.out.println("Argument= "+args[1]);
        System.out.println("Argument= "+args[2]);
        System.out.println("Sum= "+(s1+s2+s3));

    }
}
```

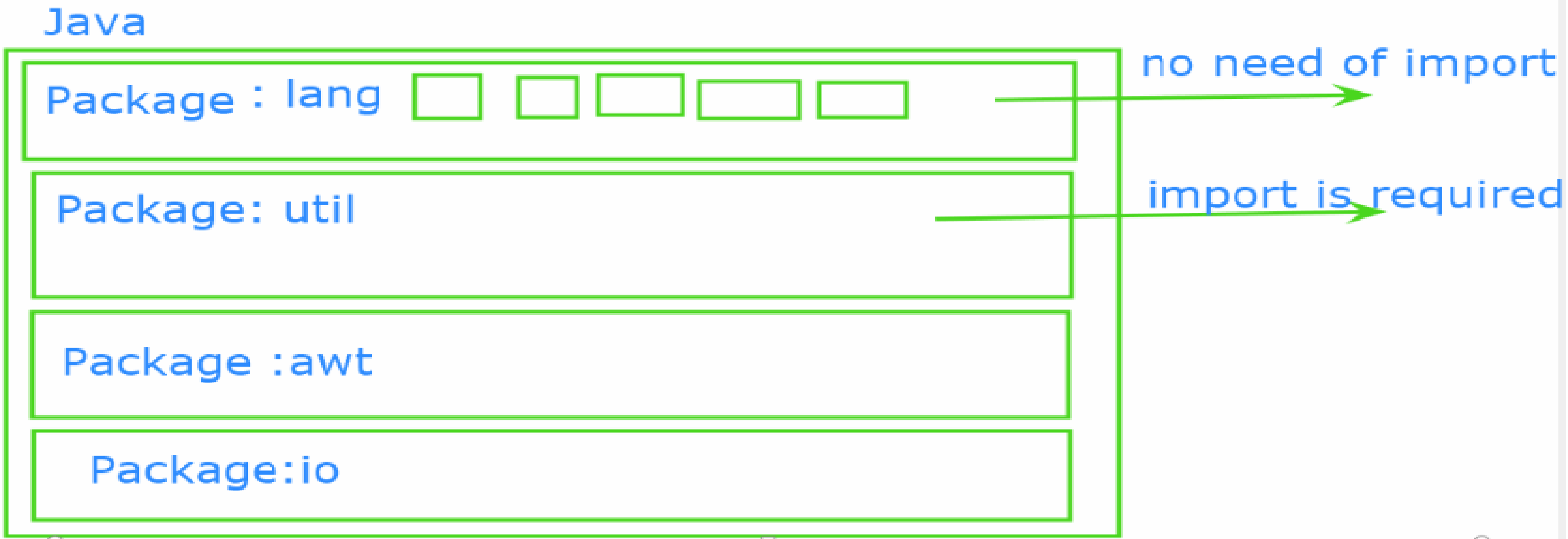




```
class CmdArgs1
{
    public static void main(String args [])
    {
        String s1 = args[0];
        String s2 = args[1];
        String s3 = args[2];


        System.out.println("Argument= "+args[0]);
        System.out.println("Argument= "+args[1]);
        System.out.println("Argument= "+args[2]);
        System.out.println("Sum= "+(s1+s2+s3));

    }
}
```

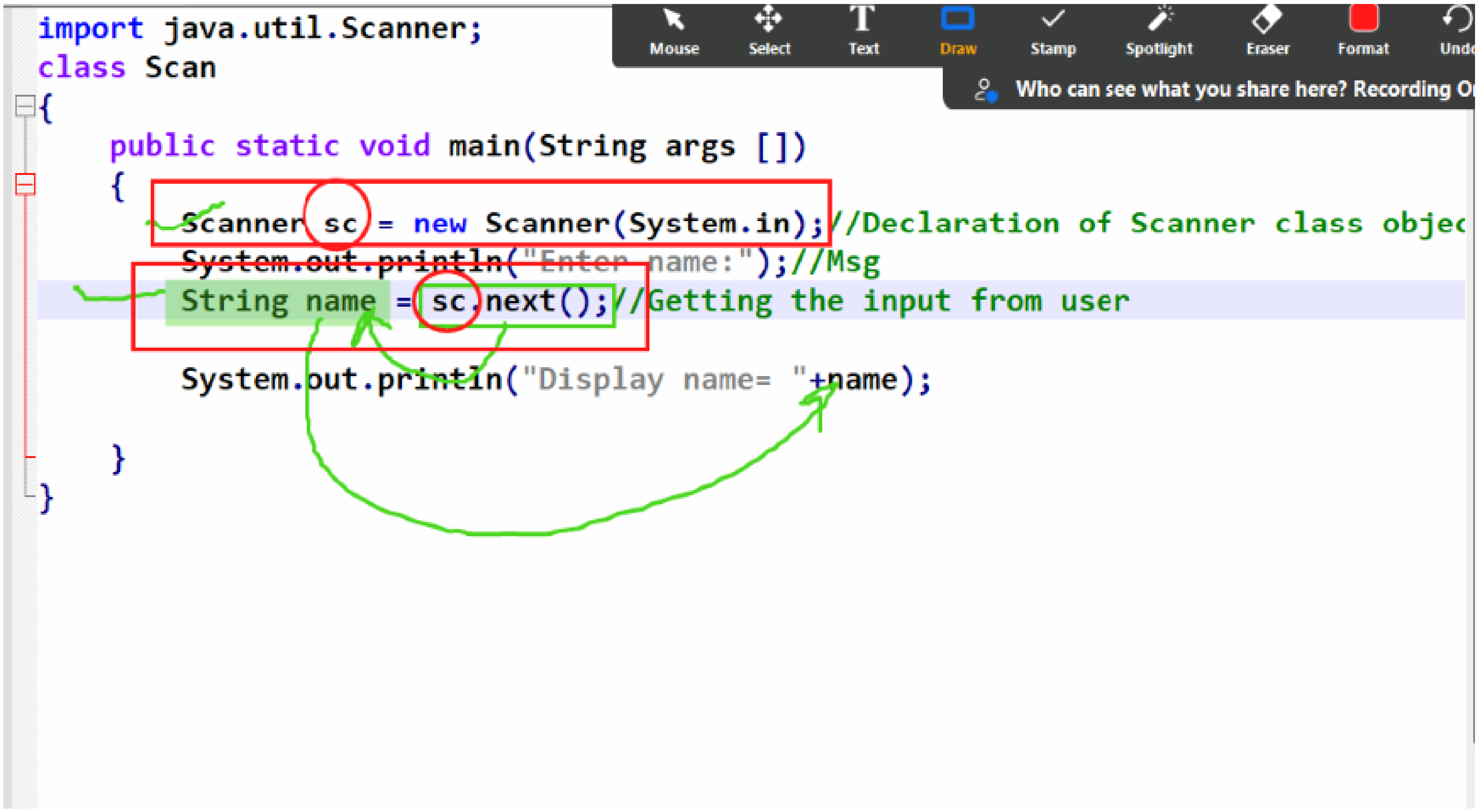
```
USer Input:
1. Command Line Arguments
2. Scanner class:

Package: import java.util.*;
```

import java.lang.*;
import java.util.Scanner;



Scanner: inbuilt java class that scans the given resource for any String or primitive value

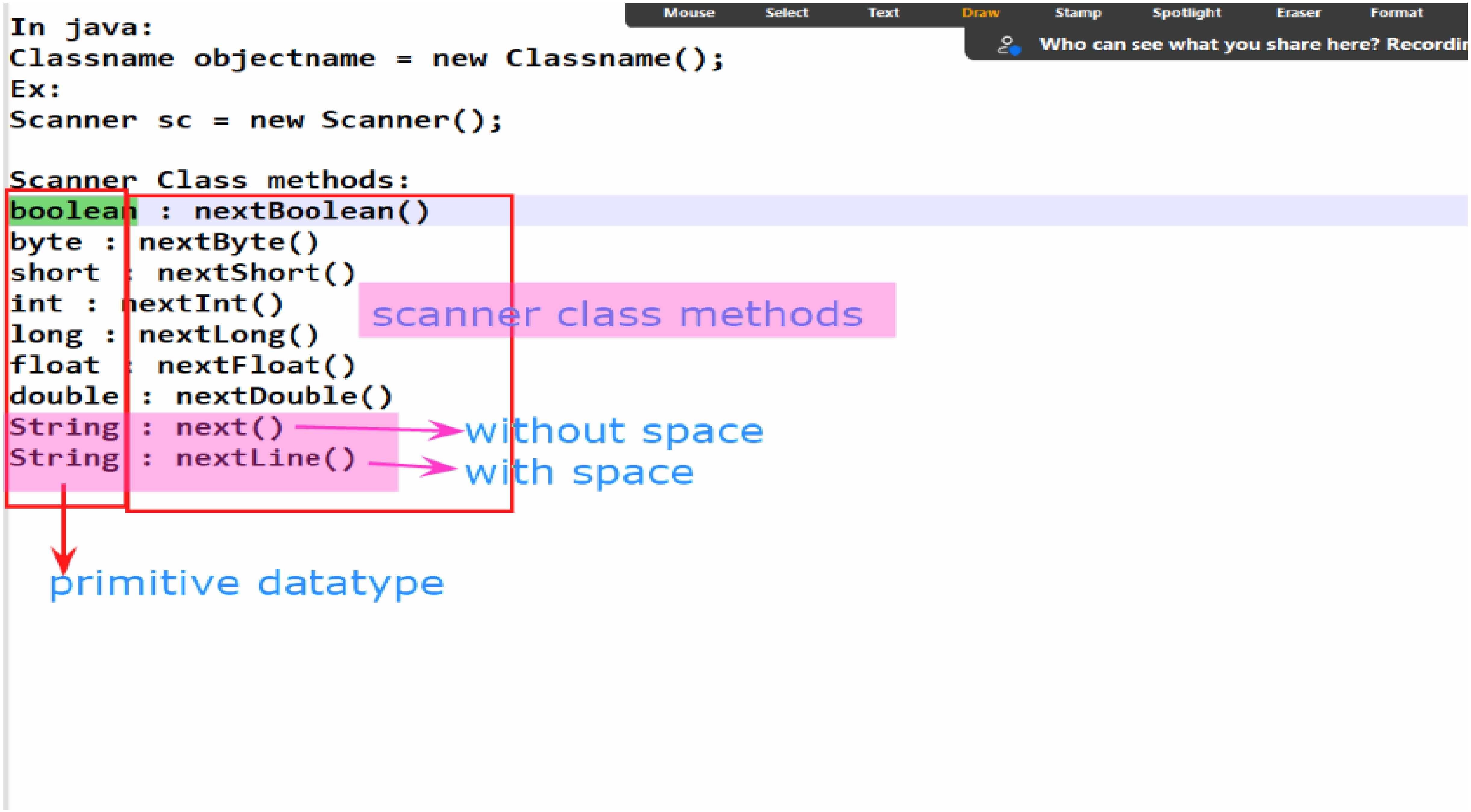
```
USer Input:
1. Command Line Arguments
2. Scanner class:

Package: import java.util.*;

syntax for object creation:
---------------------------
Classname objectname;
                                        Memory allocate
In java:
Classname objectname = new Classname();
Ex:
Scanner sc = new Scanner();
```



Mouse Select Text Draw Stamp Spotlight Eraser Format Undo

Who can see what you share here? Recording On

```
import java.util.Scanner;
class Scan
{
    public static void main(String args [])
    {
        Scanner sc = new Scanner(System.in);//Declaration of Scanner class objec
        System.out.println("Enter name:");//Msg
        String name = sc.next();//Getting the input from user

        System.out.println("Display name= "+name);

    }
}
```



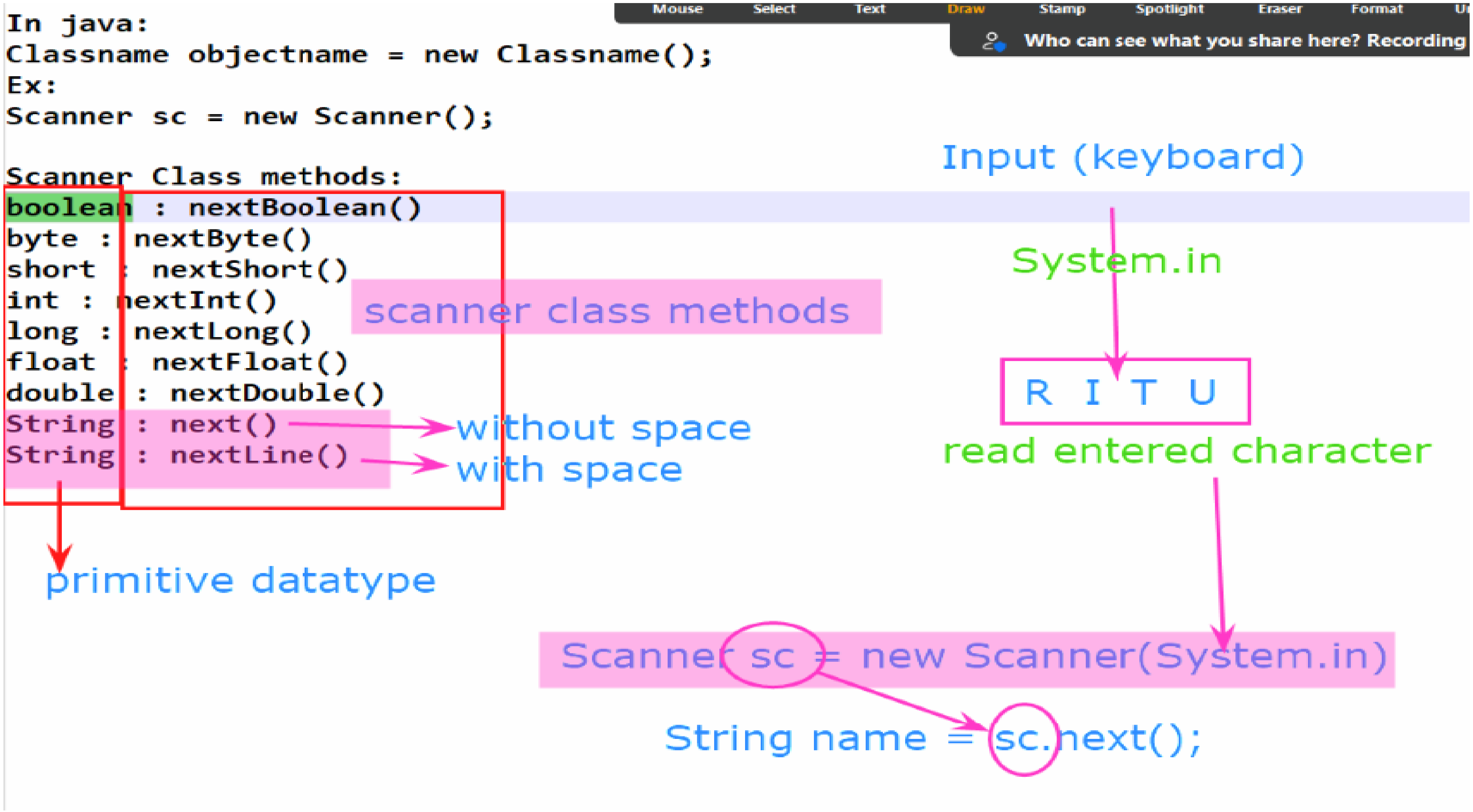
```
In java:
Classname objectname = new Classname();
Ex:
Scanner sc = new Scanner();

Scanner Class methods:
boolean : nextBoolean()
byte : nextByte()
short : nextShort()
int : nextInt()
long : nextLong()
float : nextFloat()
double : nextDouble()
String : next()
String : nextLine()
```

scanner class methods

next() → without space

nextLine() → with space

primitive datatype



```
In java:
Classname objectname = new Classname();
Ex:
Scanner sc = new Scanner();

Scanner Class methods:
boolean : nextBoolean()
byte : nextByte()
short : nextShort()
int : nextInt()
long : nextLong()
float : nextFloat()
double : nextDouble()
String : next()
String : nextLine()
```

scanner class methods

String : next() → without space

String : nextLine() → with space

primitive datatype

Input (keyboard)

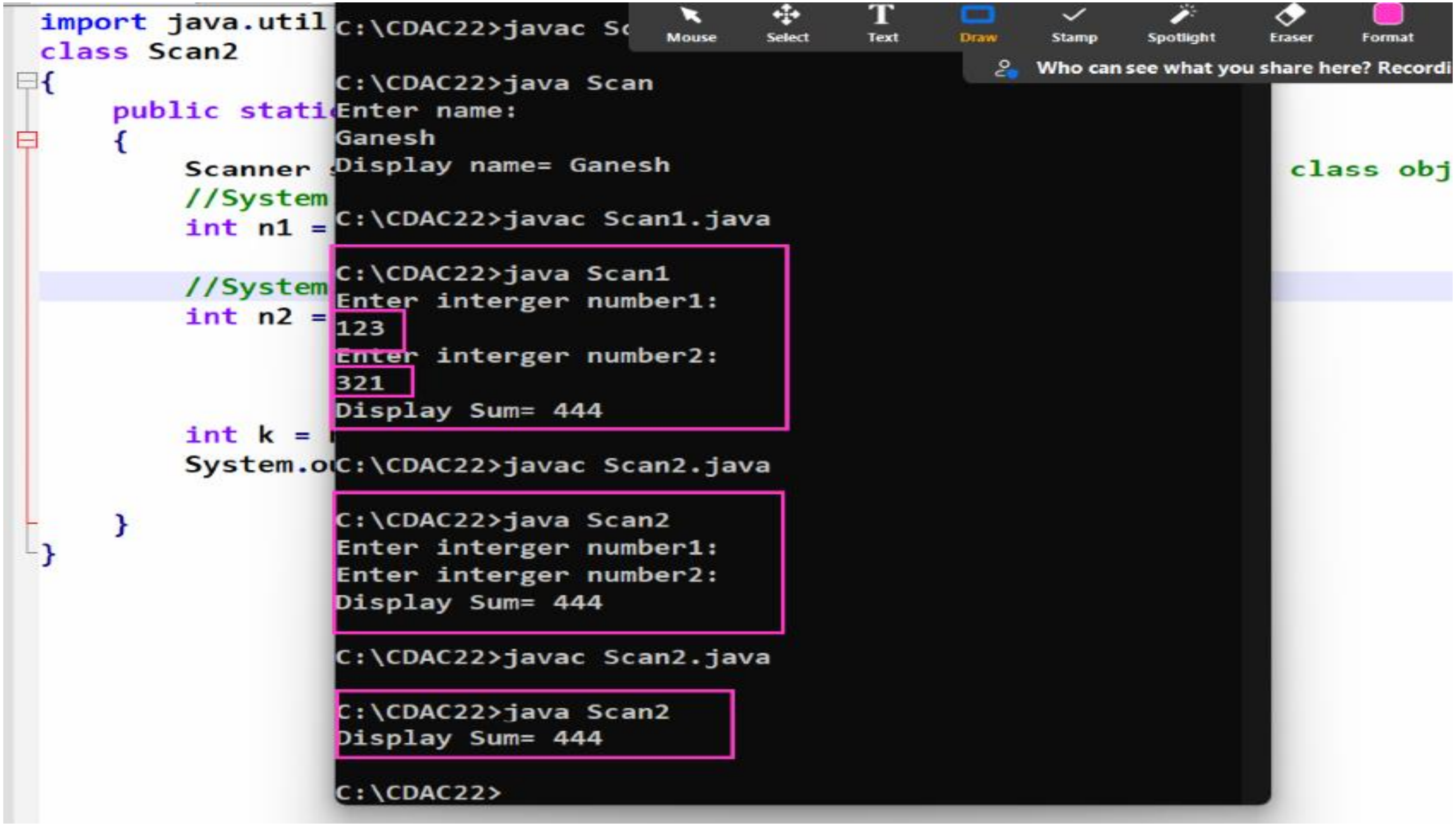
System.in

R I T U

read entered character

Scanner sc = new Scanner(System.in)

String name = sc.next();



```
import java.util
class Scan2
{
    public stati
    {
        Scanner
        //System
        int n1 =

        //System
        int n2 =


        int k =
        System.o

    }
}
```

```
C:\CDAC22>javac Sc

C:\CDAC22>java Scan
Enter name:
Ganesh
Display name= Ganesh

C:\CDAC22>javac Scan1.java

C:\CDAC22>java Scan1
Enter interger number1:
123
Enter interger number2:
321
Display Sum= 444

C:\CDAC22>javac Scan2.java

C:\CDAC22>java Scan2
Enter interger number1:
Enter interger number2:
Display Sum= 444

C:\CDAC22>javac Scan2.java

C:\CDAC22>java Scan2
Display Sum= 444

C:\CDAC22>
```

```
class obj
```

Operator Type	**Category**	**Precedence**
Unary	postfix	*expr*++ *expr*--
	prefix	++*expr* --*expr* +*expr* -*expr* ~ !
Arithmetic	multiplicative	* / %
	additive	+ -
Shift	shift	<< >> >>>
Relational	comparison	< > <= >= instanceof
	equality	== !=
Bitwise	bitwise AND	&
	bitwise exclusive OR	^
	bitwise inclusive OR	|
Logical	logical AND	&&
	logical OR	||
Ternary	ternary	? :
Assignment	assignment	= += -= *= /= %= &= ^= |= <<= >>= >>>=

Expressions

- Java provides a rich set of expressions:
  - Arithmetic
  - Bit level
  - Relational
  - Logical
  - Strings related

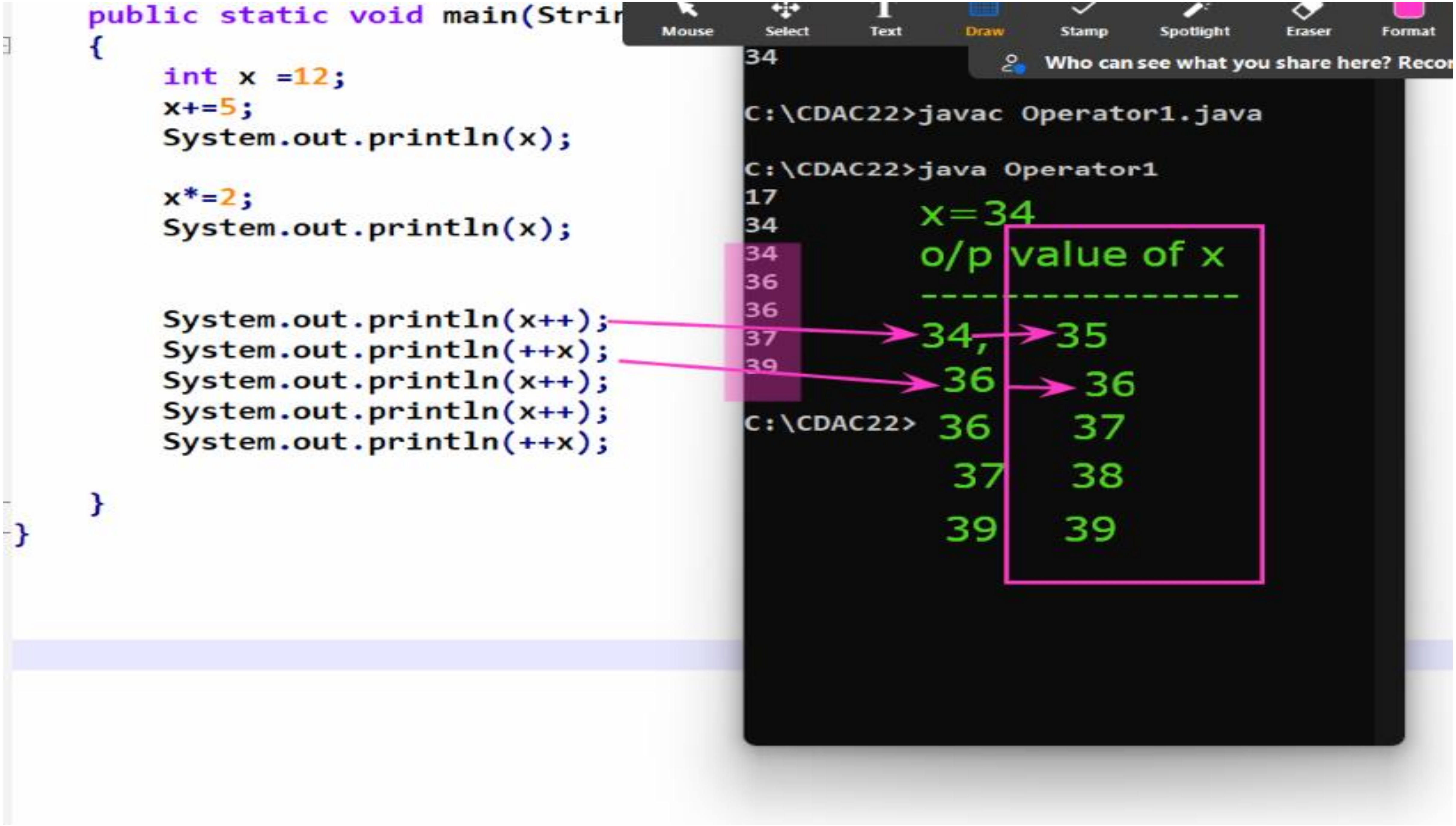
Arithmetic expressions

- Java provides the usual set of arithmetic operators:
  - addition (+)
  - subtraction (-)
  - division (/)
  - multiplication (‡)
  - modulus (%)

Relational expressions

- Java provides the following relational operators:
  - equivalent (==)
  - not equivalent (!=)
  - less than (<)
  - greater that (>)
  - less than or equal (<=)
  - greater than or equal (>=)

- Important: relational expressions always return a **boolean** value.



```java
    public static void main(Stri
    {
        int x =12;
        x+=5;
        System.out.println(x);

        x*=2;
        System.out.println(x);


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

    }
}
```

```
34

C:\CDAC22>javac Operator1.java

C:\CDAC22>java Operator1
17
34
34
36
36
37
39

C:\CDAC22>
```

x=34

```
o/p  value of x
-----------------
34, → 35
36  → 36
36    37
37    38
39    39
```

Bit level operators

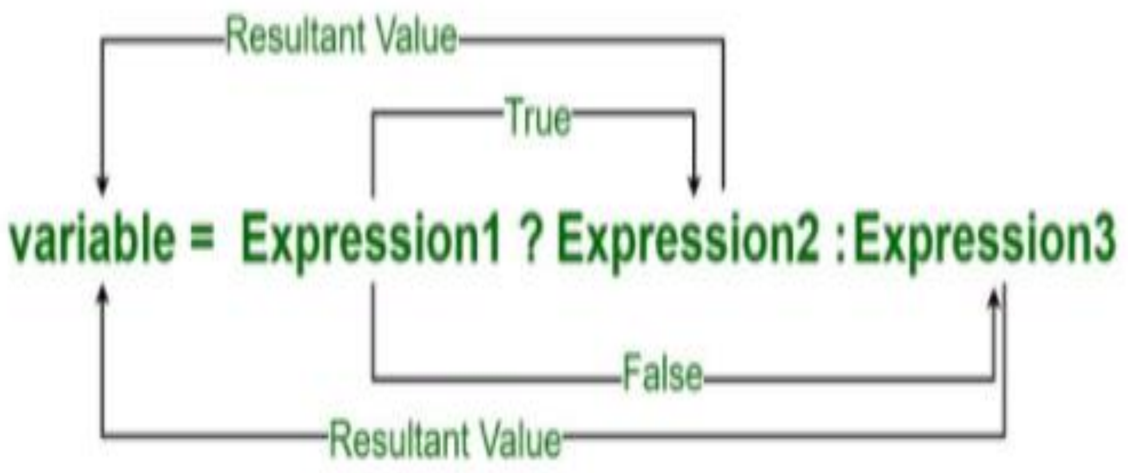
- Java provides the following operators:
  - and (&)
  - or (|)
  - not(˜)
  - shift left (<<)
  - shift right with sign extension (>>)
  - shift right with zero extension (>>>).
- ***Important***: **char**, **short** and **byte** arguments are promoted to **int** before and the result is an **int**.

Logical operators

- Java provides the following operators:
  - and (&&)
  - or (||)
  - not(!)

- ***Important***: The logical operators can only be applied to **boolean** expressions and return a **boolean** value.

Conditional or Ternary Operator (?:) in Java

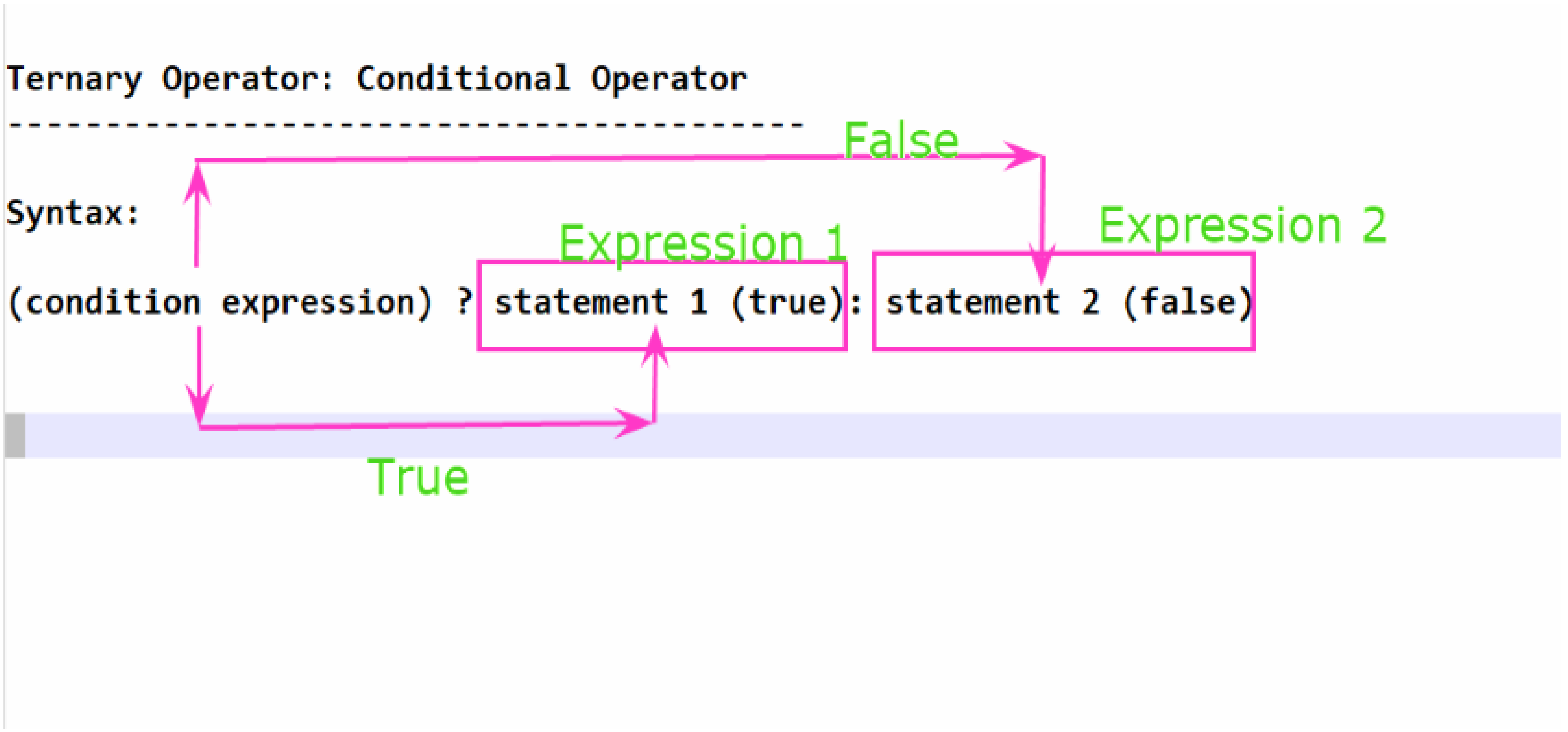


Syntax:

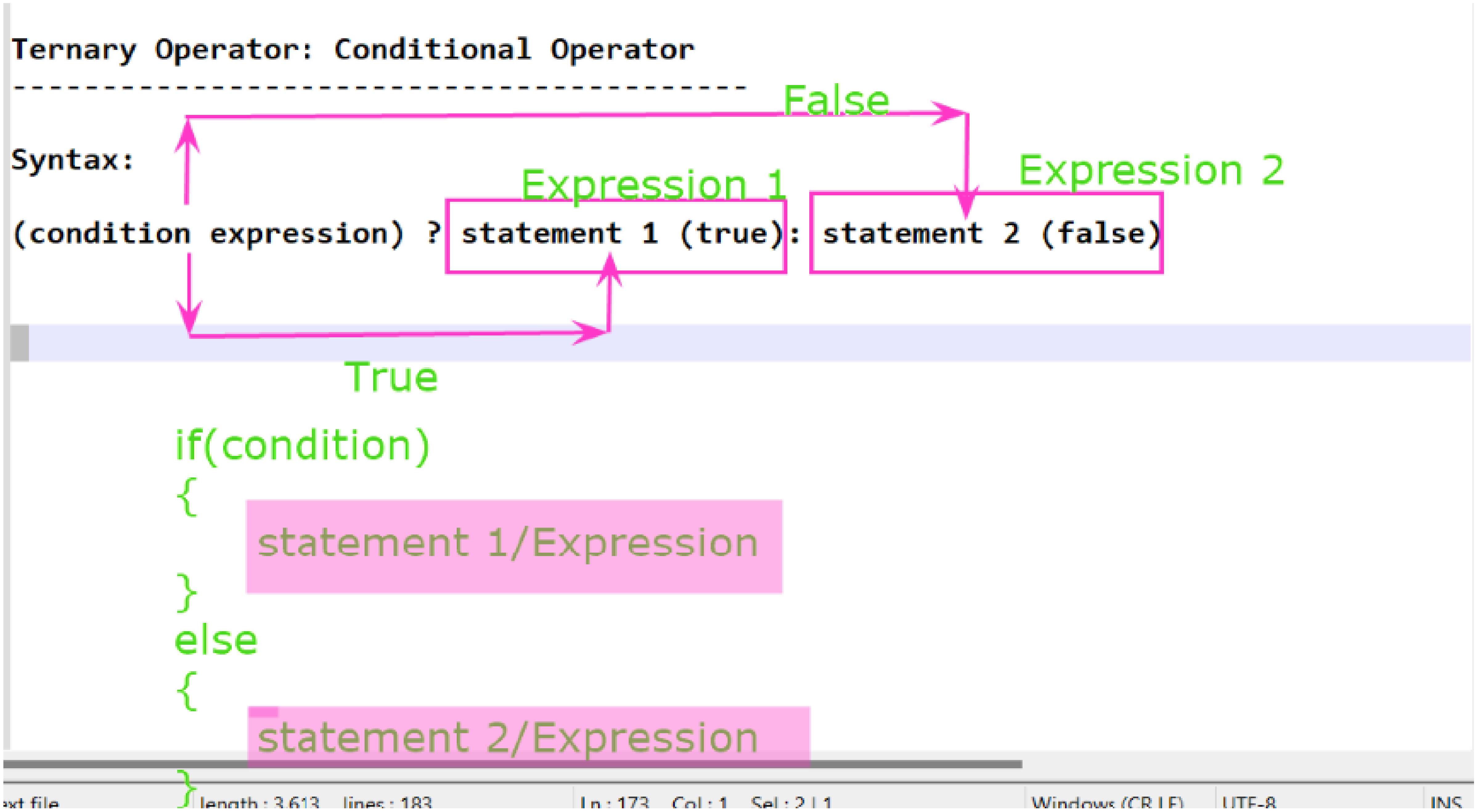
variable = Expression1 ? Expression2: Expression3

Or

```
if(Expression1)
{
    variable = Expression2;
}
else
{
    variable = Expression3;
}
```



```
Ternary Operator: Conditional Operator
-----------------------------------------

Syntax:

(condition expression) ? statement 1 (true): statement 2 (false)
```



```
Ternary Operator: Conditional Operator
----------------------------------------

Syntax:

(condition expression) ? statement 1 (true): statement 2 (false)
```

```
if(condition)
{
    statement 1/Expression
}
else
{
    statement 2/Expression
}
```

```
xt file   length : 3,613   lines : 183   Ln : 173   Col : 1   Sel : 2 | 1   Windows (CR LF)   UTF-8   INS
```