**Primitive data type in java :**

Example: int, short, long, float, double, etc.

**Wrapper Class in java:**

Integer, Short, Long, Float, Double etc.

Wrapper classes contains information about primitive data type …

**For Example**

a = Integer.MAX\_VALUE;

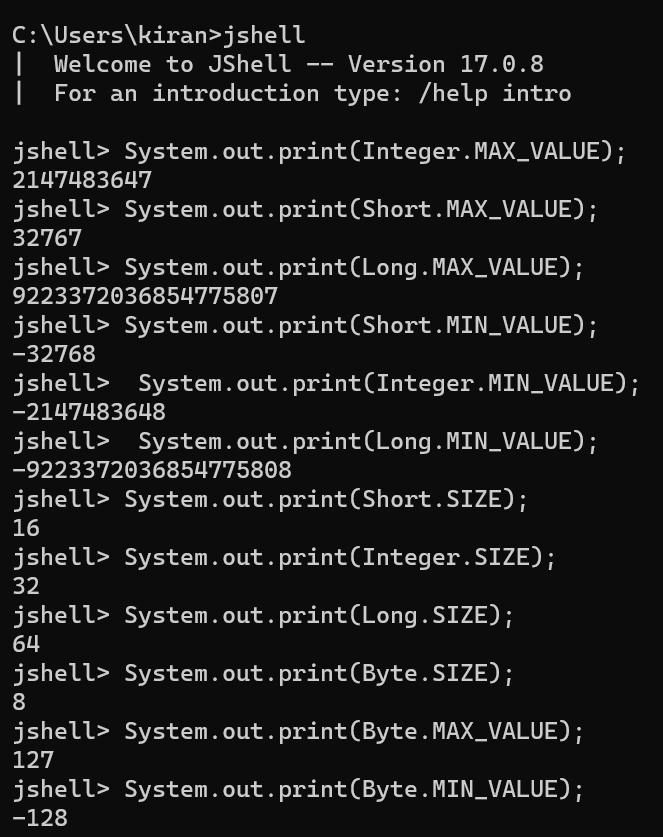
b = Integer.MIN\_VALUE;

c = Short.MAX\_VALUE;

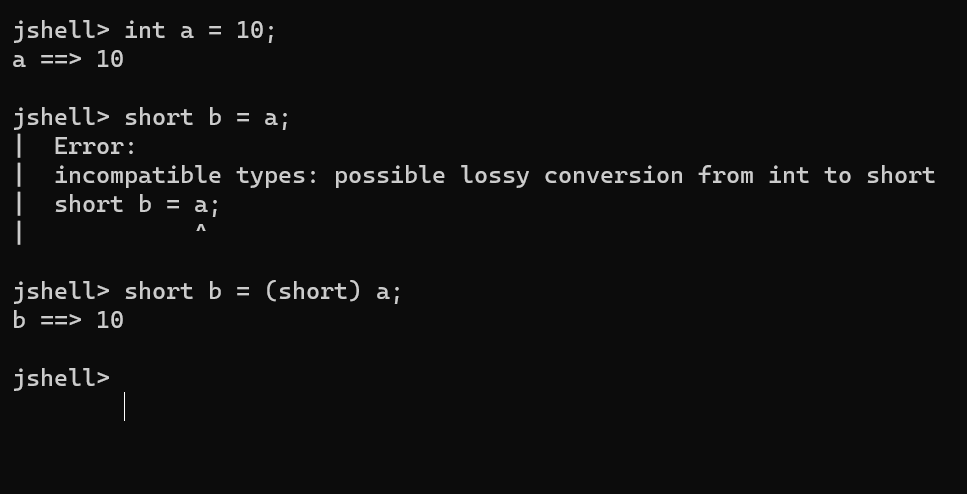
d = Short.MIN\_VALUE;

e = Long.SIZE (IT TELLS ABOUT THE WIDTH OR WE CAN SAY NO. OF SPACES IN LONG AND ITS 64..)

**Picture of above codes is below …**

****

**How to change data type of int to short :**

****

**Char class in java :**

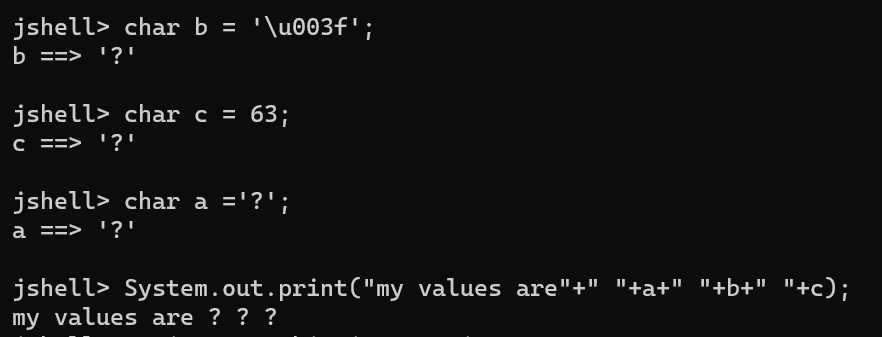
Here ‘\u003f’ is Unicode

Every character on keyboard as well as emojeys have Unicode ….

Characters also have integer values assign to them….

See the below picture for easy understanding

Note: Unicode is typed by back-slash…

****

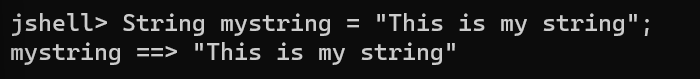
**Char VS String:**

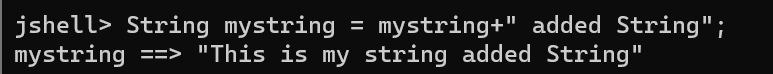
Char stores single character

String stores multiple characters …

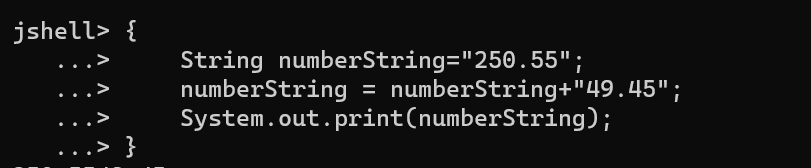
Char is useful when a single character is pressed on keyboard while playing the game.

**String:**





You can also type codes in multiple lines …or type all the code in single line



Note string is immutable (means u cannot make changes to sting )

In java if u add new string to current string … it will create new variable to store the string …

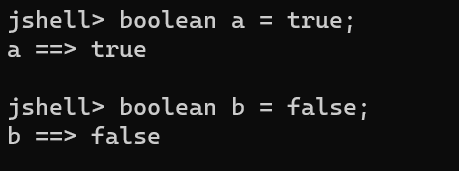
But we have another better to play with strings …. That is String Builder

String Builder is mutable method …(you can make changes to string)….

We will discuss it below, but before that we will discuss Boolean data types:

**Boolean**

Boolean class can only store 2 types of data …that is true or false…



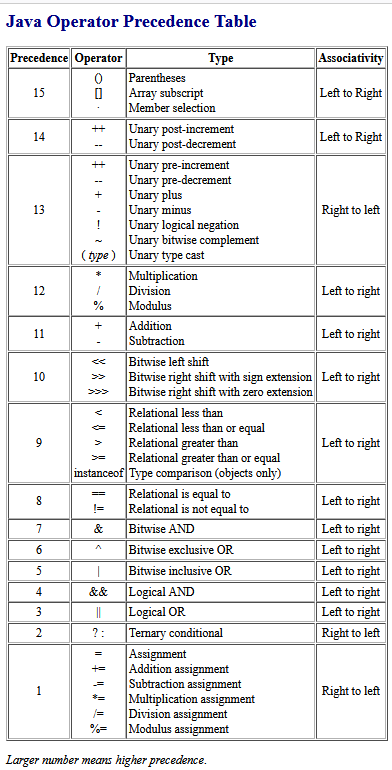
**Codes:**

1. **Hello World (explanation in code folder)**
2. **And Operator (explanation in code folder)**
3. **Or Operator (explanation in code folder)**
4. **Ternary Operator (explanation in code folder)**

**Operator Precedence and Operator Challenge:**

**Java Operator Precedence Table:**

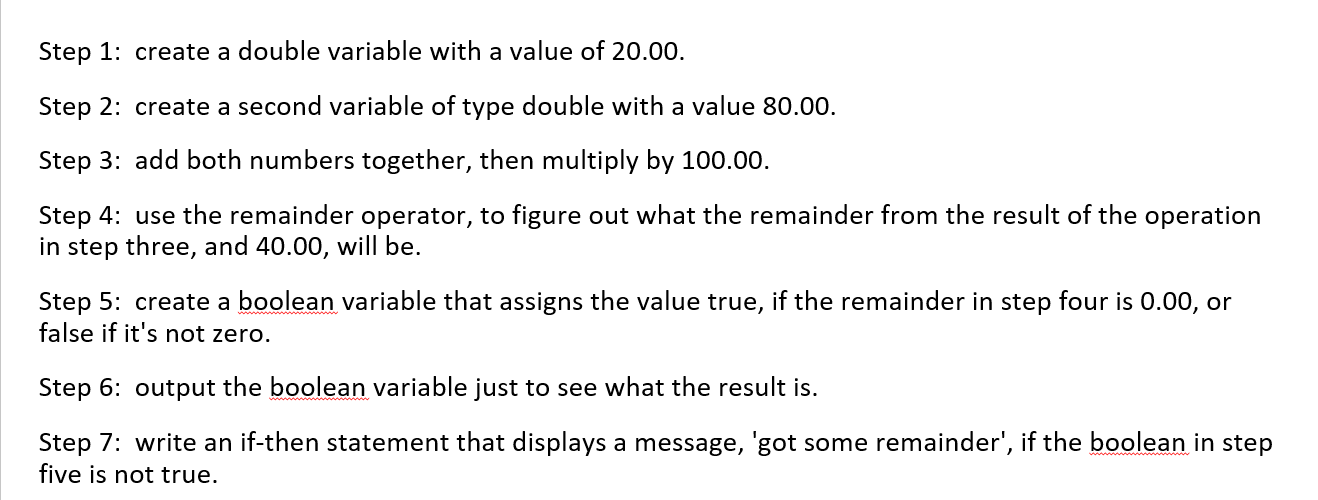
It means which operator has higher priority … and which operator will work first …



You can search on google : Java Operator Precedence Table.

Open first link … and you can see the above table more clearly…

**Challenge**

****

**Challenge Code:**

1. **Operator Precedence Challenge**

**Keywords, Expressions and Statements:**

**Keywords:** key words are pre defined words in java … for example int, float, double, for etc ..

There are more keywords in java u can check that on google…. We cannot use these keywords as variable names… point to be noted.

**Expression:** An expression computes to a single value . (asaan bhasha me)

An expression contains variables, values and operators.

For example: int a = (10+2)\*2;

In above code a = (10+2)\*2 is expression.

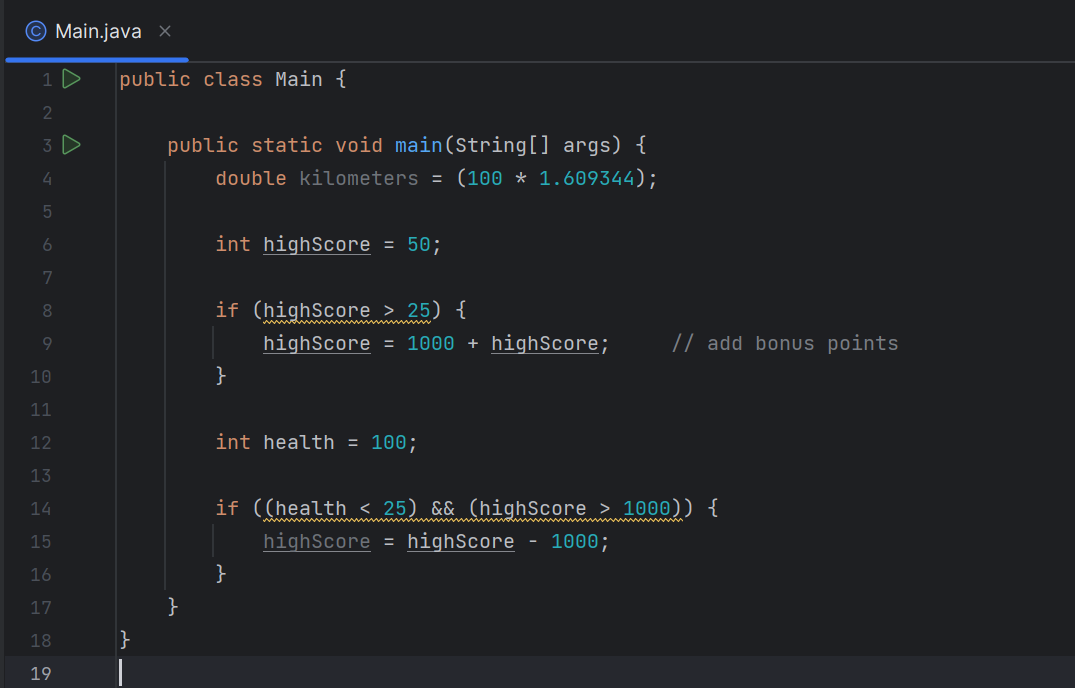
And full line of code is called statement. int a = (10+2)\*2; (this is statement)

**Statements:** Statements are stand alone units of work. (asaan bhasha me) full line of code is statement explained above.

**Code Blocks:** A code block is a set of zero, one, or more statements, usually grouped together in some way to achieve a single goal. (asaan bhasha me) group of code … or group of statements .

**Task 1 :**

Tell the difference between keywords, expressions and statements in below code image …

****

**Code:**

1. **Keywords and Statements Code**

**White Spaces and Indentations**

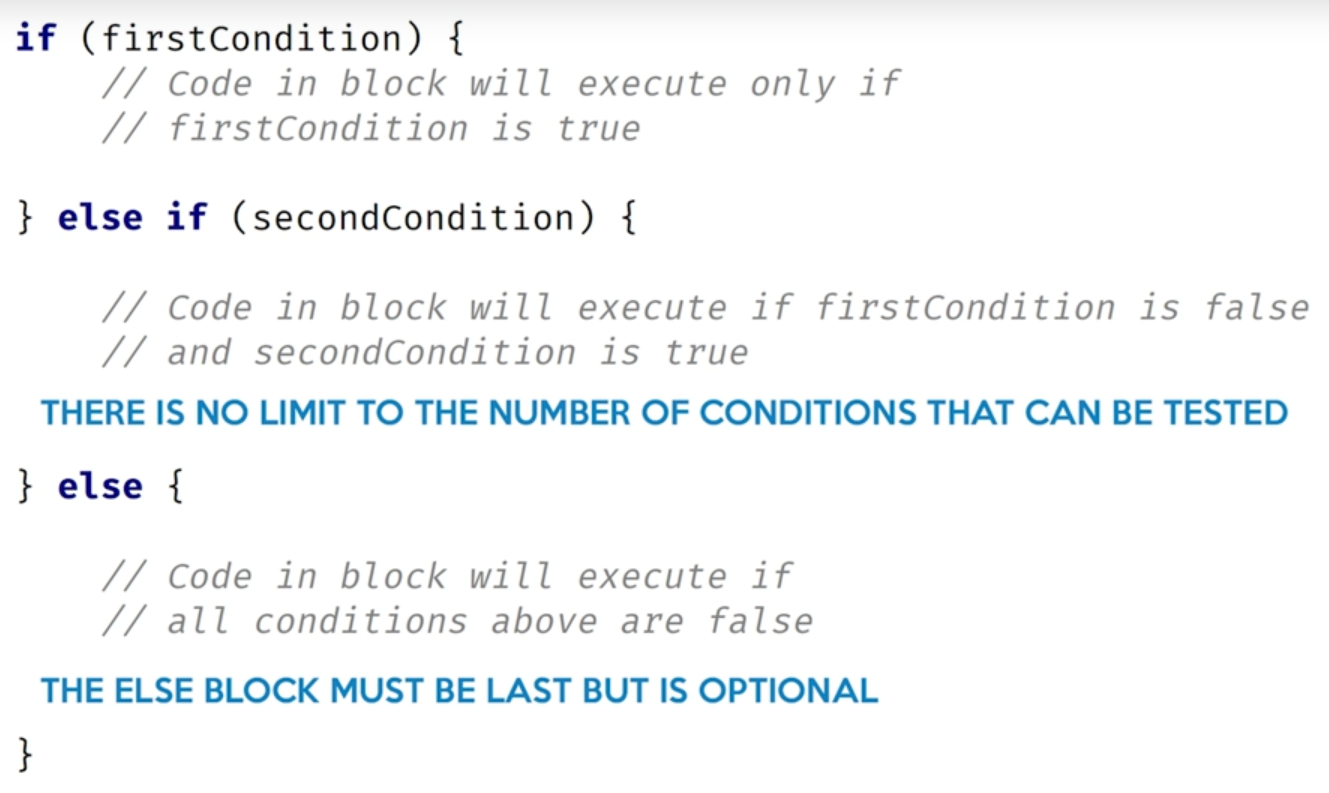
White spaces can be used in java horizontally as well as vertically .. so that code could be more readable … java ignores white spaces..

**Indentations .**

Indentations are in java so that code could be more readable ..

Intellij idea have feature to reformat code … so it will automatically add indentations to code …

**If, else if, else :**

****

**Code:**

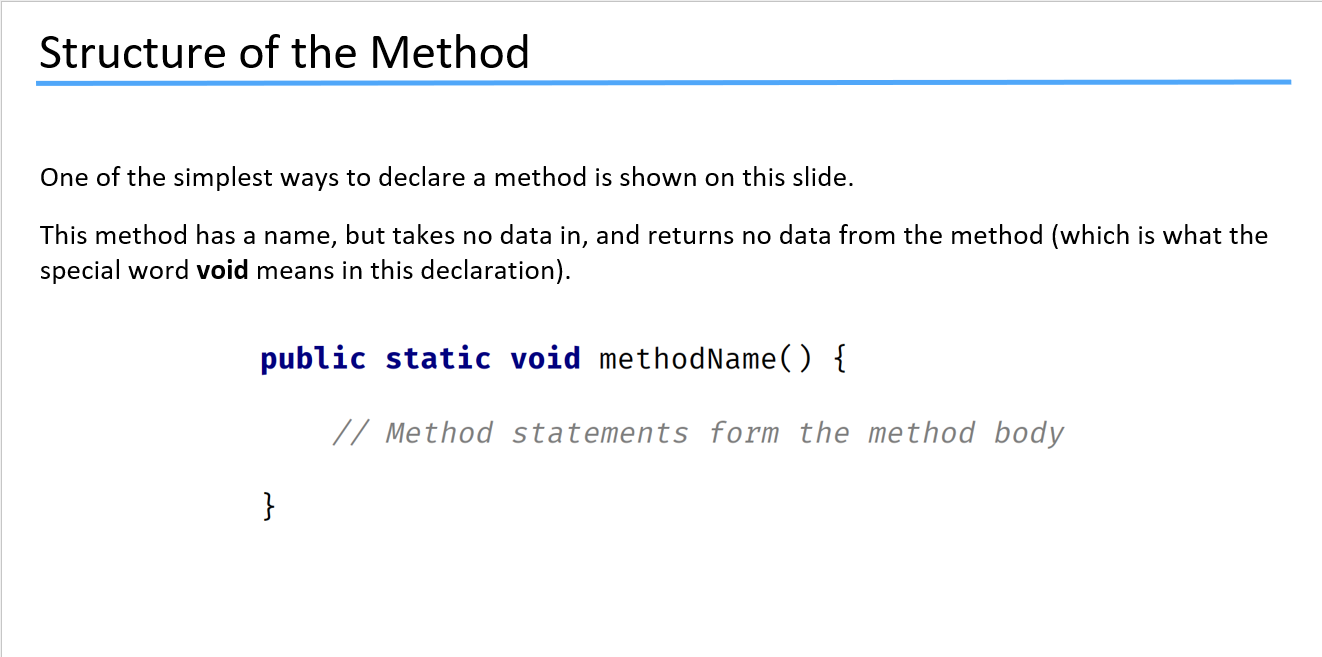
1. **If Else Code**
2. **If Else Challenge**

**Methods in Java:**

Methods are created in java to save time …

Methods are type of code block in java … you can call them n number of times in the code

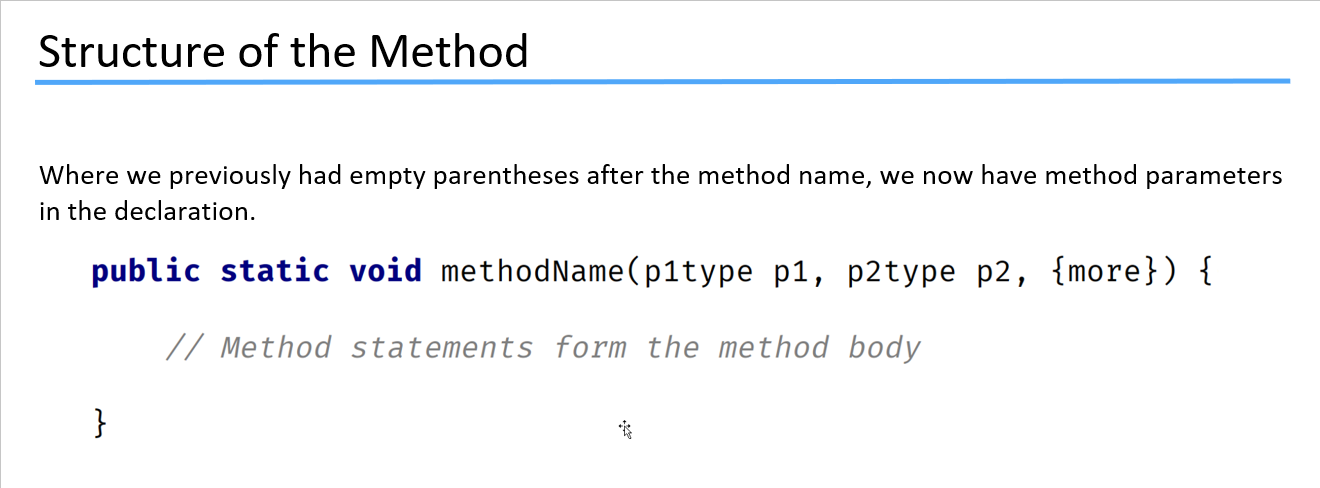
This will save time …



This is how we declare method in java …

**Public static void** methodname**( ) {**

**}**

****

**This is the structure of method …**

**Now lets do the hands on …**

For better understanding of topic see code no. 9 method declaration…

**Code:**

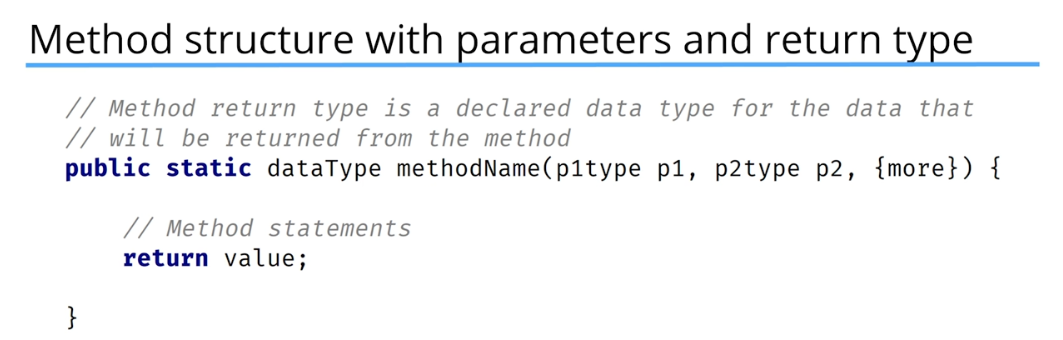
1. **Method Declaration**

**Another way of Method Declaration**

**Code:**

1. **Method Declaration Another**

**Method Declaration using the return method :**

****

then we have to print the method to see its value …

System.out.print(methodName(value, value));

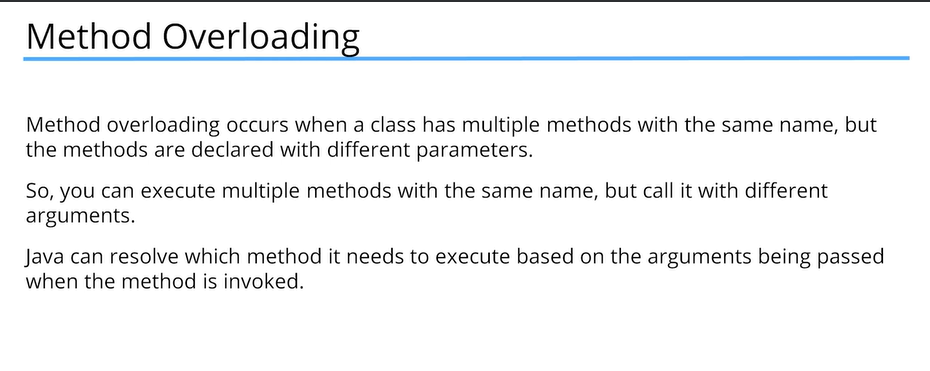
**Code:**

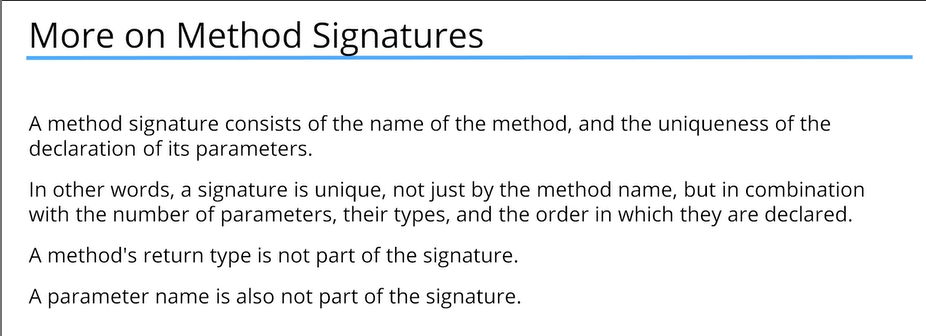
1. **Method Declaration of Simple Interest**

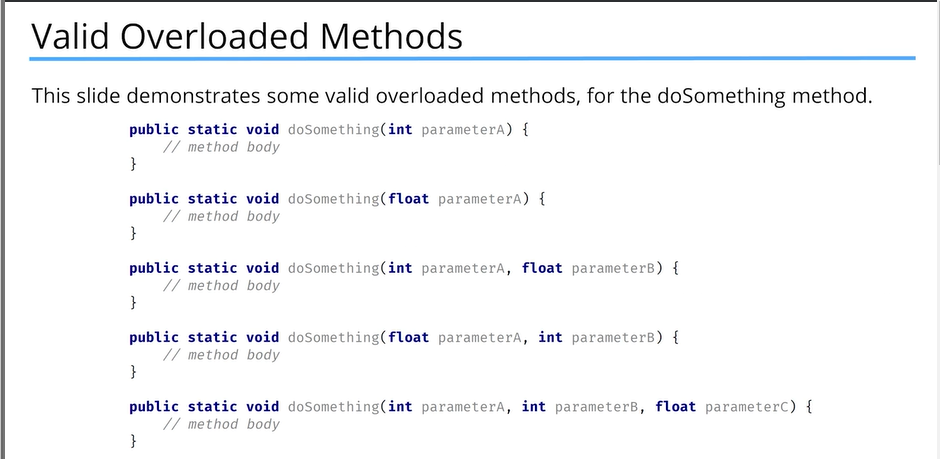
**Do Questions on Methods in Java Programming Master Class Questions folder:**

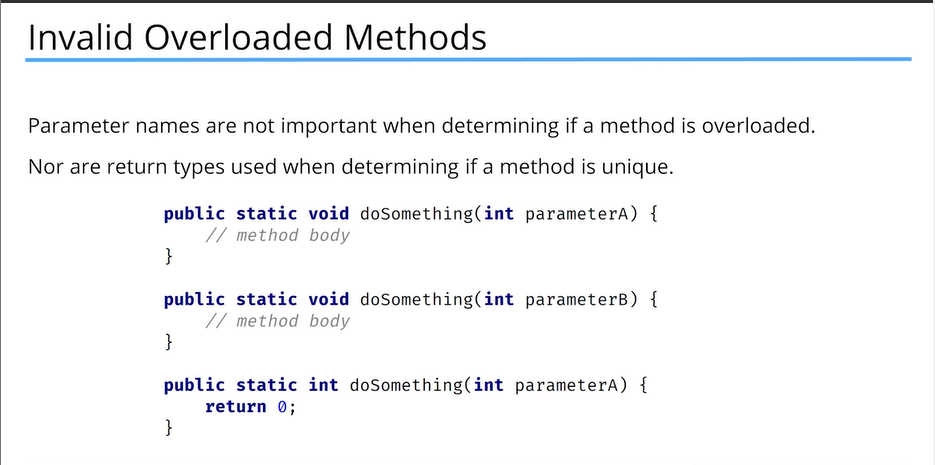
1. **Method Questions**

**Overloading of Methods:**

****

****

****

****

**Code:**

1. **Method Overloading**

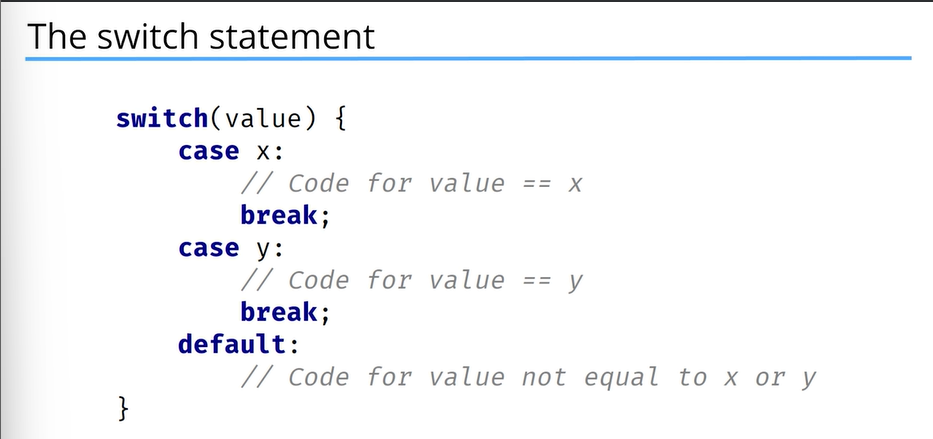
In other languages we can define default parameters in methods but in java we can not … **but there is a way … I will show you how ?**

**Code:**

1. **Default Parameter in Method**

**Code:**

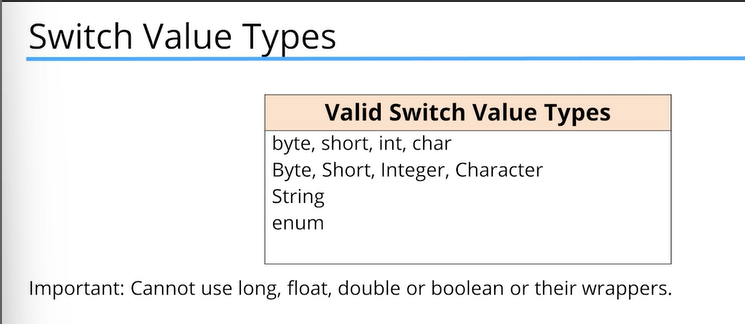
1. **Total Height in cm**
2. **Seconds and Minute Challenge**

****

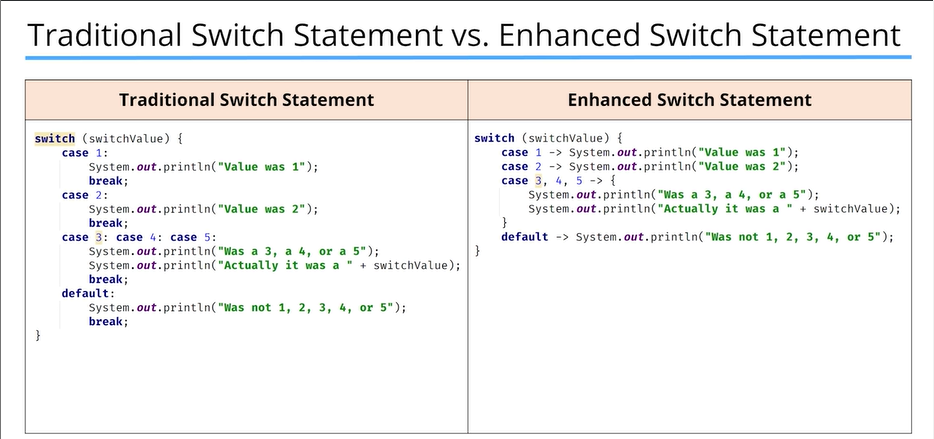
break; is used to get out from the switch statement after the execution if value matched with the case.

**Code:**

1. **Switch Statement**

****

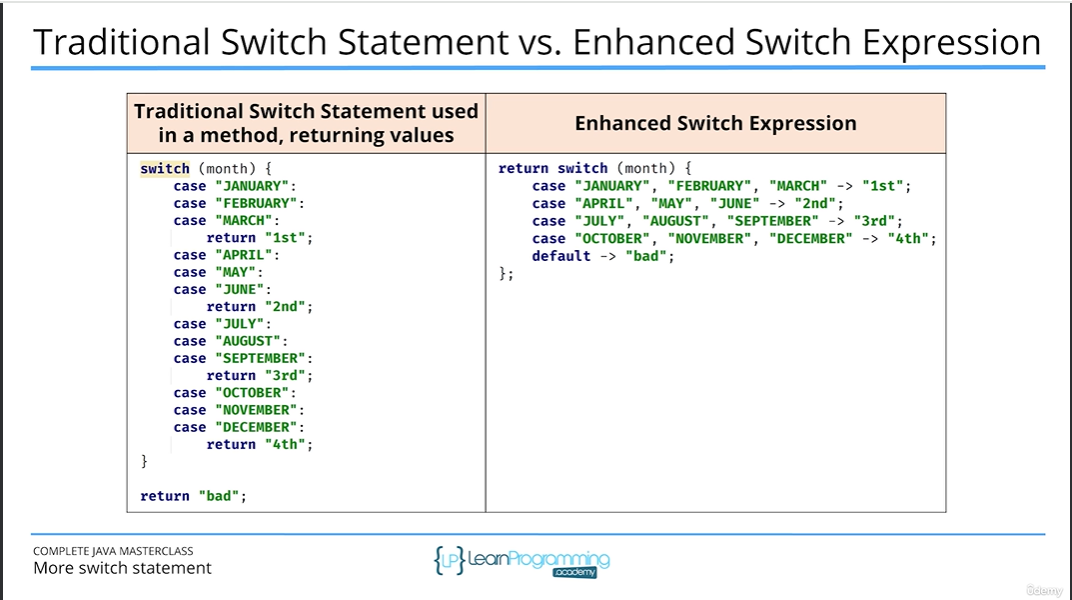
**Traditional VS Enhanced Switch Statement:**

****

There is no need to use break; statement in enhanced switch statement and Enhanced Switch Statement is more readable

Both Switch Statements return the same output. It is just the matter of style.

**Method of Traditional Switch Statement VS Enhanced Switch Statement:**

****

**In enhanced switch expression if you want to return any mathematical operation result so you need to use yield to return the output instead of using return.**

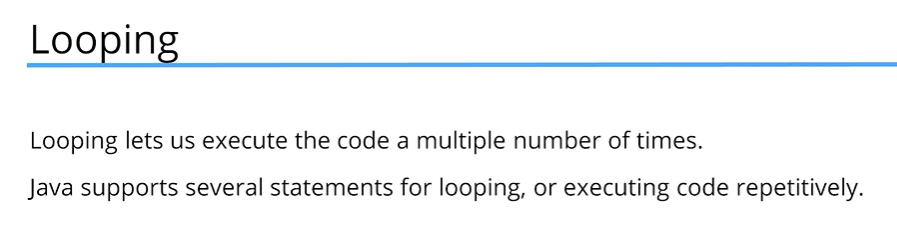
**Code:**

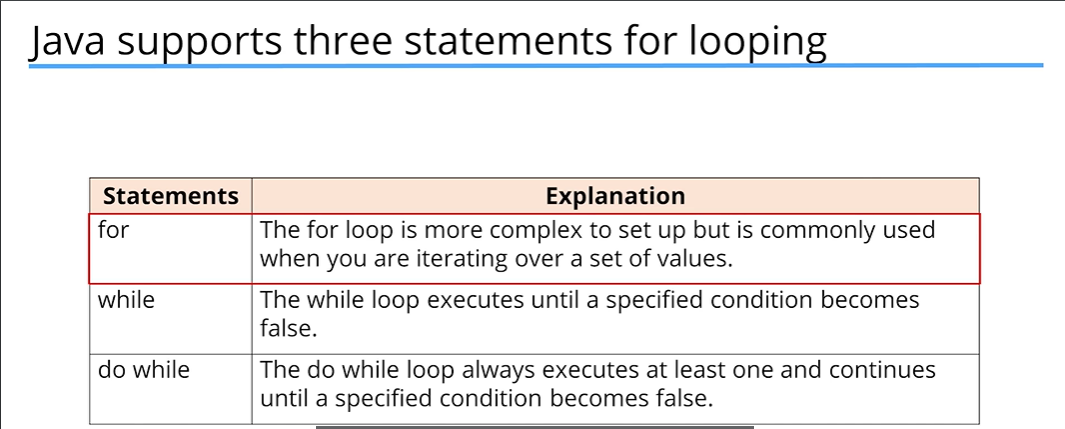
1. **Enhanced Switch Statement**
2. **Method of Enhanced Switch Statement**

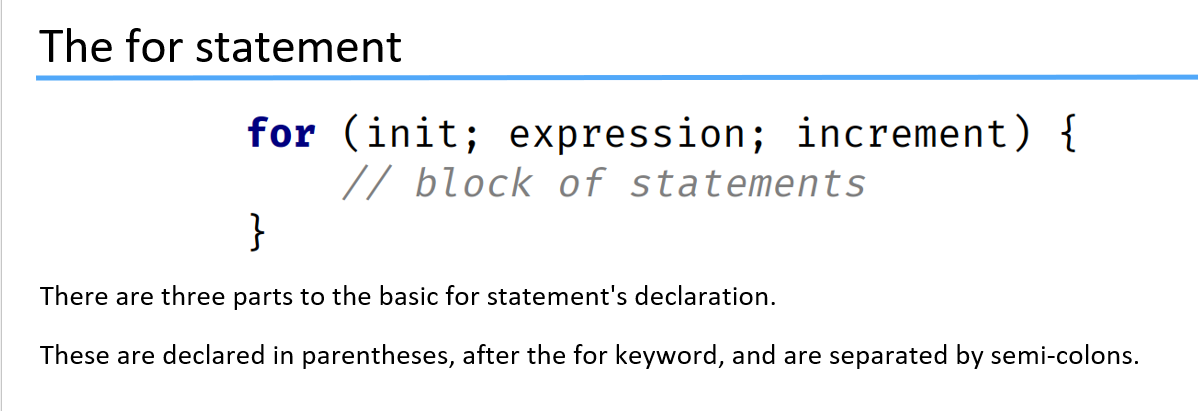
**Code:**

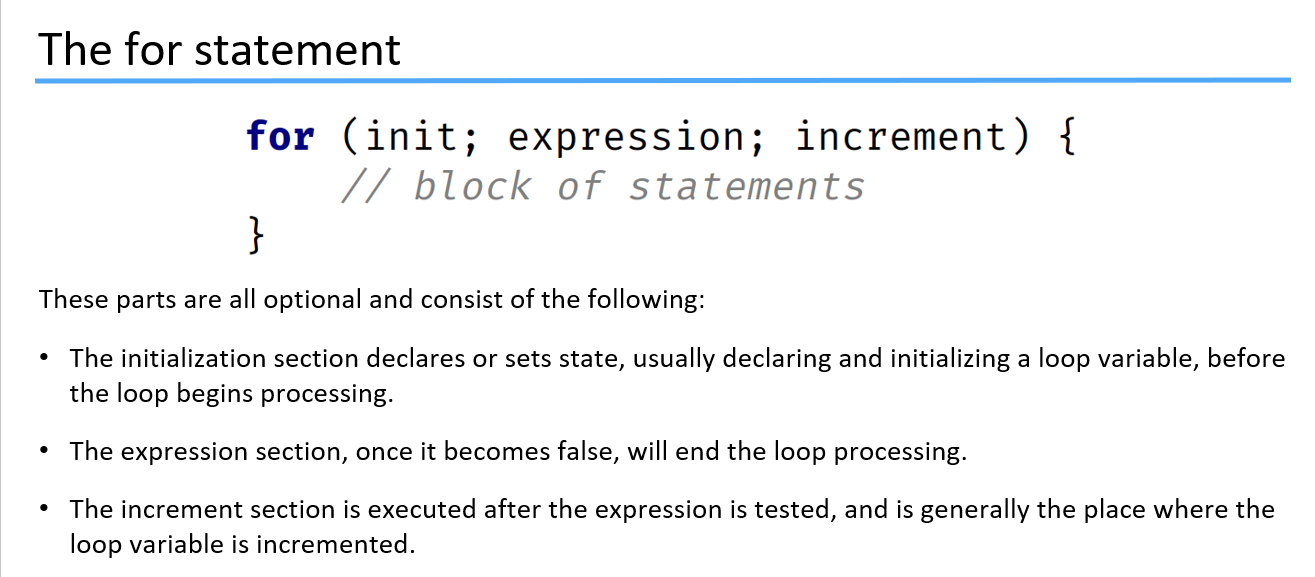
1. **Traditional Switch Challenge**
2. **Method Enhanced Switch Statement Challenge**

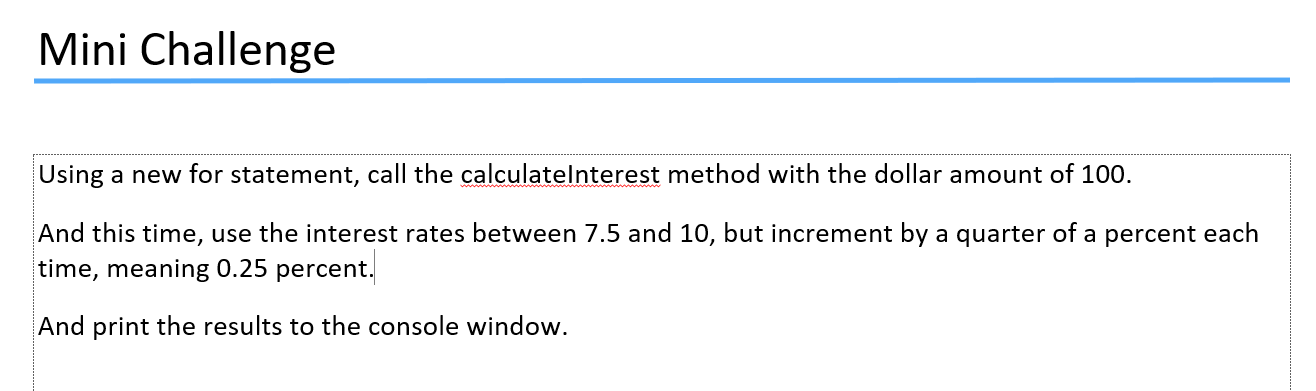
**Loops:**

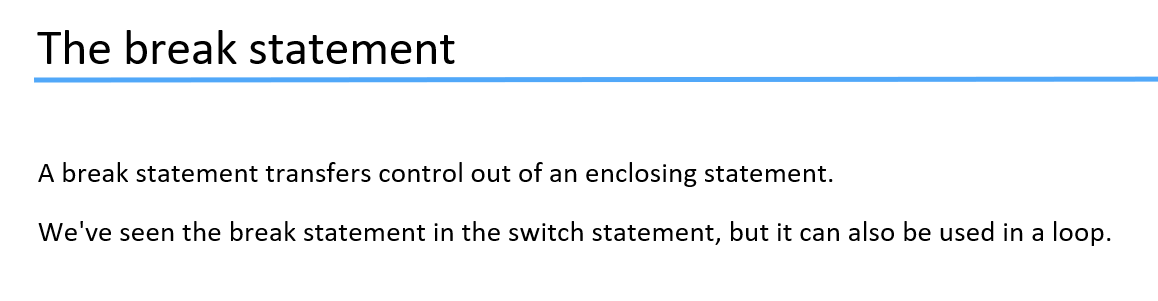
****

****

****

****

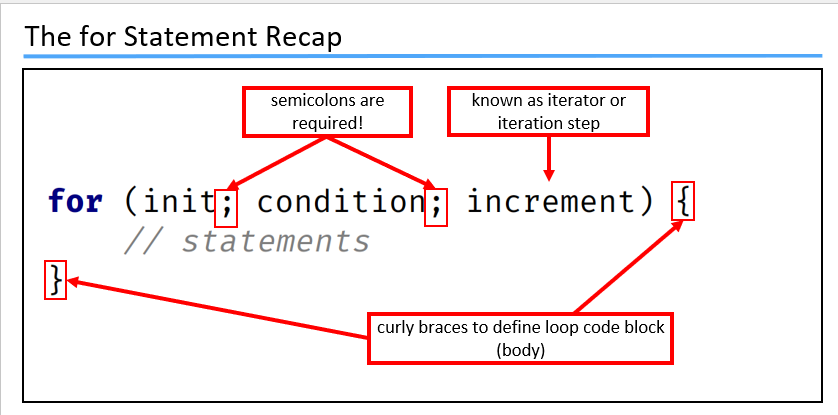
****

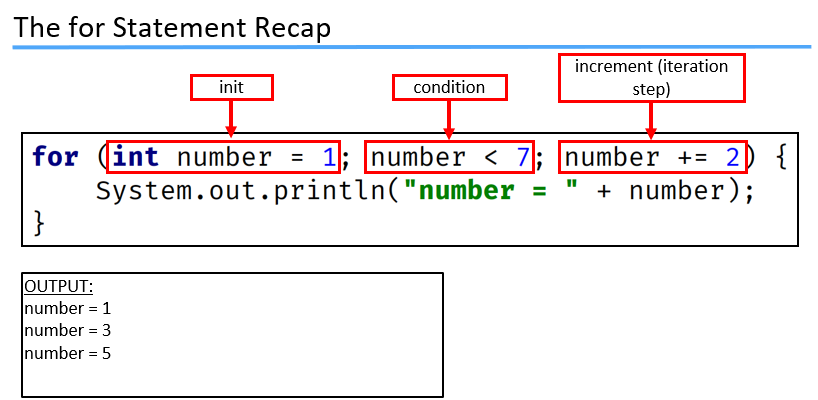
****

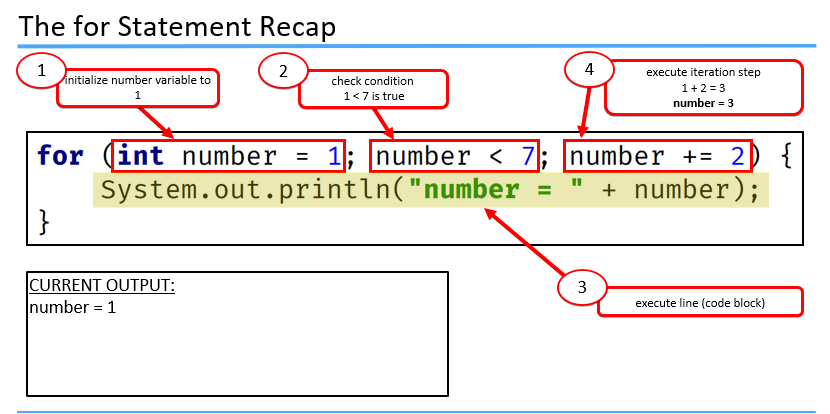
**Code:**

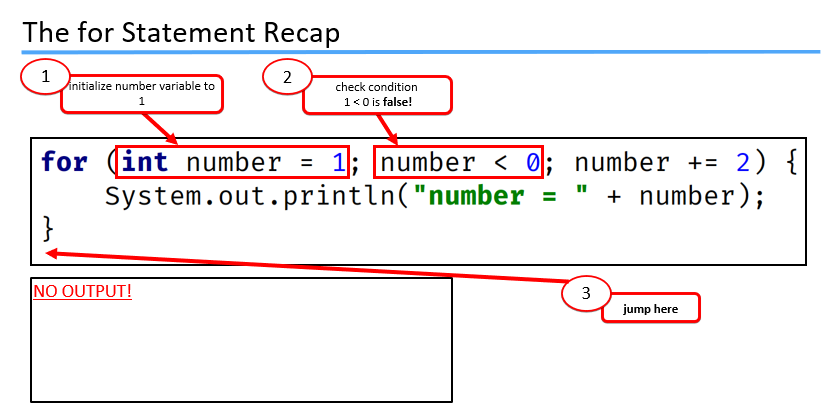
1. **Simple for loop**
2. **Calculate Interest for loop**
3. **Method to check Prime Numbers**
4. **Prime Number Challenge**
5. **Sum 3 and 5**
6. **40 + 40=100(Gym)**

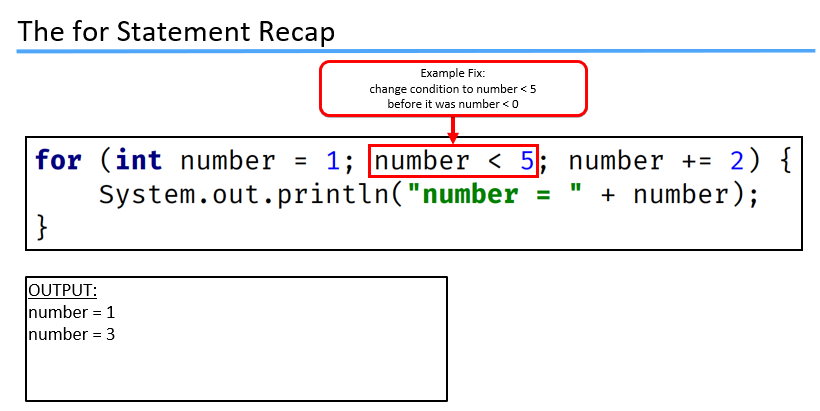
Use debugger to debug the code and find errors…

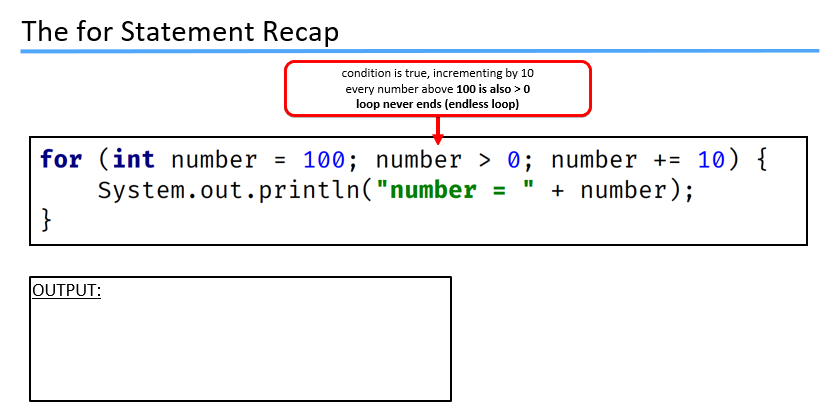


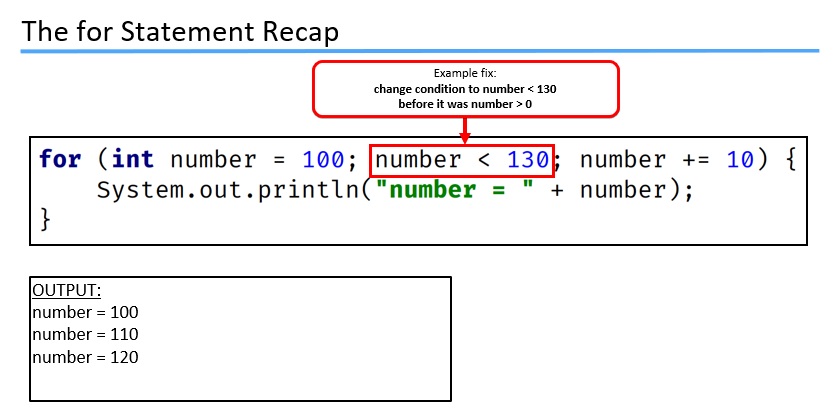


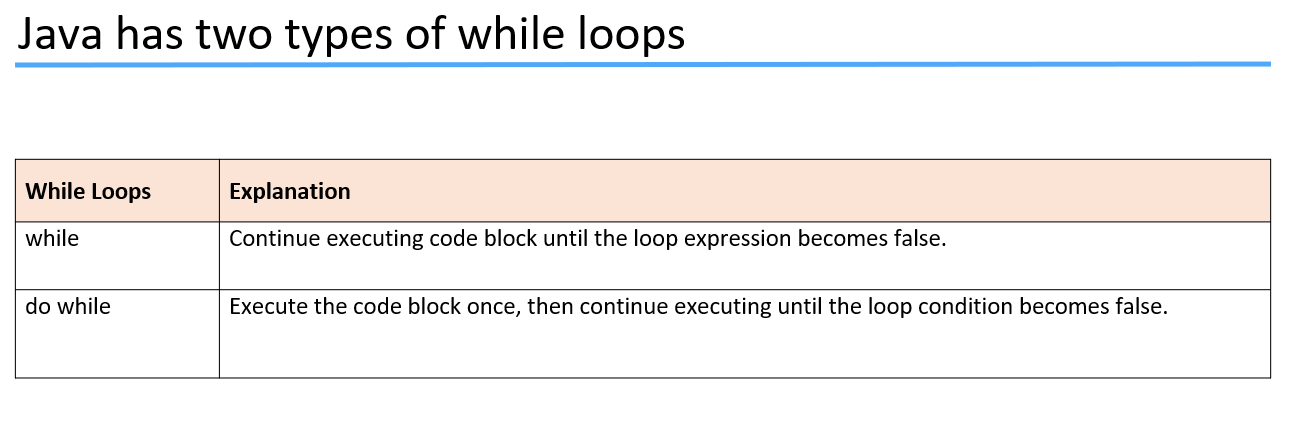


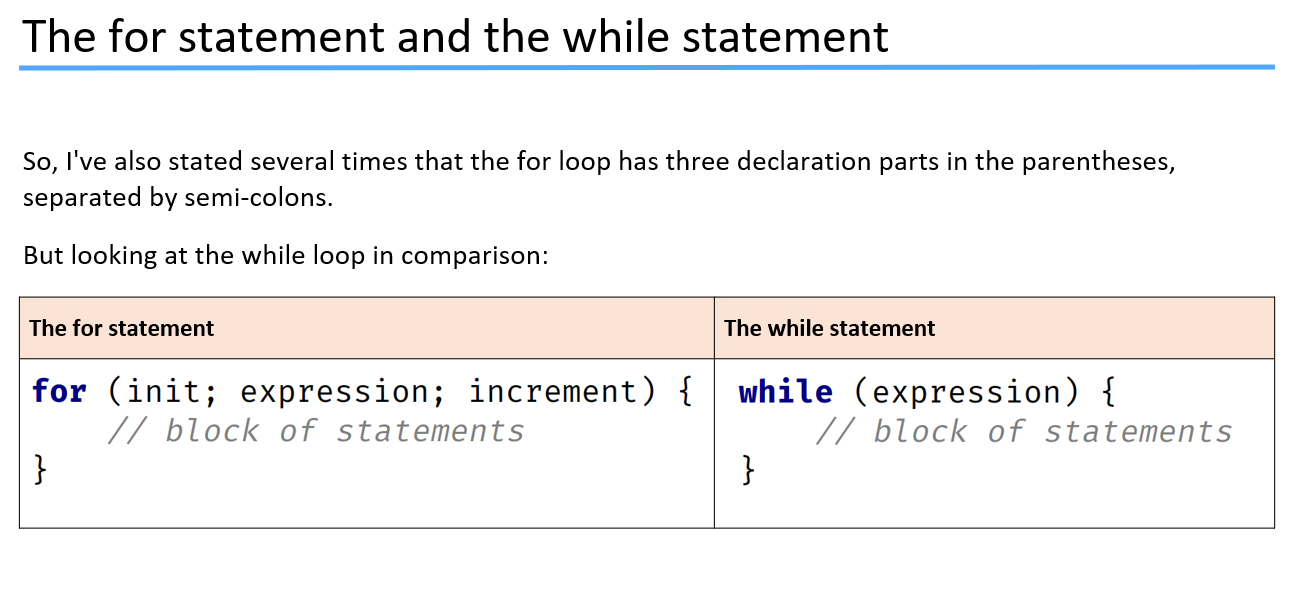


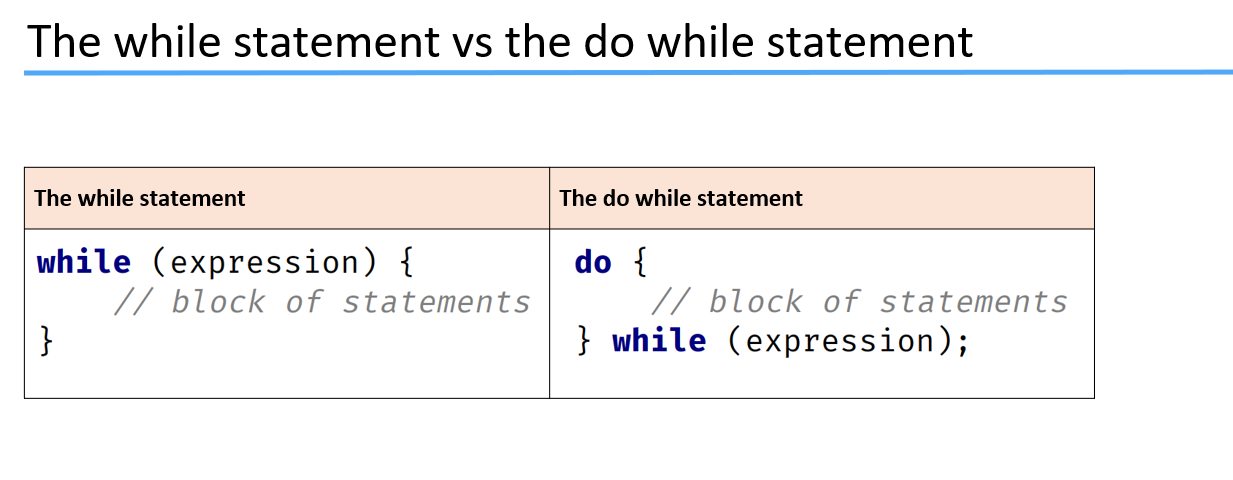


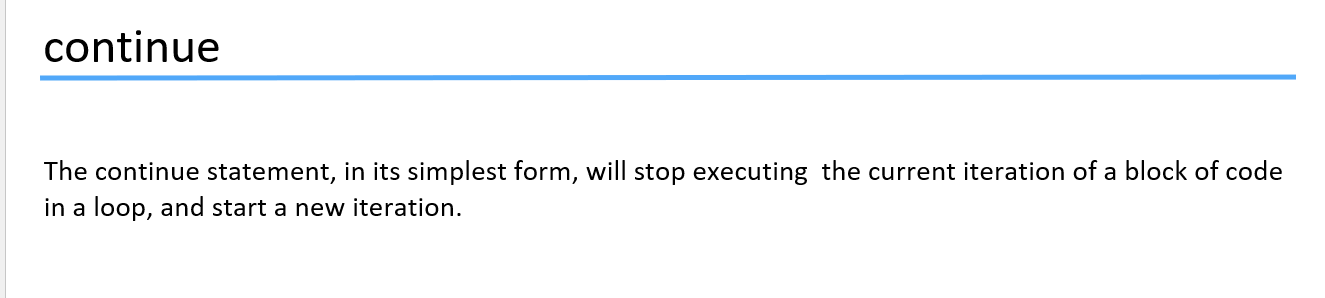


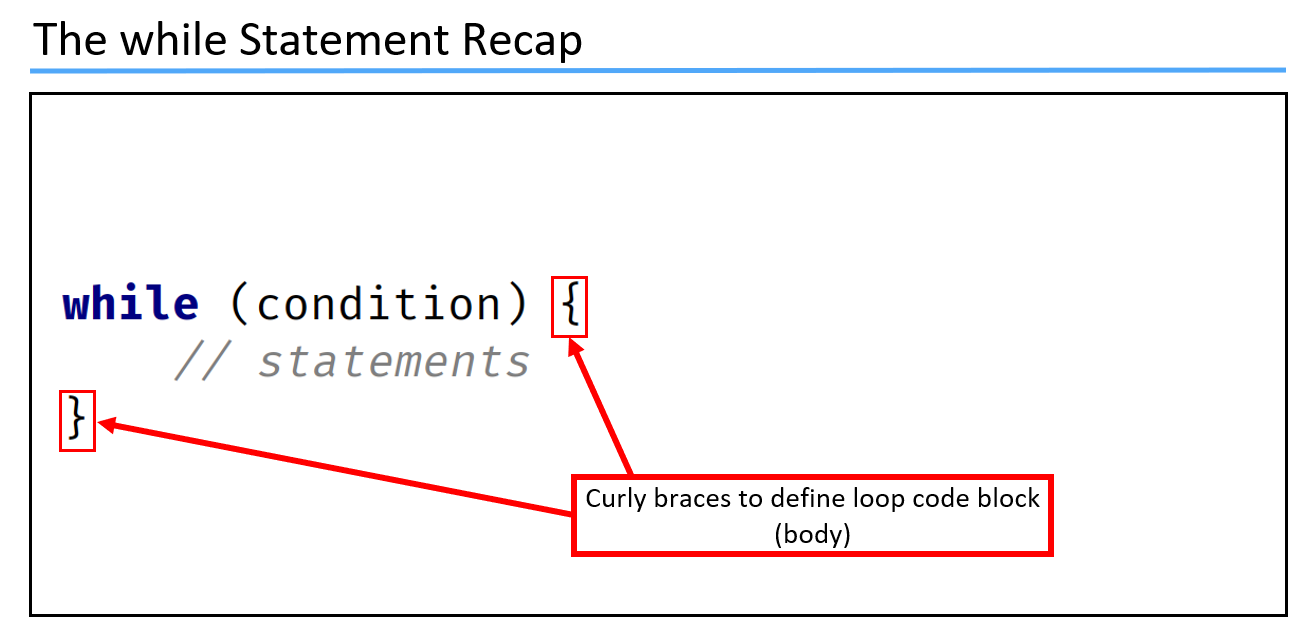


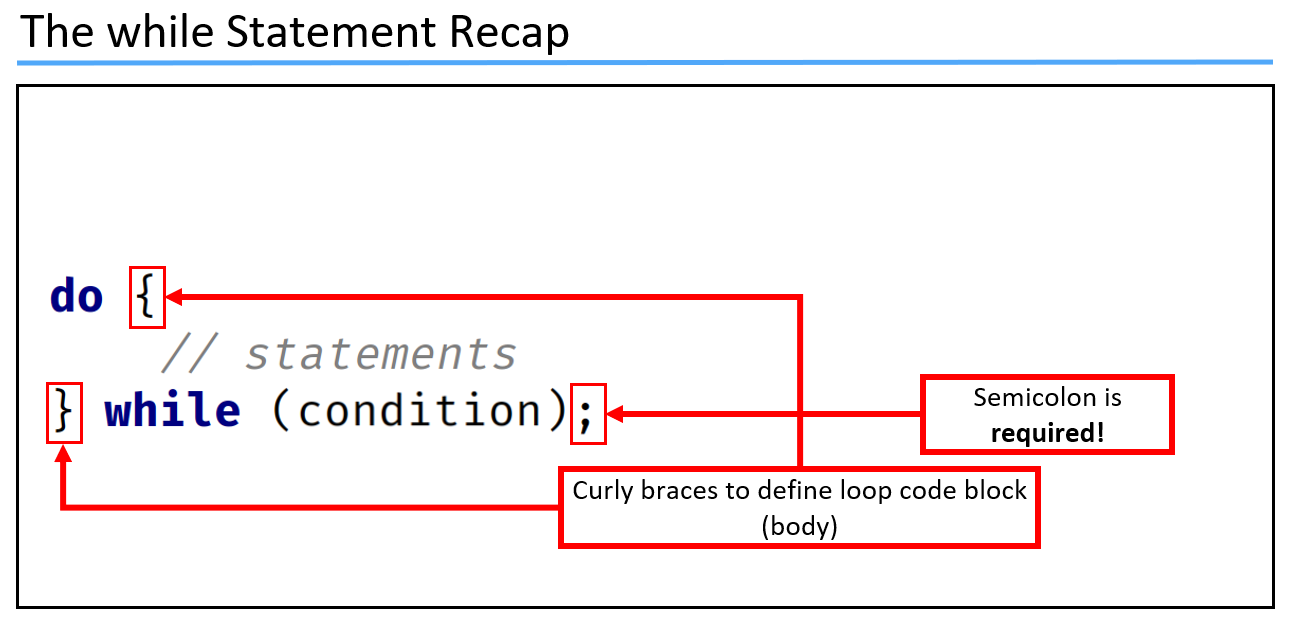


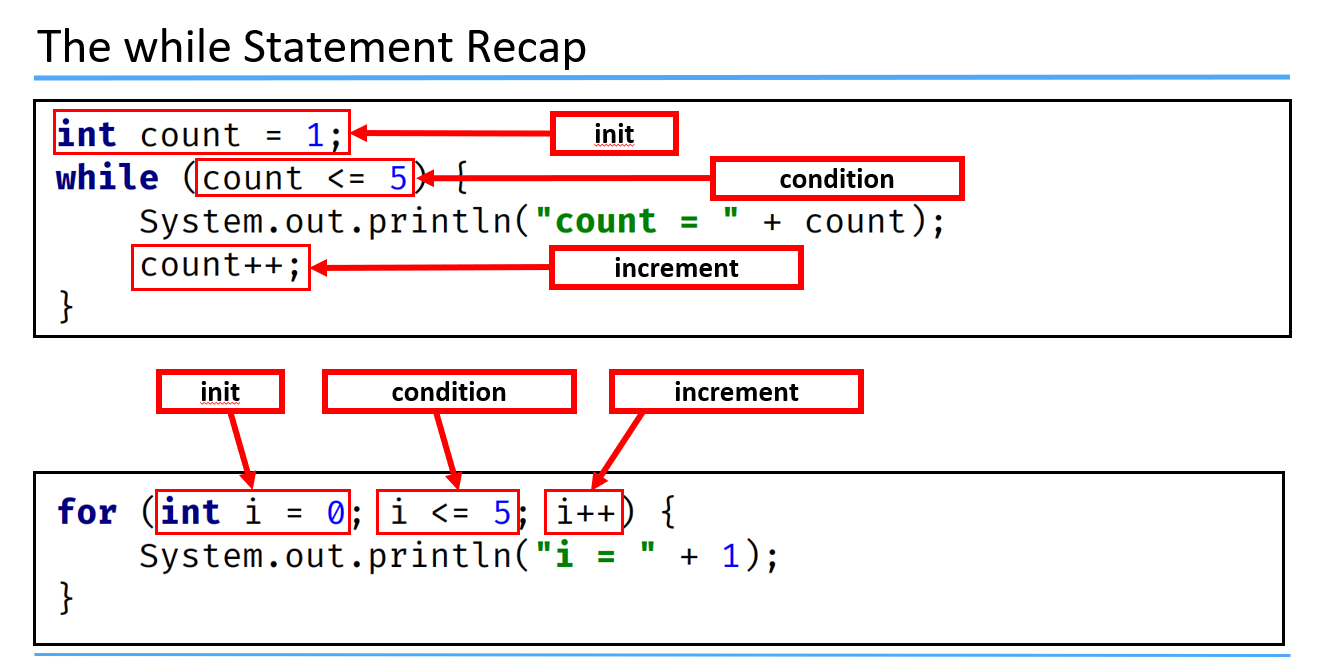


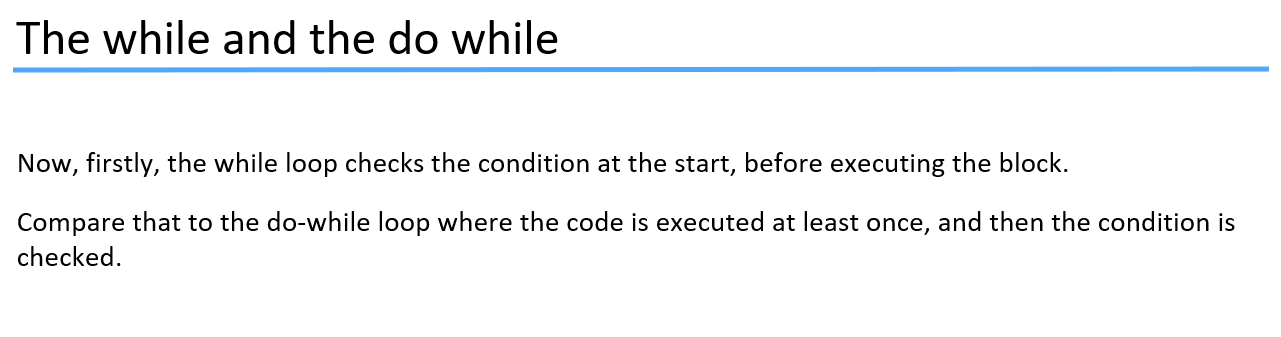


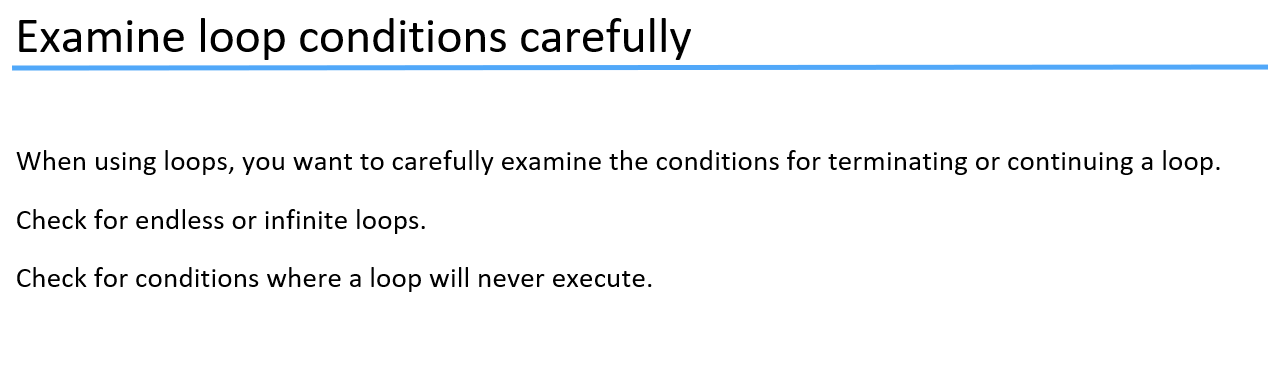


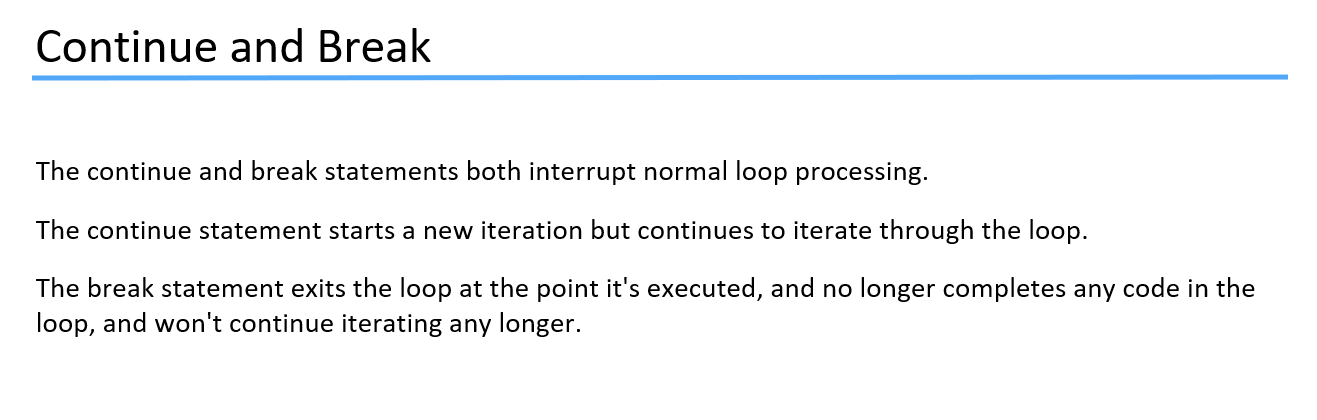












**Code :**

1. **While Loop Example**
2. **While Loop Another Example**
3. **Do While Example**
4. **Continue Statement**
5. **Print Even Numbers Using While Loop**
6. **Digit Sum Challenge**
7. **Method to check positive negative or zero**
8. **Method to convert Kilobytes to Megabytes**
9. **Are Equal by three Decimal Spaces**
10. **Check Palindrome**
11. **Sum First Digit and Last Digit of number**
12. **Even Digit Sum**
13. **Shared Digit**
14. **Same Last Digit**
15. **Print Factors**
16. **Print 3rd last consonant in a string**
17. **For loop print 3rd last consonant in a string**
18. **Greatest Common Divisor**
19. **Perfect Number**
20. **Largest Prime Factor**
21. **Diagonal Star Pattern**
22. **Number to Words**

**How to convert String input to Integer and Double …**

**Code :**

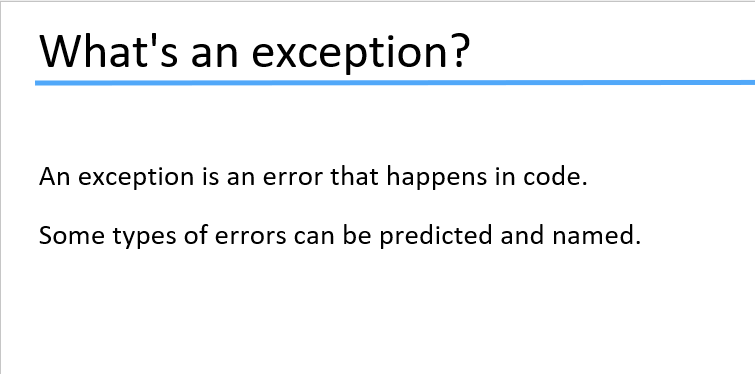
1. **Parsing Values**

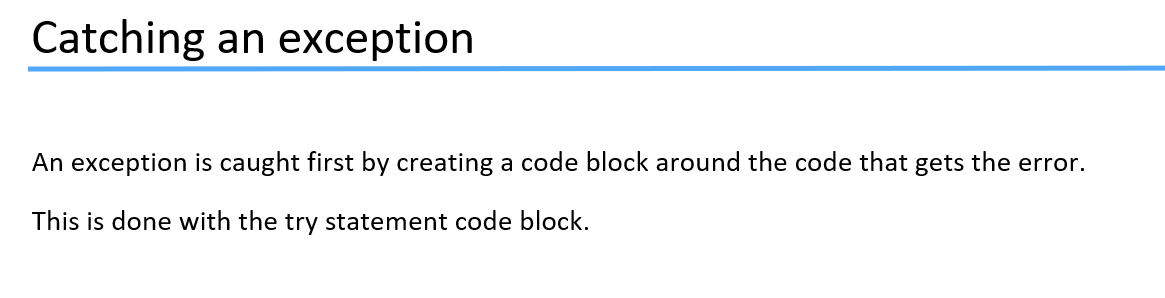
**Below code will tell you. How to take String input from user using System.console().readLine("Enter Number : ");**

**Code:**

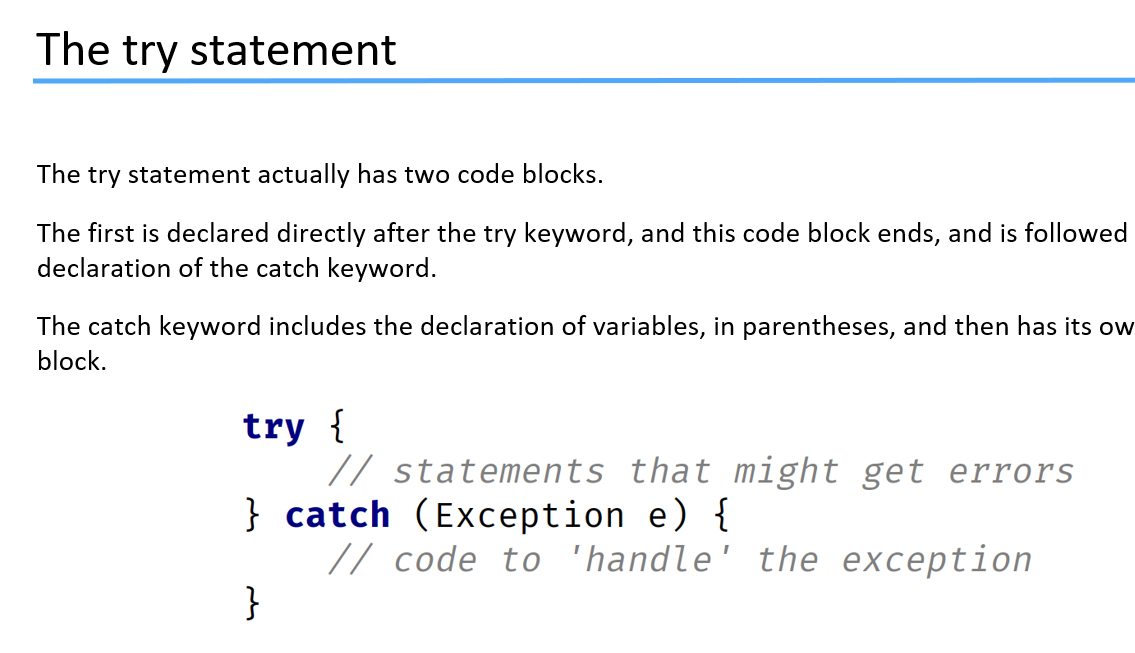
1. **Console and Scanner Read**
2. **Scanner Input**

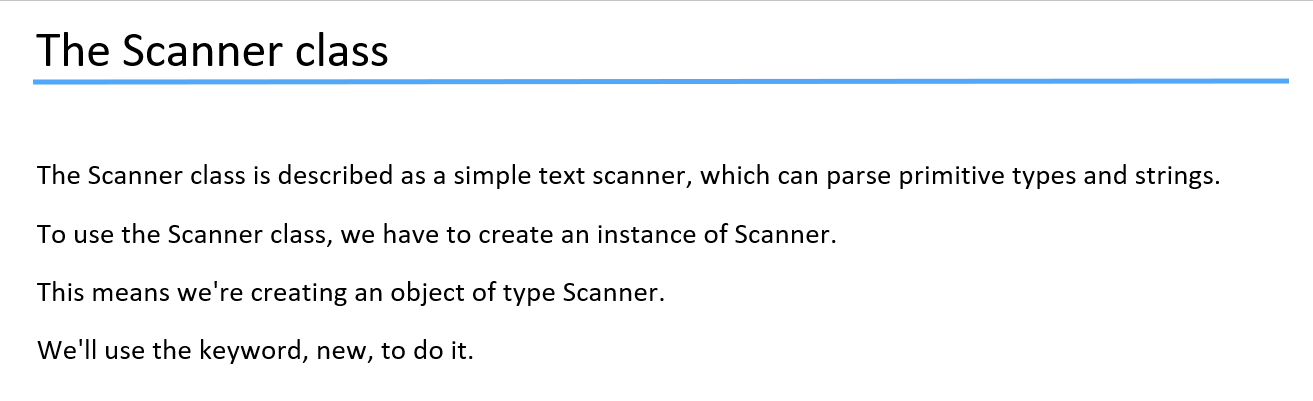
**Exception Handling in Java:**

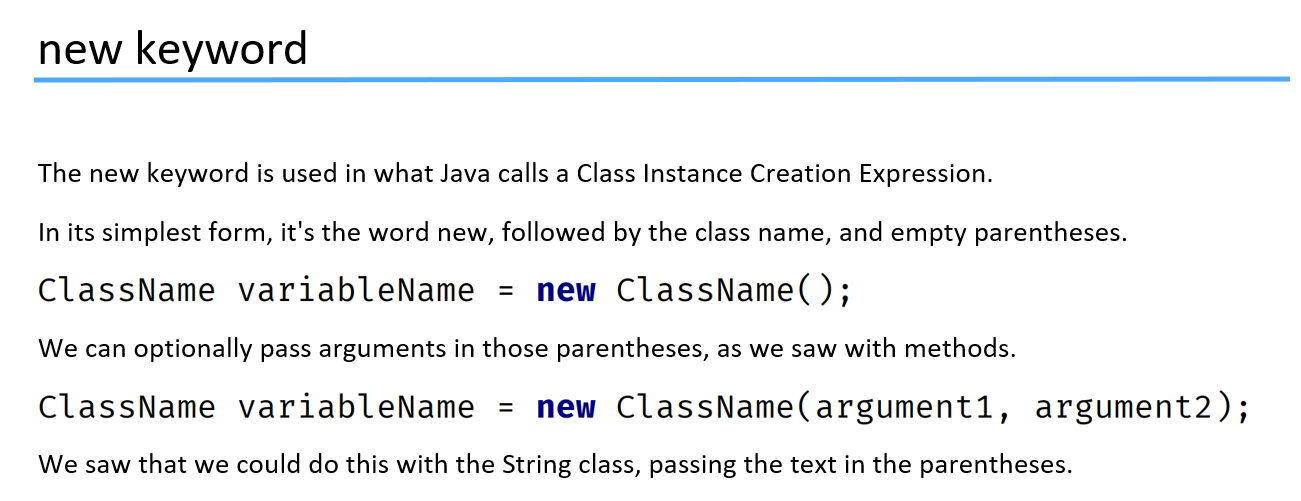


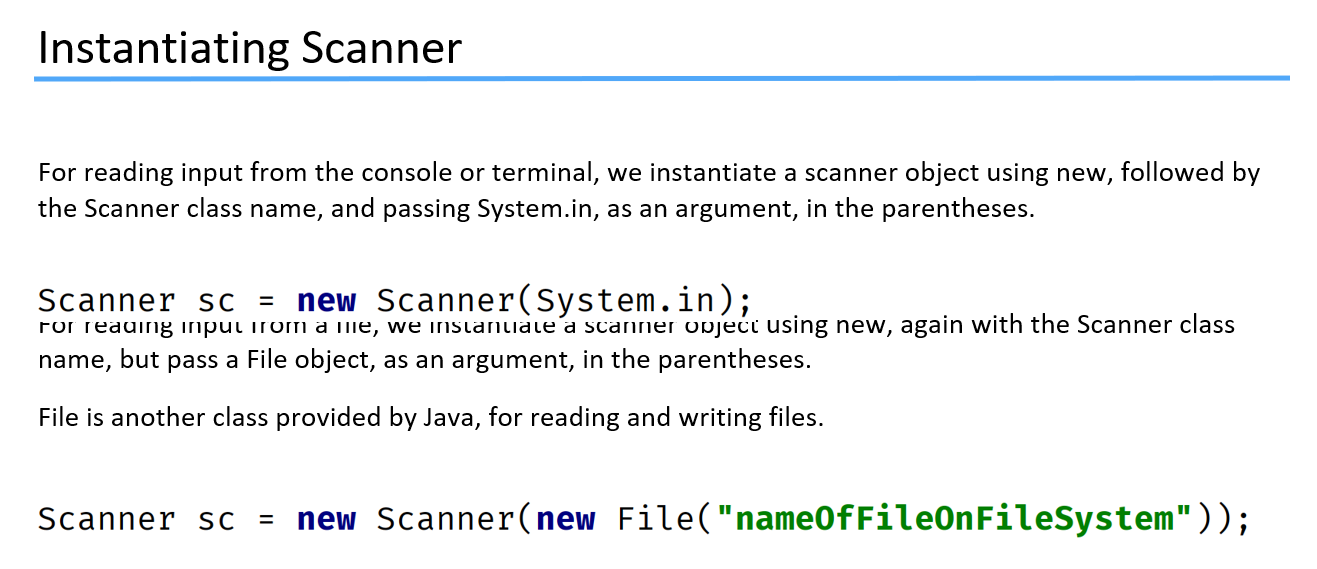


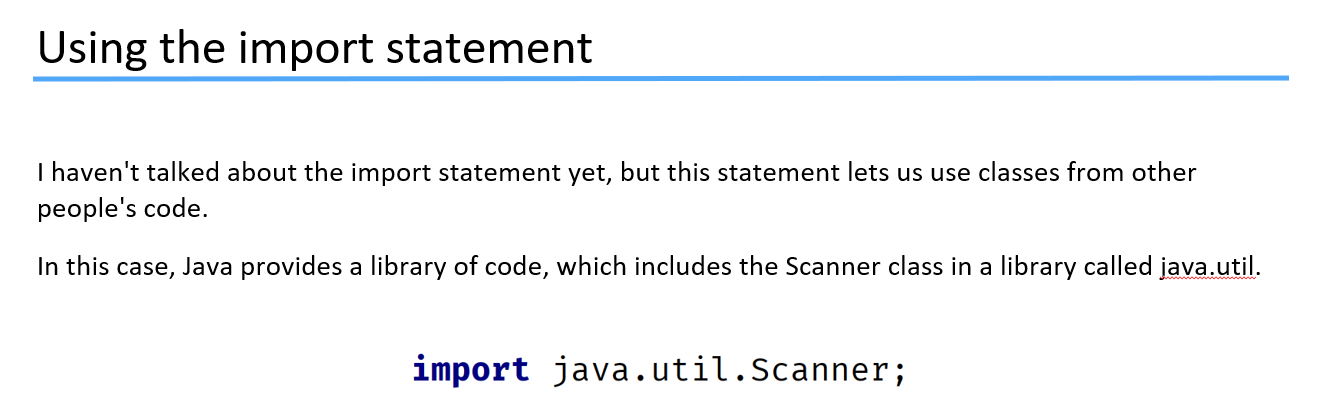
Catch statement is used with try statement … if any error occurs in try block then to handle that error we have catch statement. Catch block is used to perform a task if there is error in code…

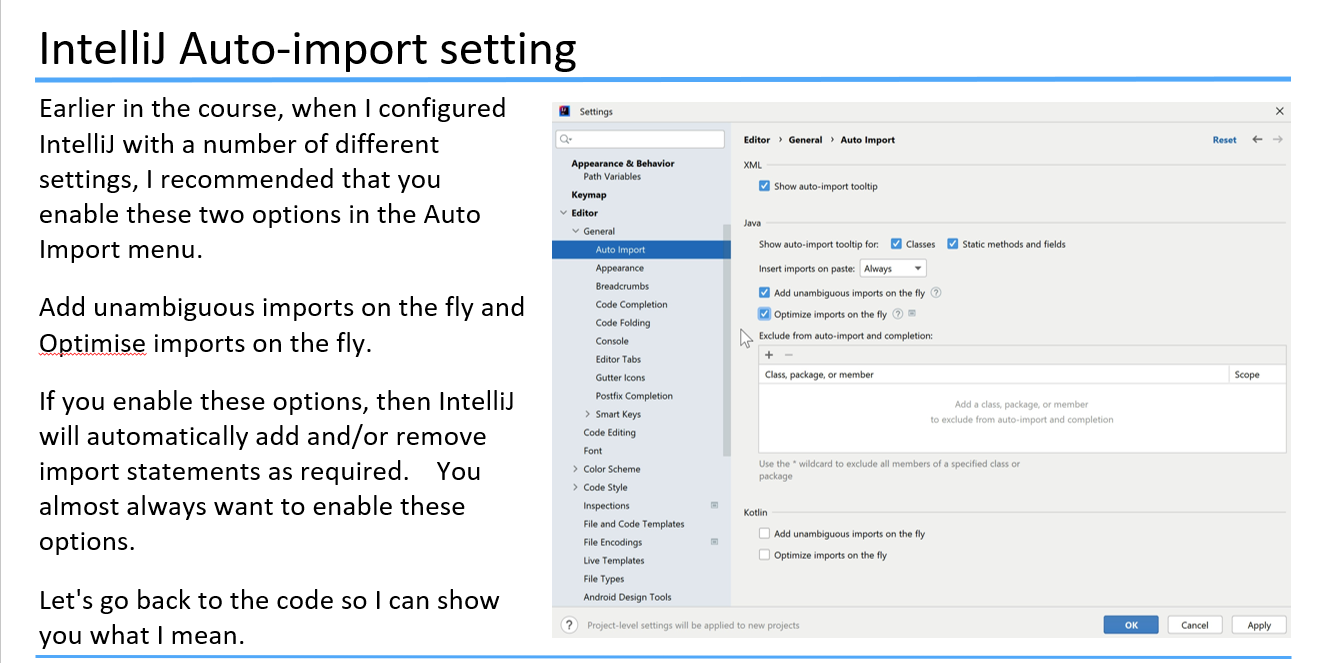












**Code:**

1. **More Interractive Scanner Input Code**
2. **Reading user Input Challenge**
3. **More Better Reading User Input Challenge**
4. **Min and Max Challenge**
5. **Paint Job**
6. **Sum and Average Calculator**

**Object Oriented Programming:**

**Classes and Objects:**

**Refer to ppt:**

**44. Introduction to Classes and Objects**

**45. Introduction to Classes Using Getter Methods**

**46. Classes Using Setters and Creating Objects**

**Code:**

1. **Classes and Object Start**
2. **Classes and Object Challenge**

**Refer to ppt:**

**47. Constructor Part 1**

**48. Constructor Part 2**

**Code:**

1. **Constructors in Classes and Objects Part 1**
2. **Constructors in Classes and Objects Part 2**