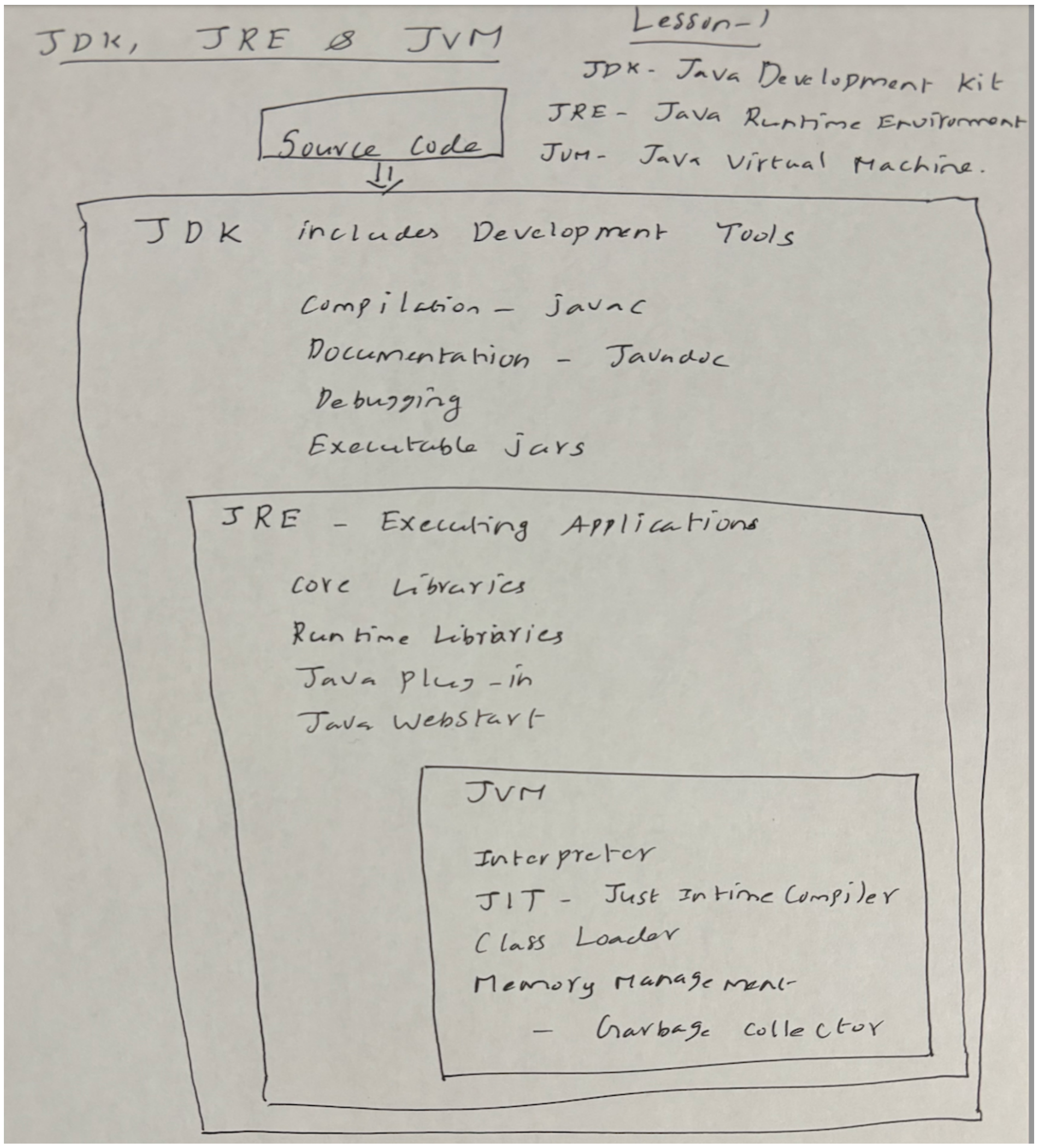
Lesson - 1

Java Support OO, Functional Programming

OO - Concepts

- Class
- Object
- Polymorphism - Multiple form
- Inheritance - Derive a new class from the existing class



Lesson – 2

Data Types

1. Primitive Types
2. Reference Type / Object Type

How == Works on float and double

- The == operator compares the **bitwise representation** of the two float or double values.
- If the values are exactly equal at the binary level, the comparison returns true.
- == works for exact equality but may fail due to precision errors in calculations.
- Consider BigDecimal for precision-critical applications and can use equals() to check.

char type - Assign single character in quotes

char ch = 'A';

char ch1 = '\u0041';

char ch2 = 65; \\ 65 is considered as constant and should be in the range 0-*65535*

To print the Unicode special characters on the console

IntelliJ - Go to File -> File Properties -> File Encoding to UTF-8

How to get input from the console

1. JoptionPane - Swing library(JDK1.2) - Read as String, need parsing to convert number type
2. BufferedReder & InputStreamReader(JDK1.1) - Read as String using Read(),

   ReadLine()

3. Scanner - JDK 5.0 (Mostly recommended) - Read as String, Int, Float

BufferedReader ob = new BufferedReader(new

InputStreamReader(System.in));

Scanner ob = new Scanner(System.in);

int x =ob.nextInt();

```
int x = 10;

 int y = 5;
 int z = 0;
 if( x>y)
    z = x;
else
   x = y;
```

Ternary operator(?:) z = (x>y)?x:y;

Find greatest among three numbers using Ternary operator

```
int a = 100, b=55,c=19;
int max = a > b ? (a > c ? a : c) : (b > c ? b : c) ;
System.out.println(max);
```

Math class is called as Utility class. You cannot create object for the Math class due to private constructor and not inherit from Math class due to final class. It's not immutable class.

Math ob = new Math(); //will get compilation error

It includes static fields and static methods.

Automatic Promotion

```
byte b1 = 10;
byte b2 = 11;
byte b3 = (byte) (b1+b2); // or int b3 = b1 + b2;
```

Java String

String x = "Java";

// Within Double quotes – sequence of Character

Strings are Immutable

Try to modify the x value

X = x + “Programming”;

If modify the value it does not modify the original, instead of modifying it will create a new String with the value of
“Java Programming”

// How to declare strings

String x = “Java” ; // String Literal

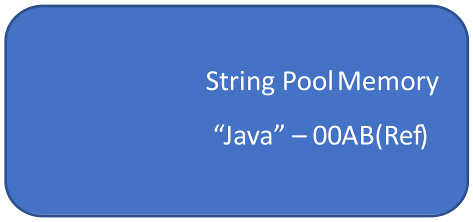
String x1 = new String(“Java”); String object

String Literal	**String Object**
Declare using Equal (=) like assignment	Create using new keyword
It Stores on String pool memory	It stores on heap memory
If string literals are equal or not compare using ==	If string objects are equal or not compare using equals() method
String comparison using == checks the references are same	String comparison using equals() checks the contents are same
Come from java.lang.String	Come from java.lang.String

String x = “Java” ; → Refer the pool memory 00AB

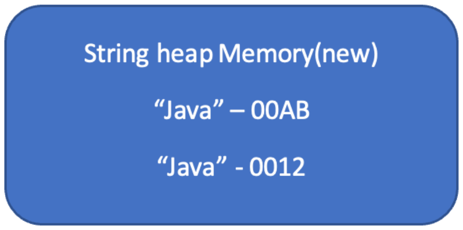
String y = “Java”; → Refer the pool memory 00AB

String z = y; // 00AB



String x1 = new String("Java"); // 00AB

String x1 = new String("Java"); // 0012



Reason behind Immutability

1. Security
    a. Strings are often used to store sensitive information like passwords, user credentials, and other data. If strings were mutable, modifying a string holding sensitive information could lead to security vulnerabilities.
2. Thread Safety
    a. In multithreaded environments, strings can be shared among multiple threads without the need for synchronization.

Formatting Strings

1. System.out.printf() – Will directly print on the console
2. String.format() → Which returns a string.

Refer: https://docs.oracle.com/javase/tutorial/java/data/numberformat.html

Looping

1. While(cond) { } – It execute only the condition becomes true – Entry controlled loop
2. Do {} while(condition) – Atleast one time statement will execute whether condition is true or false. → Exit Controlled loop
3. For(index, condition, inc/decrem)
4. For each

Array Copy Approaches

1. System.ArrayCopy(src, srcindex, destination, desindex, number of elements) – When you copy from one index to another index.
2. Arrays.copyOf(source, new size), Copy the original data along with resize.
3. Arrays.copyOfRange(Source, start, end) – Copy the range of values
4. Collection.Clone() – No resize , just get another copy

Command line Arguments.

Requirements

1. Read the input from the Commandline
2. Read from the args[] and add each word in to the string, separated by Comma and add (.) at the end.

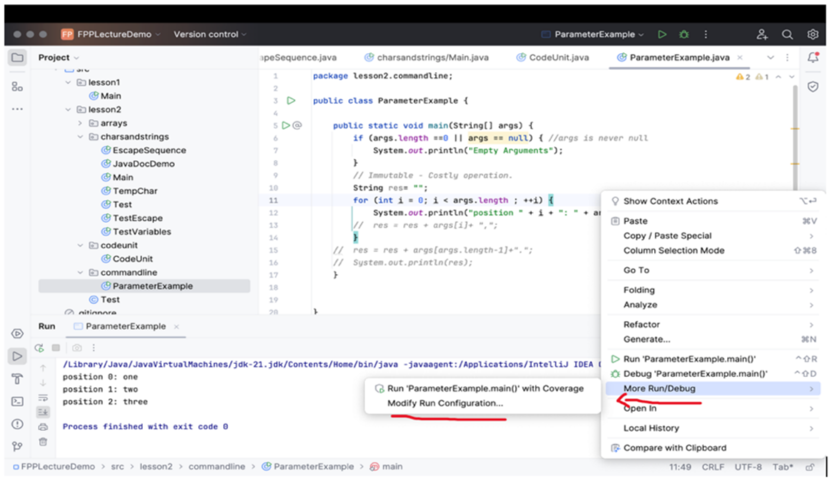
Inputs : args[] = {“Java”,”HTML”,”C++”} Outputs(String) : Java,HTML,C++.

```
String res = null;

For(int i =0; i<args.length-1;i++)

    Res = res + args[i] + “,”;

Res = res + args[length-1] + “.”;
```

Strings are immutable, to overcome costly concatenation

Give inputs for the Command line Arguments, main method args[] in IntelliJ IDEA,

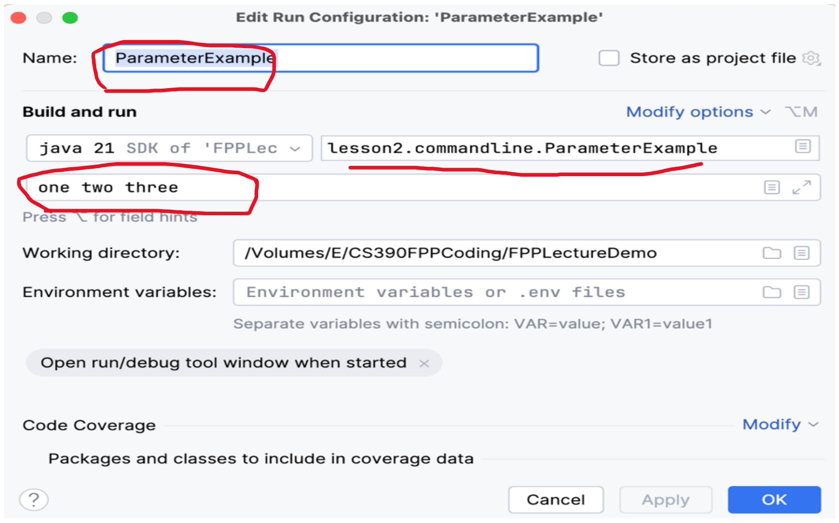
1. Right Click, Select More Run/Debug > Modify Run Configuration



```
package lesson2.commandline;

public class ParameterExample {

    public static void main(String[] args) {
        if (args.length ==0 || args == null) { //args is never null
            System.out.println("Empty Arguments");
        }
        // Immutable - Costly operation.
        String res= "";
        for (int i = 0; i < args.length ; ++i) {
            System.out.println("position " + i + ": " + ar
        //  res = res + args[i]+ ",";
        }
    //  res = res + args[args.length-1]+".";
    //  System.out.println(res);
    }


}
```

```
/Library/Java/JavaVirtualMachines/jdk-21.jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA
position 0: one
position 1: two
position 2: three

Process finished with exit code 0
```

Make sure your Class Name, and give input arguments separated by space like one two three, then click OK.


Edit Run Configuration: 'ParameterExample'

Name: ParameterExample

Store as project file

Build and run

Modify options ⌥M

java 21 SDK of 'FPPLec

lesson2.commandline.ParameterExample

one two three

Press ⌥ for field hints

Working directory: /Volumes/E/CS390FPPCoding/FPPLectureDemo

Environment variables: Environment variables or .env files

Separate variables with semicolon: VAR=value; VAR1=value1

Open run/debug tool window when started

Code Coverage

Modify

Packages and classes to include in coverage data

Java Mutable String Libraries for concatenation

1. StringBuilder – Thread Safe – Single threaded environment
2. StringBuffer – Thread Safe – Multithread environment

<u>About Switch Expression:</u>

When to use Arrow (->) Syntax: Use this for simple cases where the right-hand side of the arrow is a single expression or a straightforward block of code. It makes the code more concise and readable.

```
switch (variable) {
    case value1 -> expression1;
    case value2 -> expression2;
    // ...
    default -> defaultExpression;
}
```

When to use yield Keyword: Use this when you need to execute multiple statements or perform more complex logic within a case. The yield keyword is used to return a value from the case block.

The yield statement must be the last statement in each case block. It specifies the value to be returned from the switch expression for that case.

```
switch (variable) {
   case value1->{
      // multiple statements
         yield expression1;
}
   case value2->{
      // multiple statements
      yield expression2;
}
   // ...
   Default-> {
      // multiple statements
      yield defaultExpression;
}
}
```

HW Problem – 3 - Hints

"231A,Light Bulb,123,Wilco,1.75:"
"113D,Hairbrush,19,Aamco,3.75:"

231A	Light Bulb	123	Wilco	1.75
113D	Hairbrush	19	Aamco	3.75

String.Split()
Row Split use delimeter (:)
Column split use delimeter(,)