BITS Pilani

# Object-Oriented Programming (CS F213)

## Module I: Object-Oriented and Java Basics

### CS F213 RL 6.5: Command-Line Arguments

**Dr. Pankaj Vyas**
Department of Computer Science, BITS-Pilani, Pilani Campus

## CS F213 RL 6.5 : Topics

- Command-Line Agruments/Parameters

# Command-Line Arguments

- Parameters Passed to Driver main() Method during execution

```
public static  void   main(String args[]) { }
```

- Arguments are stored in a String type Array.
- 'How many number of arguments are passed' can be checked via <args.length>
- Example : Consider a driver class named 'Test'
- If you execute the file following commands

```
java Test                    → No Command-Line Arguments is passed
java Test 10 20 30           → Three Arguments are passed.
                               args[0] = "10",    args[1] = "20",        args[2] = "30"
java  Test Object Oriented Programming 20 30 → Five Arguments are passed.
                               args[0] = "Object",          args[1] = "Oriented",
                               args[2] = "Programming",     args[3] = "20" and     args[4] = "30"
```

Note : Actual Arguments starts after the name of the driver class. Each Argument should be space separated

## Command-Line Arguments : Example 1

```
// File Name: Test.java
class Test
{
		public	static	void	main(String args[])
		{
			System.out.println("Number of Arguments: " + args.length);
		}
}
```

```
F:\>java Test
Number of Arguments: 0

F:\>java Test I Love India 20 30
Number of Arguments: 5

F:\>java Test 1 2 3 4 Hello 5 6 I AM FINE
Number of Arguments: 10
```

# Command-Line Arguments : Example 2

```
// File Name: Test.java
class Test
{
        public      static      void        main(String args[])
        {
                System.out.println("Number of Arguments: " + args.length);
                // Displaying Individual Arguments
                for(int i =0; i< args.length; i++)
                        System.out.println(args[i]);

        } // End of Method
} // End of class Test
```

```
F:\>java Test
Number of Arguments:
0
```

```
F:\>java Test 1 2 3 4 Hello 5 6 I AM FINE
Number of Arguments: 10
1 ----------------------------------------------> args[0]
2 ----------------------------------------------> args[1]
3
4                                                   .
Hello                                               .
5                                                   .
6                                                   .
I
AM
FINE -------------------------------------------> args[9]
```

```
F:\>java Test I Love India 20 30
Number of Arguments: 5
I ------------------------------> args[0]
Love ---------------------------> args[1]
India                                .
20                                   .
30  ----------------------------> args[4]
```

# Retrieving The Actual Type of Parameters

- Every type of argument is stored in a String type array
- To retrieve the actual primitive type value, following methods can be used
  1. int Integer.parseInt(String s) → parseInt() is a static method of 'Integer' library class. Takes a String type as argument and returns an integer type.
  2. float Float.parseFloat(String s) → parseFloat() is a static method of 'Float' library class. Takes a String type as argument and returns a float type.
  3. double Double.parseDouble(String s) → parseDouble() is a static method of 'Double' library class. Takes a String type as argument and returns a double type
  4. char charAt(int index) → charAt() is a method of 'String' library class. Returns the character from a index.
  5. long Long.parseLong(String s) → parseLong() is a static method of 'Long' library class. Takes a String type as argument and returns a long type
  6. byte Byte.parseByte(String s) → parseByte() is a static method of 'Byte' library class. Takes a String type as argument and returns a byte type
  7. boolean Boolean.parseBoolean(String s) → parseBoolean() is a static method of 'Boolean' library class. Takes a String type as argument and returns a boolean type
  8. short Short.parseShort(String s) → parseShort() is a static method of 'Short' library class. Takes a String type as argument and returns a short type

**Note: Each of the above Method [except charAt()] Throws a NumberFormatException if the string does not represent the desired type**

## Examples of parsing

- int x = Integer.parseInt(“10”); **// Executes Successfully**
- int y = Integer.parseInt(“10.56”); **// Throws NumberFormatException**
- double y1 = Double.parseDouble(“10.56”); **// Executes Successfully**
- double y2 = Double.parseDouble(“xyz”); **// Throws NumberFormatException**

# Command-Line Arguments : Example 3

Write a program to perform addition, subtraction, multiplication of two numbers passed via command line. The type of operation which is to be performed on the numbers is to be decided via third command line which should be be either ‘a’ (for addition) or ‘s’ (for subtraction) or ‘m’ (for multiplication).

```
// File Name: Test.java
class Test
{
          public     static     void      main(String args[])
          {
                     double a = Double.parseDouble(args[0]);
                     double b = Double.parseDouble(args[1]);
                     char   c = args[2].charAt(0);

                     if( c == ‘a’)                  System.out.println(a+b);
                     else       if( c == ‘s’)       System.out.println(a-b);
                     else       if( c == ‘m’)       System.out.println(a * b);

          } // End of Method
}// End of class Test
```

# Command-Line Arguments : Example 3 ….

```
F:\>java Test
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 0
        at Test.main(You.java:6)
```

```
F:\>java Test 10 20 a
30.0
```

```
F:\>java Test 10 20 a
30.0
```

```
F:\>java Test 10 20 m
200.0
```

```
F:\>java Test 10 20 x
```

```
F:\>java Test 10 x 30
Exception in thread "main" java.lang.NumberFormatException: For input string: "x
"
        at sun.misc.FloatingDecimal.readJavaFormatString(Unknown Source)
        at sun.misc.FloatingDecimal.parseDouble(Unknown Source)
        at java.lang.Double.parseDouble(Unknown Source)
        at Test.main(You.java:7)
```

# Thank You