CSE215L: Programming Language II Lab

Faculty: Dr. Mohammad Rashedur Rahman (RRn)
Lab Manual 02: Primitive data types, Operators & Conditional Statements
Lab Instructor: Md. Mustafizur Rahman

Objectives:

- To learn about the primitive data types and variables in Java
- To know about the user input and output
- To learn about the different types of operators (Assignment, Arithmetic, Relational, Logical Operators and etc.)
- To learn to use conditional statements (if-else, switch case)
- To know about Enumerated Data Types

Primitive Data Types:

Primitive data types are predefined types of data, which are supported by the programming language. For example, integer, character, float, and double are all primitive data types. However, there are 8 primitive data types available in Java. The following code shows how to declare variables of different data types using literals (Any constant value assigned to the variable is called literal/constant).

```
public class DataTypes {

	public static void main(String[] args) {
		int anInt = 100;
		long aLong = 200L;
		byte aByte = 99;
		short aShort = -902;
		char aChar = 'A';
		float aFloat = 99.98F;
		double aDouble = 999.89;
		boolean aBoolean = true;

		System.out.println("anInt = " + anInt);
		System.out.println("aLong = " + aLong);
		System.out.println("aByte = " + aByte);
		System.out.println("aShort = " + aShort);
		System.out.println("aChar = " + aChar);
		System.out.println("aFloat = " + aFloat);
		System.out.println("aDouble = " + aDouble);
		System.out.println("aBoolean = " + aBoolean);
	}
}
```

Java Variables (Widening, Narrowing/Typecasting, Overflow):

Variable widening: When a small primitive type value is automatically accommodated in a bigger/wider primitive data type, this is called widening of the variable. In given example, *int* type variable is assigned to the *float-type* variable without any data loss or error.

Variable Narrowing: When a larger primitive type value is assigned in a smaller size primitive data type, this is called narrowing of the variable. It can cause some data loss due to less number of bits available to store the data. It requires **explicit type-casting** to the required data type. In given example, the *float* type variable is assigned to the *int* type variable with data loss.

Overflow: Overflow occurs when we assign such a value to a variable that is more than the maximum permissible value. In the given example, the overflow occurred since *the byte datatype value range lies between -128 to 127 (inclusive), much less than the int datatype value range*.

```
public class Variables {

	public static void main(String[] args) {
		// Widening
		int aInt1 = 10;
		float aFloat1 = aInt1;
		System.out.println(aInt1);
		System.out.println(aFloat1);

		// Narrowing/Typecasting
		float aFloat2 = 10.5F;
		//int aInt2 = aFloat2; //Compile time error
		int aInt2 = (int)aFloat2;
		System.out.println(aFloat2);
		System.out.println(aInt2);

		// Overflow
		int aInt = 130;
		byte aByte = (byte)aInt;
		System.out.println(aInt);
		System.out.println(aByte);
	}
}
```

User Input and Output:

There are several ways to get input from the user in Java. You will learn to get input by using the *Scanner* object in this course. For that, you need to *import Scanner* class and have to create an object of *Scanner* class which will be used to get input from the user. However, for the output, you can simply use *System.out.println(), System.out.print()* or *System.out.printf()* to send output to standard output (screen).

```
public class UserInput {

	public static void main(String[] args) {
		Scanner input = new Scanner(System.in);

		System.out.print("Enter an Integer: ");
		int i = input.nextInt();

		System.out.print("Enter a Double: ");
		double d = input.nextDouble();
```

```
            System.out.print("Enter a String: ");
            String s = input.next();

            System.out.println("Integer: " + i);
            System.out.println("Double: " + d);
            System.out.println("String: " + s);

            input.close();
      }
}
```

Operators:

Java provides a rich set of operators to manipulate variables, such as Arithmetic Operators, Relational Operators, Logical Operators, and Ternary operator. The following code shows how to find the min and max of two numbers using the Ternary Operator.

```
public class Operators {

      public static void main(String[] args) {
            int a = 10;
            int b = 50;
            int min = (a<b) ? a:b;
            int max = (a>b) ? a:b;
            System.out.println("Min: " + min);
            System.out.println("Max: " + max);
      }
}
```

Conditional statement (If-else):

The Java if statement is used to test the condition. It checks the boolean condition: *true* or *false*. The following code shows how to determine grade using the basic *if-else-if* ladder.

```
public class ConditionalStatement1 {

      public static void main(String[] args) {
            int marks = 90;

            if(marks<50){
                  System.out.println("fail");
            }
            else if(marks>=50 && marks<60){
                  System.out.println("D grade");
            }
            else if(marks>=60 && marks<70){
                  System.out.println("C grade");
            }
            else if(marks>=70 && marks<80){
                  System.out.println("B grade");
            }
            else if(marks>=80 && marks<90){
                  System.out.println("A grade");
            }
```

```
            else if(marks>=90 && marks<100){
                  System.out.println("A+ grade");
            }
            else{
                  System.out.println("Invalid!");
            }
      }
}
```

Conditional statement (Switch):

The switch statement works with *byte*, *short*, *int*, *long*, *enum* types, *String,* and some wrapper types like *Byte*, *Short*, *Integer*, and *Long*. (A Wrapper class or types is a class whose object wraps or contains primitive data types. Every primitive type has a corresponding wrapper type or class.)

```
public class ConditionalStatement2 {

      public static void main(String[] args) {
          int number = 20;
          switch(number){
                case 10:
                   System.out.println("10");
                   break;
                case 20:
                   System.out.println("20");
                   break;
                case 30:
                   System.out.println("30");
                   break;
                default:
                   System.out.println("Not in 10, 20 or 30");
          }
      }
}
```

Enumerated Types:

Enumerations represent a group of named constants in Java. Enums are used when we know all possible values at compile time, such as choices on a menu, command line flags, etc. In Java, enums are represented using *enum* data type. Additionally, we can add variables, methods, and constructors to it. The main objective of *enum* is to define our own data types, called Enumerated Data Types.

```
public class Enumeration {

      enum Coffee {Small, Medium, Large}

      public static void main(String[] args) {
            Coffee morningCoffee = Coffee.Large;
            System.out.println(morningCoffee);
      }
}
```

Tasks:

1. Write a program that will read N numbers using a **while-loop** and will display their sum. N will be input to your program.

2. Write a program that takes an arithmetic operator ('+', '-', '*' or '/') and two operands as input and performs the corresponding arithmetic operation on the operands. You must use **switch-case** to solve the problem.

```
Enter operator: +
Enter 1st Operand:  23
Enter 2nd Operand: 12
Result is : 35
```

```
Enter operator: /
Enter 1st Operand:  12
Enter 2nd Operand: 6
Result is : 2
```

3. Write a program that prints the following pattern.

```
*
+++
*****
+++++++
*********
```

4. Write a program to find the roots of a quadratic equation.

   For $ax^2 + bx + c = 0$, the values of x which are the solutions of the equation are given by :

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Input

```
Enter the coefficients:
a: 2
b: -4
c: -3
```

Output

```
Root 1: 2.58
Root 2: -0.58
```