Usman Institute of Technology

Department of Computer Science

Course Code: SE312

Course Title: Software Construction and Development

SPRING 2024

Lab 04

Objective: Understand the basic syntax of Java, including data types, variables, operators, and control flow structures.

Student Information

Student Name	Syed Muhammad Zaid
Student ID	20B-052-SE
Date	23/3/2024

Assessment

Marks Obtained	
Remarks	
Signature	

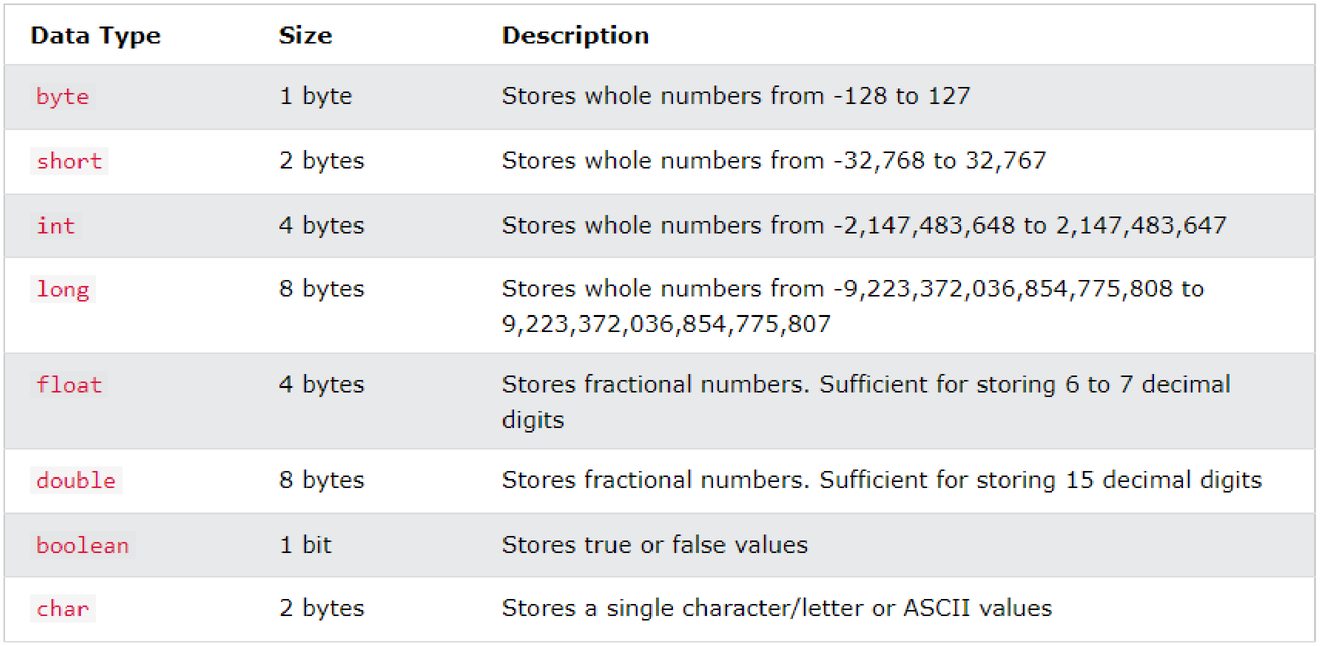
LAB #04

Data types

- In programming, a data type is a classification that specifies which type of value a variable can hold and what operations can be performed on that variable.
- Data types define the structure and behavior of variables in a programming language, allowing the compiler or interpreter to allocate memory and perform operations efficiently.

➢ Primitive data types

- Java has eight primitive data types: byte, short, int, long, float, double, char, and boolean.
- Primitive data types represent single values and are not objects.
- They have predefined sizes and ranges.



Data Type	Size	Description
`byte`	1 byte	Stores whole numbers from -128 to 127
`short`	2 bytes	Stores whole numbers from -32,768 to 32,767
`int`	4 bytes	Stores whole numbers from -2,147,483,648 to 2,147,483,647
`long`	8 bytes	Stores whole numbers from -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
`float`	4 bytes	Stores fractional numbers. Sufficient for storing 6 to 7 decimal digits
`double`	8 bytes	Stores fractional numbers. Sufficient for storing 15 decimal digits
`boolean`	1 bit	Stores true or false values
`char`	2 bytes	Stores a single character/letter or ASCII values

➢ Reference Data Types

Example 1: Using String reference data type

```
String message = "Hello, Java!";

System.out.println(message);
```

Example 2: Using arrays (reference data type)

```
int[] numbers = {1, 2, 3, 4, 5};  // Declaration and initialization of an integer array

System.out.println("First element: " + numbers[0]);
```

Example 3: Manipulating string objects

```
String firstName = "Sitwat";

String lastName = "Ashraf";

String fullName = firstName + " " + lastName;

System.out.println("Full Name: " + fullName);
```

Variables

- Variables are used to store data values in Java.
- Variable names must start with a letter, underscore, or dollar sign, followed by letters, digits, underscores, or dollar signs.

Declaring (Creating) Variables

To create a variable, you must specify the type and assign it a value:

```
type variableName = value;
```

Example

```
int x = 5;

double y = 4.5;
```

****Lab Activity**

1. Create a variable called name of type String and assign it the value "your name".
2. Create a variable that should store your Roll number.

You can also declare a variable without assigning the value, and assign the value later:

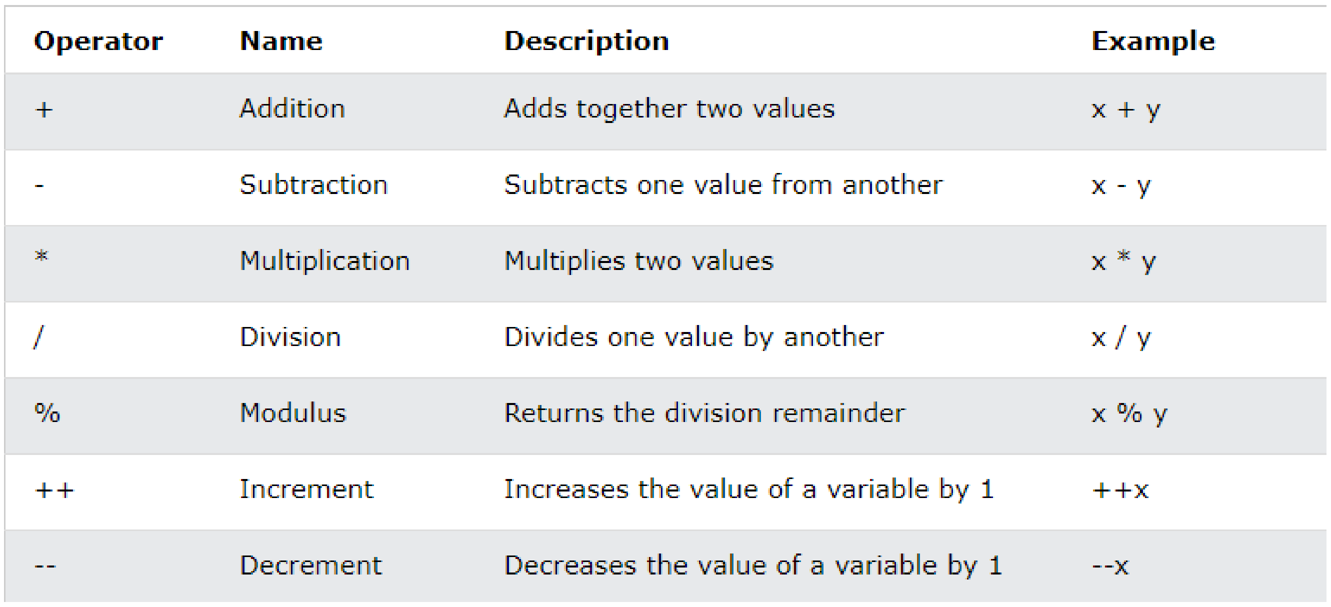
```
int myRollNum;
myRollNum = 001;
System.out.println(myRollNum);
```

Operators

- Operators are used to perform operations on variables and values.
- Java divides the operators into the following groups:

➢ Arithmetic operators

Arithmetic operators are used to perform common mathematical operations.



Operator	Name	Description	Example
+	Addition	Adds together two values	x + y
-	Subtraction	Subtracts one value from another	x - y
*	Multiplication	Multiplies two values	x * y
/	Division	Divides one value by another	x / y
%	Modulus	Returns the division remainder	x % y
++	Increment	Increases the value of a variable by 1	++x
--	Decrement	Decreases the value of a variable by 1	--x

➢ Assignment Operators

Assignment operators are used to assign values to variables.

Example

```
int x = 10;
```

Operator	Example	Same As
=	x = 5	x = 5
+=	x += 3	x = x + 3
-=	x -= 3	x = x - 3
*=	x *= 3	x = x * 3
/=	x /= 3	x = x / 3
%=	x %= 3	x = x % 3
&=	x &= 3	x = x & 3
|=	x |= 3	x = x | 3
^=	x ^= 3	x = x ^ 3
>>=	x >>= 3	x = x >> 3
<<=	x <<= 3	x = x << 3

➢ **Comparison Operators**

- Comparison operators are used to compare two values (or variables).
- This is important in programming, because it helps us to find answers and make decisions.
- The return value of a comparison is either true or false

Example

```
int x = 5;
int y = 3;
System.out.println(x > y);
```

Operator	Name	Example
==	Equal to	x == y
!=	Not equal	x != y
>	Greater than	x > y
<	Less than	x < y
>=	Greater than or equal to	x >= y
<=	Less than or equal to	x <= y

➢ **Logical Operators**

- You can also test for true or false values with logical operators.
- Logical operators are used to determine the logic between variables or values.

Operator	Name	Description	Example
&&	Logical and	Returns true if both statements are true	x < 5 && x < 10
||	Logical or	Returns true if one of the statements is true	x < 5 || x < 4
!	Logical not	Reverse the result, returns false if the result is true	!(x < 5 && x < 10)

Control flow structures

➢ **If-Else Statement:**

The if-else statement executes a block of code if a specified condition is true. If the condition is false, another block of code can be executed.

Example

```
int number = 10;
if (number > 0) {
   System.out.println("Number is positive");
} else {
   System.out.println("Number is negative");
}
```

- **Switch Statements**
  - Instead of writing many if..else statements, you can use the switch statement.
  - The switch statement selects one of many code blocks to be executed.

- ✓ The switch statement works with;
- ✓ The switch expression is evaluated once.
- ✓ The value of the expression is compared with the values of each case.
- ✓ If there is a match, the associated block of code is executed.
- ✓ The break and default keywords are optional.

Example

```
int day = 4;
switch (day) {
  case 1:
    System.out.println("Monday");
    break;
  case 2:
    System.out.println("Tuesday");
    break;
  case 3:
    System.out.println("Wednesday");
    break;
  case 4:
    System.out.println("Thursday");
    break;
  case 5:
    System.out.println("Friday");
    break;
  case 6:
    System.out.println("Saturday");
    break;
  case 7:
    System.out.println("Sunday");
    break;
}
```

Break

In Java, when the program encounters a break keyword within a switch block, it immediately exits the switch block. This termination prevents further execution of code within the block and halts any additional case testing. Once a match is found and the corresponding actions are performed, the break statement signals that further evaluation is unnecessary, thus ending the switch block's execution at that point.

Default

The default keyword specifies some code to run if there is no case match.

➢ Loops

Loops are capable of repeatedly executing a block of code until a certain condition is met. They offer practical benefits such as time-saving, error reduction, and improved code readability.

- **While loop**

  The while loop iterates over a block of code as long as a given condition remains true:

Example

```
public class lab4 {
  public static void main(String[] args) {
    int i = 0;
    while (i < 5) {
      System.out.println(i);
      i++;
    }
  }
}
```

- **Do/while loop**

  The do/while loop serves as an alternative to the while loop. It begins by executing the code block once, then evaluates the condition. If the condition is true, it continues to repeat the loop; otherwise, it terminates.

Example

```
public class lab4 {
  public static void main(String[] args) {
    int i = 0;
    do {
      System.out.println(i);
      i++;
    }
    while (i < 5);
  }
}
```

- **For loop**

  When you have a predetermined number of iterations for looping through a code block, it's preferable to use the for loop over a while loop.

Example

```
for (int i = 0; i < 5; i++) {
  System.out.println(i);
}
```

TASKS

1. Write a Java program that prompts the user to enter marks obtained in three subjects and calculates the total marks. Display the total marks obtained by the student.

```java
import java.util.Scanner;

public class task1 {
    int SDC;
    int AI;

    public task1() {
        Scanner scanner = new Scanner(System.in); //Getting the input from
the user.
        System.out.println("Enter SDC marks: ");
        this.SDC = scanner.nextInt();

        System.out.println("Enter AI marks: ");
        this.AI = scanner.nextInt();

    }

    public int getSDC() {
        return SDC;
    }

    public int getAI() {
        return AI;
    }
}
```

2. Write a Java program that prompts the user to enter ages of three people and determines the oldest person among them. Display the age of the oldest person.

```java
/*Write a Java program that prompts the user to enter ages of three people
 and determines the oldest person among them. Display the age of the
oldest person.*/

import java.util.Scanner;

public class task2 {
    int person = 0;

    public task2() {
        Scanner scanner = new Scanner(System.in);
        for (int i = 0; i < 3; i++) {
            System.out.println("""
                    Enter Person %d age:
                    """.formatted(i));
            int newAge = scanner.nextInt();

            if (newAge > person) {
                person = newAge;
            }
        }
//          scanner.close();

    }

    public int getPerson() {
        return person;
    }
}
```

3. Write a Java program to check whether a person is eligible to vote. Prompt the user to enter their age and display whether they are eligible to vote or not.

```java
import java.util.Scanner;

public class task3 {
    public task3() {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter your age to check the eligibility: ");
        int age = scanner.nextInt();

        if (age < 18) {
            System.out.println("You're NOT eligible.");
        } else {
            System.out.println("You're eligible.");
        }
    }

}
```

4. Write a Java program to print the multiplication table of a given number (n). Prompt the user to enter a number (n) and then print its multiplication table up to 10.

```java
import java.util.Scanner;

public class task4 {
    public task4() {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter a number to create table of: ");
        int table_num = scanner.nextInt();
        System.out.println("Printing the table of " + table_num);
        for (int i = 0; i < 11; i++) {
            System.out.println(table_num + " x " + i + " = " + table_num *
i);
        }
    }
}
```

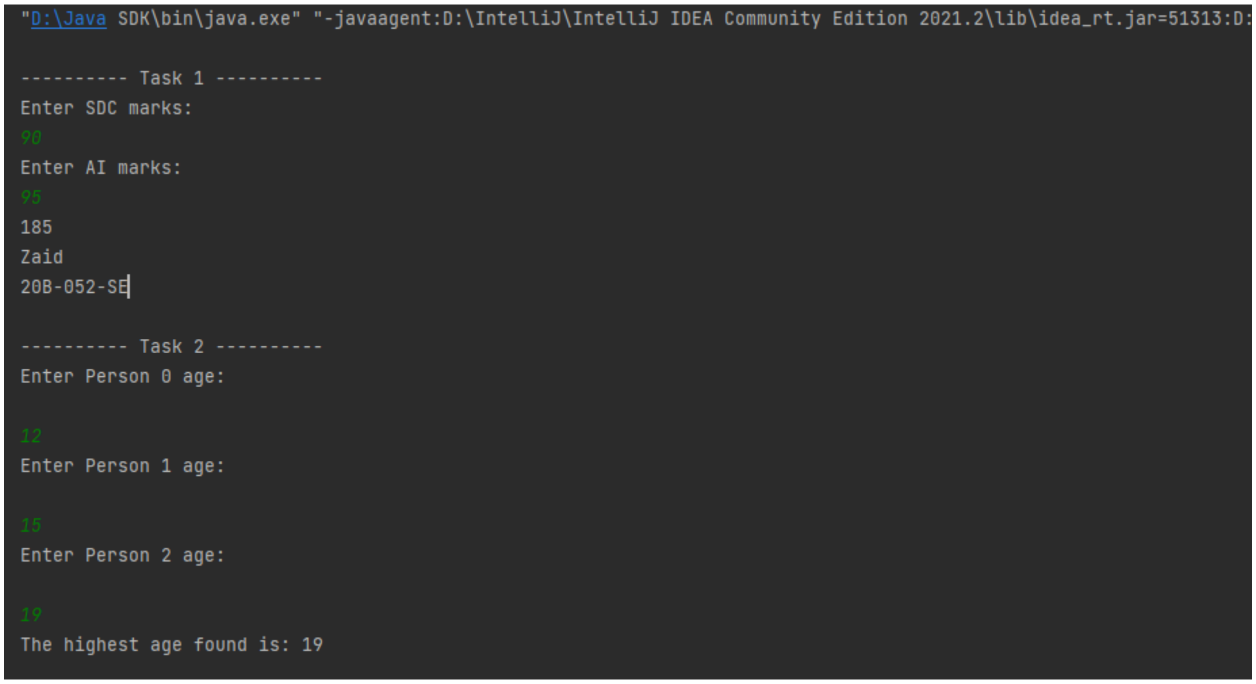
5. Write a Java program to calculate the sum of numbers from 1 to a given number (n). Prompt the user to enter a number (n) and then calculate the sum of all numbers from 1 to n. Display the result.

```java
import java.util.Scanner;

public class task5 {
    int res = 0;

    public task5() {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter a number to calculate sum from 1 to that
number: ");
        int num = scanner.nextInt();
        System.out.println("Calculating the sum...");
        for (int i = 0; i < num + 1; i++) {
            res += i;
        }
    }

    public int getRes() {
        return res;
    }
}
```

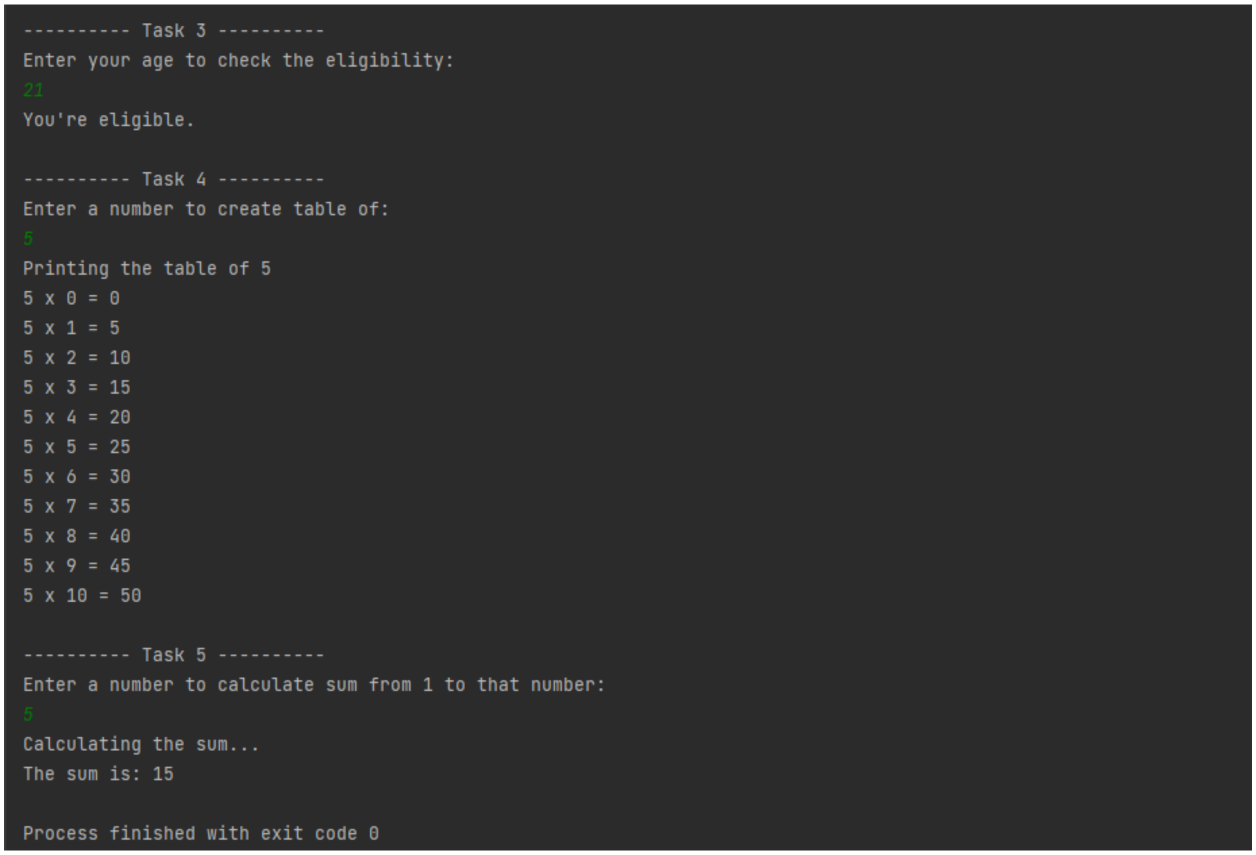
Creating a Main class to run all the classes:

```
public class Main {
    String studentName = "Zaid";
    String rollNumber = "20B-052-SE";

    public static void main(String[] args) {
        Main mainObject = new Main();

        System.out.println("\n---------- Task 1 ----------");
        task1 task1Object = new task1();
        System.out.println(task1Object.AI + task1Object.SDC);
        System.out.println(mainObject.studentName);
        System.out.println(mainObject.rollNumber);

        System.out.println("\n---------- Task 2 ----------");
        task2 task2Object = new task2();
        System.out.println("The highest age found is: " +
task2Object.person);

        System.out.println("\n---------- Task 3 ----------");
        task3 task3Object = new task3();

        System.out.println("\n---------- Task 4 ----------");
        task4 task4Object = new task4();

        System.out.println("\n---------- Task 5 ----------");
        task5 task5Object = new task5();
        System.out.println("The sum is: " + task5Object.getRes());
    }
}
```

OUTPUT (Tasks 1 – 5):



```
"D:\Java SDK\bin\java.exe" "-javaagent:D:\IntelliJ\IntelliJ IDEA Community Edition 2021.2\lib\idea_rt.jar=51313:D:

---------- Task 1 ----------
Enter SDC marks:
90
Enter AI marks:
95
185
Zaid
20B-052-SE

---------- Task 2 ----------
Enter Person 0 age:

12
Enter Person 1 age:

15
Enter Person 2 age:

19
The highest age found is: 19
```



```
---------- Task 3 ----------
Enter your age to check the eligibility:
21
You're eligible.

---------- Task 4 ----------
Enter a number to create table of:
5
Printing the table of 5
5 x 0 = 0
5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
5 x 6 = 30
5 x 7 = 35
5 x 8 = 40
5 x 9 = 45
5 x 10 = 50

---------- Task 5 ----------
Enter a number to calculate sum from 1 to that number:
5
Calculating the sum...
The sum is: 15

Process finished with exit code 0
```

All the code files are attached with the document.