Java Programming Notes

Hia Al Saleh

September 19th, 2024

Contents

1 Math Library in Java

The Java Math library offers various functions to perform mathematical operations, such as calculating exponents, square roots, and logarithms.

1.1 Key Math Functions

Here are some common functions from the Math library:

- `Math.pow(base, exponent)`: Calculates the value of the base raised to the power of the exponent.
- `Math.sqrt(number)`: Returns the square root of the given number.
- `Math.log10(number)`: Returns the base 10 logarithm of the given number.

1.2 Example: Exponent Calculation

Here is an example that calculates an exponent using `Math.pow()`:

```
double base = 2;
double exponent = 3;
double result = Math.pow(base, exponent);
System.out.println("Result: " + result);
```

2 Conditional Logic in Java

Conditional logic in Java allows the program to make decisions based on certain conditions. The most common conditional structures are the `if`, `else if`, and `else` statements.

2.1 If Statement

The `if` statement executes a block of code if the given condition evaluates to true.

```
int a = 5;
if (a > 3) {
    System.out.println("a is greater than 3");
}
```

2.2 If-Else Statement

The `if-else` statement provides an alternative block of code to execute if the condition evaluates to false.

```
int b = 2;
if (b > 3) {
    System.out.println("b is greater than 3");
} else {
    System.out.println("b is less than or equal to 3");
}
```

2.3 Else-If Statement

The `else-if` statement allows multiple conditions to be checked in sequence. If the first condition fails, the program checks the next one.

```
int c = 7;
if (c > 10) {
    System.out.println("c is greater than 10");
} else if (c > 5) {
    System.out.println("c is greater than 5 but less than
       or equal to 10");
} else {
    System.out.println("c is less than or equal to 5");
}
```

3 Boolean Data Type

In Java, the `boolean` data type represents two possible values: `true` or `false`. Boolean values are commonly used in conditional expressions.

```
boolean isHungry = true;
if (isHungry) {
    System.out.println("I am hungry");
} else {
    System.out.println("I am not hungry");
}
```

4 Comparison Operators

Comparison operators are used to compare two values. These operators return a boolean value (`true` or `false`).

- `==`: Equal to
- `!=`: Not equal to
- `>`: Greater than
- `<`: Less than

- `>=`: Greater than or equal to
- `<=`: Less than or equal to

4.1 Example: Using Comparison Operators

```
int x = 10;
int y = 5;
System.out.println(x > y); // true
System.out.println(x == y); // false
System.out.println(x != y); // true
```

5 Logical Operators

Logical operators allow combining multiple boolean expressions into a single expression.

- `&&`: Logical AND (both conditions must be true)
- `||`: Logical OR (at least one condition must be true)
- `!`: Logical NOT (inverts the boolean value)

5.1 Example: Logical Operators

```
boolean isRaining = false;
boolean haveUmbrella = true;

if (!isRaining || haveUmbrella) {
    System.out.println("I can go outside");
}
```

6 Creating a Java Class in IntelliJ

To create a new Java class in IntelliJ, follow these steps:

1. Open IntelliJ and select `File > New > Project`.
2. Choose Java and set the project SDK to version 1.8.
3. Give the project a name, e.g., `FunctionCalculator`.
4. Right-click on the `src` folder and select `New > Java Class`.
5. Name the class `FunctionCalculator` and hit enter.

6.1 Writing a Program

In the new `FunctionCalculator.java` file, you can start by creating the `main` method using the following command:

```
public class FunctionCalculator {
    public static void main(String[] args) {
        System.out.println("Welcome to Function Calculator
            !");
    }
}
```

7 Homework and Next Steps

Read Chapter 3 and 4 of your textbook to prepare for next week's topics, which will cover:

- Repetition Structures
- Sentinel Values
- While and For Loops
- Nested Loops