**Chapter-2**

**Writing a Simple Program:**

Every Java program begins with a class definition in which the keyword **class** is followed by the class name. The outline of the program would look like this:

**public class** ComputeArea {

// Details to be given later

}

Every Java program must have a **main** method where program execution begins. The program is then expanded as follows:

**public class** ComputeArea {

**public static void** main(String[] args) {

// Step 1: Read in radius

// Step 2: Compute area

// Step 3: Display the area

}

}

**What is a variable?**

A Java variable is a piece of memory that can contain a data value. A variable thus has a data type. Data types are covered in more detail in the text on Java data types. Every variable has a name, a type, a size, and a value. A variable represents a value stored in the computer’s memory. Example:

Double x = 10;

Double y = 15;

**What is primitive data type?**

Primitive types are the most basic data types available within the Java language. There are 8: boolean , byte , char , short , int , long , float and double . These types serve as the building blocks of data manipulation in Java. These data types are defined into the Java type system by default, they come with a number of operations predefined.

**What is reserved word?**

Java reserved words are keywords that are reserved by Java functions or other uses that cannot be used as identifiers (e.g., variable names, function names, class names). Example: return, void, static, public, this.

**What is tracing a program?**

The method of reviewing how a program works is called tracing a program. Tracing programs are helpful for understanding how programs work, and they are useful tools for finding errors in programs.

N.B The plus sign (+) has two meanings: one for addition and the other for concatenating (combining) strings.

**What is IPO?**

IPO stands for- input, processing, output. Input is receiving input from the user; process is producing results using the input; and output is displaying the results.

**What is identifier?**

Identifiers are the names that identify the elements such as classes, methods, and variables in a program. All identifiers must obey the following rules:

■ An identifier is a sequence of characters that consists of letters, digits, underscores (\_), and dollar signs ($).

■ An identifier must start with a letter, an underscore (\_), or a dollar sign ($). It cannot start with a digit.

■ An identifier cannot be a reserved word. (See Appendix A for a list of reserved words.)

■ An identifier cannot be true, false, or null.

■ An identifier can be of any length.

**What is variable?**

Variables are used to represent values that may be changed in the program. variables are used to store values to be used later in a program. They are called variables because their values can be changed. Variables are for representing data of a certain type. The variable declaration tells the compiler to allocate appropriate memory space for the variable based on its data type. The syntax for declaring a variable is

datatype variableName;

N.B. A variable must be declared before it can be assigned a value. A variable declared in a method must be assigned a value before it can be used. Whenever possible, declare a variable and assign its initial value in one step. This will make the program easy to read and avoid programming errors.

**Assignment Statements and Assignment Expressions**

An assignment statement designates a value for a variable. An assignment statement can be used as an expression in Java. the equal sign (=) is used as the assignment operator. The syntax for assignment statements is as follows:

variable = expression;

An expression represents a computation involving values, variables, and operators that, taking them together, evaluates to a value.

**What is constant?**

A constant is a variable whose value cannot change once it has been assigned. Java doesn't have built-in support for constants, but the variable modifiers **static** and **final** can be used to effectively create one. Constants can make your program more easily read and understood by others. In addition, a constant is cached by the JVM as well as your application, so using a constant can improve performance. Here is the syntax for declaring a constant:

final datatype CONSTANTNAME = value;

**What is the benefit of using constant?**

There are three benefits of using constants:

(1) you don’t have to repeatedly type the same value if it is used multiple times;

(2) if you have to change the constant value (e.g., from 3.14 to 3.14159 for PI), you need to change it only in a single location in the source code; and

(3) a descriptive name for a constant makes the program easy to read.

**What are the conventions for naming variable, methods, and classes?**

conventions for naming variables, methods, and classes are:-

■ Use lowercase for variables and methods. If a name consists of several words, concatenate them into one, making the first word lowercase and capitalizing the first letter of each subsequent word—for example, the variables radius and area and the method print.

■ Capitalize the first letter of each word in a class name—for example, the class names ComputeArea and System.

■ Capitalize every letter in a constant and use underscores between words—for example, the constants PI and MAX\_VALUE. It is important to follow the naming conventions to make your programs easy to read.

**Numeric Data Types and Operations:**

Java has six numeric types for integers and floating-point numbers with operators +, -, \*, /, and %.

Java provides eight primitive data types for numeric values, characters, and Boolean values.

**Name Range Storage Size**

byte -27 to 27 - 1 (-128 to 127) **8-bit signed**

short -215 to 215 - 1 (-32768 to 32767) 16 -**bit signed**

int -231 to 231 - 1 (-2147483648 to 2147483647) **32-bit signed**

long -263 to 263 - 1 **64-bit signed** (i.e., -9223372036854775808 to 9223372036854775807)

float Negative range: -3.4028235E + 38 to -1.4E - 45 32-bit IEEE 754 Positive range: 1.4E - 45 to 3.4028235E + 38

double Negative range: -1.7976931348623157E + 308 to -4.9E – 324 64-bit IEEE 754

N.B. Java uses four types for integers: byte, short, int, and long. Java uses two types for floating-point numbers: float and double.

**Methods for Scanner Objects**

**Method Description**

nextByte() reads an integer of the byte type.

nextShort() reads an integer of the short type.

nextInt() reads an integer of the int type.

nextLong() reads an integer of the long type.

nextFloat() reads a number of the float type.

nextDouble() reads a number of the double type.

N.B. The + and - operators can be both unary and binary. A unary operator has only one operand; a binary operator has two. For example, the - operator in -5 is a unary operator to negate number 5, whereas the - operator in 4 - 5 is a binary operator for subtracting 5 from 4.

**Math.pow:** The Math.pow(a, b) method can be used to compute ab. The pow method is defined in the Math class in the Java. the syntax Math.pow(a, b) (e.g., Math.pow(2, 3)), which returns the result of ab (23). Here, a and b are parameters for the pow method and the numbers 2 and 3 are actual values used to invoke the method.

**What is Literal?**

A literal is a constant value that appears directly in a program. Literals are syntactic representations of boolean, character, numeric, or string data For example, 34 and 0.305 are literals in the following statements:

int numberOfYears = 34;

double weight = 0.305;

N.B. You can make a number a float by appending the letter f or F, and you can make a number a double by appending the letter d or D. For example, you can use 100.2f or 100.2F for a float number, and 100.2d or 100.2D for a double number. A float value has **7** to **8** number of significant digits and a double value has **15** to **17 n**umber of significant digits.

**What are the software Development Process?**

1. Requirements
2. Specification
3. Analysis
4. Design
5. Implementation
6. Testing
7. Deployment
8. And maintenance.

**What are the common errors and pitfalls?**

Common elementary programming errors often involve undeclared variables, uninitialized variables, integer overflow, unintended integer division, and round-off errors.

**CHAPTER SUMMARY**

1. Identifiers are names for naming elements such as variables, constants, methods, classes, packages in a program.

2. An identifier is a sequence of characters that consists of letters, digits, underscores (\_), and dollar signs ($). An identifier must start with a letter or an underscore. It cannot start with a digit. An identifier cannot be a reserved word. An identifier can be of any length.

3. Variables are used to store data in a program. To declare a variable is to tell the compiler what type of data a variable can hold.

4. There are two types of import statements: specific import and wildcard import. The specific import specifies a single class in the import statement; the wildcard import imports all the classes in a package.

5. In Java, the equal sign (=) is used as the assignment operator.

6. A variable declared in a method must be assigned a value before it can be used.

7. A named constant (or simply a constant) represents permanent data that never changes.

8. A named constant is declared by using the keyword final.

9. Java provides four integer types (byte, short, int, and long) that represent integers of four different sizes.

10. Java provides two floating-point types (float and double) that represent floatingpoint numbers of two different precisions.

11. Java provides operators that perform numeric operations: + (addition), – (subtraction), \* (multiplication), / (division), and % (remainder).

12. Integer arithmetic (/) yields an integer result.

13. The numeric operators in a Java expression are applied the same way as in an arithmetic expression.

14. Java provides the augmented assignment operators += (addition assignment), –= (subtraction assignment), \*= (multiplication assignment), /= (division assignment), and %= (remainder assignment).

15. The increment operator (++) and the decrement operator (––) increment or decrement a variable by 1.

16. When evaluating an expression with values of mixed types, Java automatically converts the operands to appropriate types.

17. You can explicitly convert a value from one type to another using the (type) value notation.

18. Casting a variable of a type with a small range to a variable of a type with a larger range is known as widening a type.

19. Casting a variable of a type with a large range to a variable of a type with a smaller range is known as narrowing a type.

20. Widening a type can be performed automatically without explicit casting. Narrowing a type must be performed explicitly.

21. In computer science, midnight of January 1, 1970, is known as the UNIX epoch.

**Chapter-3**

**What is Boolean Data Type?**

The boolean data type declares a variable with the value either true or false. A Boolean expression is an expression that evaluates to a Boolean value: true or false. A variable that holds a Boolean value is known as a Boolean variable. The Boolean data type is used to declare Boolean variables. The boolean-expression is enclosed in parentheses. A boolean variable can hold one of the two values: true or false. Example: boolean lightsOn = true;

**What is if statement?**

An if statement is a construct that enables a program to specify alternative paths of execution. An if statement executes statements if the boolean-expression evaluates to true. If the boolean-expression evaluates to true, the statements in the block are executed.

**N.B.**  Java has several types of selection statements: one-way if statements, two-way if-else statements, nested if statements, multi-way if-else statements, switch statements, and conditional expressions. A one-way if statement executes an action if and only if the condition is true. The syntax for a one-way if statement is:

if (boolean-expression) {

statement(s);

}

**What are the common errors in if Statement?**

1. Forgetting Necessary Braces
2. Wrong Semicolon at the if Line
3. Redundant Testing of Boolean Values
4. Dangling else Ambiguity
5. Equality Test of Two Floating-Point Values

**What is the logical operator used for?**

The logical operators !, &&, ||, and ^ can be used to create a compound Boolean expression. Logical operators, also known as Boolean operators, operate on Boolean values to create a new Boolean value.

**Switch Statement:**

A switch statement executes statements based on the value of a variable or an expression.

Here is the full syntax for the switch statement:

switch (switch-expression) {

case value1: statement(s)1;

break;

case value2: statement(s)2;

break;

...

case valueN: statement(s)N;

break;

default: statement(s)-for-default;

}

**The switch statement observes the following rules:**

■ The switch-expression must yield a value of char, byte, short, int, or String type and must always be enclosed in parentheses. (The char and String types will be introduced in the next chapter.)

■ The value1, . . ., and valueN must have the same data type as the value of the switchexpression. Note that value1, . . ., and valueN are constant expressions, meaning that they cannot contain variables, such as 1 + x.

■ When the value in a case statement matches the value of the switch-expression, the statements starting from this case are executed until either a break statement or the end of the switch statement is reached.

■ The default case, which is optional, can be used to perform actions when none of the specified cases matches the switch-expression.

■ The keyword break is optional. The break statement immediately ends the switch statement.

**Conditional Expressions**

A conditional expression evaluates an expression based on a condition. Conditional expressions are in a completely different style, with no explicit if in the statement.

**The syntax is:**

boolean-expression ? expression1 : expression2;

**What is Debugging?**

Debugging is the process of finding and fixing errors in a program. Logic errors are called bugs. A common approach to debugging is to use a combination of methods to help pinpoint the part of the program where the bug is located.

**CHAPTER SUMMARY**

1. A boolean type variable can store a true or false value.

2. The relational operators (<, <=, ==, !=, >, >=) yield a Boolean value.

3. Selection statements are used for programming with alternative courses of actions. There are several types of selection statements: one-way if statements, two-way if-else statements, nested if statements, multi-way if-else statements, switch statements, and conditional expressions.

4. The various if statements all make control decisions based on a Boolean expression. Based on the true or false evaluation of the expression, these statements take one of two possible courses.

5. The Boolean operators &&, ||, !, and ^ operate with Boolean values and variables.

6. When evaluating p1 && p2, Java first evaluates p1 and then evaluates p2 if p1 is true; if p1 is false, it does not evaluate p2. When evaluating p1 || p2, Java first valuates p1 and then evaluates p2 if p1 is false; if p1 is true, it does not evaluate p2. Therefore, && is referred to as the conditional or short-circuit AND operator, and || is referred to as the conditional or short-circuit OR operator.

7. The switch statement makes control decisions based on a switch expression of type char, byte, short, int, or String.

8. The keyword break is optional in a switch statement, but it is normally used at the end of each case in order to skip the remainder of the switch statement. If the break statement is not present, the next case statement will be executed.

9. The operators in expressions are evaluated in the order determined by the rules of parentheses, operator precedence, and operator associativity.

10. Parentheses can be used to force the order of evaluation to occur in any sequence.

11. Operators with higher precedence are evaluated earlier. For operators of the same precedence, their associativity determines the order of evaluation.

12. All binary operators except assignment operators are left-associative; assignment

operators are right-associative.

**Chapter-4**

**What is method?**

A method is a group of statements that performs a specific task. For example: Math.pow(a, b) method to compute ab and Math.random for generating a random number . Math class can be categorized in three types. Such as:- trigonometric, exponent and service methods. Math class provides two useful double constants, PI and E.

**The min, max, and abs Methods**

The min and max methods return the minimum and maximum numbers of two numbers (int, long, float, or double). For example, max(4.4, 5.0) returns 5.0, and min(3, 2) returns 2.

The abs method returns the absolute value of the number (int, long, float, or double).

For example,

Math.max(2, 3) returns 3

Math.max(2.5, 3) returns 4.0

**Character Data Type and Operations**

A character data type represents a single character. The character data type, char, is used to represent a single character. A character literal is enclosed in single quotation marks. Consider the following code:

char letter = 'A';

char numChar = '4';

**What is encoding?**

apping a character to its binary representation is called encoding. How characters are encoded is defined by an encoding scheme.

**What is supplementary character?**

Those characters that go beyond the original 16-bit limit are called supplementary characters. Java supports the supplementary characters.

**What is String?**

A string is a sequence of characters. To represent a string of characters, use the data type called String.

Example: String message = "Welcome to Java";

N.B. The toLowerCase() method returns a new string with all lowercase letters and the toUpperCase() method returns a new string with all uppercase letters. For example,

"Welcome".toLowerCase() returns a new string welcome.

"Welcome".toUpperCase() returns a new string WELCOME.

**The trim()** method returns a new string by eliminating whitespace characters from both ends of the string. The characters ' ', \t, \f, \r, or \n are known as whitespace characters. For example,

"\t Good Night \n".trim() returns a new string Good Night.

Frequently Used Format Specifiers

Format Specifier Output Example

%b a Boolean value true or false

%c a character ‘a’

%d a decimal integer 200

%f a floating-point number 45.460000

%e a number in standard scientific notation 4.556000e+01

%s a string “Java is cool”

Here is an example:

int count = 5;

double amount = 45.56;

System.out.printf("count is %d and amount is %f", count, amount);

**CHAPTER SUMMARY**

1. Java provides the mathematical methods sin, cos, tan, asin, acos, atan, toRadians, toDegree, exp, log, log10, pow, sqrt, cell, floor, rint, round, min, max, abs, and random in the Math class for performing mathematical functions.

2. The character type char represents a single character.

3. An escape sequence consists of a backslash (\) followed by a character or a combination of digits.

4. The character \ is called the escape character.

5. The characters ' ', \t, \f, \r, and \n are known as the whitespace characters.

6. Characters can be compared based on their Unicode using the relational operators.

7. The Character class contains the methods isDigit, isLetter, isLetterOrDigit, isLowerCase, isUpperCase for testing whether a character is a digit, letter, lowercase, and uppercase. It also contains the toLowerCase and toUpperCase methods for returning a lowercase or uppercase letter.

8. A string is a sequence of characters. A string value is enclosed in matching double quotes ("). A character value is enclosed in matching single quotes (').

9. Strings are objects in Java. A method that can only be invoked from a specific object is called an instance method. A non-instance method is called a static method, which can be invoked without using an object.

10. You can get the length of a string by invoking its length() method, retrieve a character at the specified index in the string using the charAt(index) method, and use the indexOf and lastIndexOf methods to find a character or a substring in a string. 11. You can use the concat method to concatenate two strings, or the plus (+) operator to concatenate two or more strings.

12. You can use the substring method to obtain a substring from the string.

13. You can use the equals and compareTo methods to compare strings. The equals method returns true if two strings are equal, and false if they are not equal. The compareTo method returns 0, a positive integer, or a negative integer, depending on whether one string is equal to, greater than, or less than the other string.

14. The printf method can be used to display a formatted output using format specifiers.

**Chapter-5**

**What is loop? How many kinds of loops are there in Java?**

*Loops* are constructs that control repeated executions of a block of statements. The concept of looping is fundamental to programming. Java provides three types of loop statements: **while** loops, **do**-**while** loops, and **for** loops.

**What is While loop?**

*A* **while** *loop executes statements repeatedly while the condition is true.* The part of the loop that contains the statements to be repeated is called the *loop body.* A one-time execution of a loop body is referred to as an *iteration (*or *repetition) of the loop.* Each loop contains a *loop-continuation-condition* The syntax for the **while** loop is:

**while** (loop-continuation-condition) {

// Loop body

Statement(s);

}

N.B. The **while** loop repeatedly executes the statements in the loop body when the **loop-continuation-condition** evaluates to **true**.

**What is the loop Design Strategies?**

Consider three steps when writing a loop.

**Step 1:** Identify the statements that need to be repeated.

**Step 2:** Wrap these statements in a loop like this:

**while** (**true**) {

Statements;

}

**Step 3:** Code the **loop-continuation-condition** and add appropriate statements for controlling the loop.

**while** (loop-continuation-condition) {

Statements;

Additional statements for controlling the loop;

}

**What is sentinel value?**

In programming, **sentinel value** is a special **value** that is used to terminate a loop. A loop that uses a sentinel value to control its execution is called a *Sentinel-controlled loop.*

**What is do-while-loop?**

A do-while loop is the same as a while loop except that it executes the loop body first and then checks the loop continuation condition. The do-while loop executes the loop body first, then checks the loop continuation- condition to determine whether to continue or terminate the loop. The do-while loop is a variation of the while loop. Its syntax is:

do {

// Loop body;

Statement(s);

}

**The for Loop**

A for loop has a concise syntax for writing loops. The for loop statement starts with the keyword for, A for loop can be used to simplify the preceding loop as:

for (initial-action; loop-continuation-condition;

action-after-each-iteration) {

// Loop body;

Statement(s);

}

N.B. The while loop and for loop are called pretest loops because the continuation condition is checked before the loop body is executed. The do-while loop is called a posttest loop because the condition is checked after the loop body is executed.

**Nested Loop:**

A loop can be nested inside another loop. Nested loops consist of an outer loop and one or more inner loops. Each time the outer loop is repeated, the inner loops are reentered, and started anew.

**Keywords break and continue:**

The break and continue keywords provide additional controls in a loop. Continue breaks out of iteration while the break keyword breaks out of a loop. The continue statement is always inside a loop. Too many break and continue statements will produce a loop with many exit points and make the program difficult to read.

**CHAPTER SUMMARY**

1. There are three types of repetition statements: the while loop, the do-while loop, and the for loop.

2. The part of the loop that contains the statements to be repeated is called the loop body.

3. A one-time execution of a loop body is referred to as an iteration of the loop.

4. An infinite loop is a loop statement that executes infinitely.

5. In designing loops, you need to consider both the loop control structure and the loop body.

6. The while loop checks the loop-continuation-condition first. If the condition is true, the loop body is executed; if it is false, the loop terminates.

7. The do-while loop is similar to the while loop, except that the do-while loop executes the loop body first and then checks the loop-continuation-condition to decide whether to continue or to terminate.

8. The while loop and the do-while loop often are used when the number of repetitions is not predetermined.

9. A sentinel value is a special value that signifies the end of the loop.

10. The for loop generally is used to execute a loop body a fixed number of times.

11. The for loop control has three parts. The first part is an initial action that often initializes a control variable. The second part, the loop-continuation-condition, determines whether the loop body is to be executed. The third part is executed after each iteration and is often used to adjust the control variable. Usually, the loop control variables are initialized and changed in the control structure.

12. The while loop and for loop are called pretest loops because the continuation condition is checked before the loop body is executed.

13. The do-while loop is called a posttest loop because the condition is checked after the loop body is executed.

14. Two keywords break and continue, can be used in a loop.

15. The break keyword immediately ends the innermost loop, which contains the break.

16. The continue keyword only ends the current iteration.