**Day 1 and Day 2 Assignment**

1. What are the different primitive data types available in Java?

Java has 8 primitive data types:

byte, int, short, long, float, double, Boolean, char

1. Explain the difference between primitive and non-primitive data types in Java?

**Primitive:**

* Stored by value
* Cannot be null
* Int, Boolean, float

**Non-Primitive:**

* Stored by reference
* Can be null
* String, arrays, classes

1. Write a Java program that demonstrates the use of all primitive data types?

**Program:**

public class PrimitiveDemo {

public static void main(String[] args) {

byte b = 10;

short s = 200;

int a = 20;

long l = 100000L;

float f = 10.5f;

double d = 20.99;

char c = 'A';

boolean isBoolean = true;

System.out.println("byte: " + b);

System.out.println("short: " + s);

System.out.println("int: " + a);

System.out.println("long: " + l);

System.out.println("float: " + f);

System.out.println("double: " + d);

System.out.println("char: " + c);

System.out.println("boolean: " + isBoolean);

}

}

**Output:**

byte: 10

short: 200

int: 20

long: 100000

float: 10.5

double: 20.99

char: A

boolean: true

1. What is type casting? Provide an example of implicit and explicit casting in Java?

Type casting is a process of converting a variable from one data type to another.

**Implicit:**

* Converting small data type to large data type
* Done automatically by java

**Ex:** int a = 10;

double d = a; // int to double (automatically)

System.out.println(d); // prints 10.0

**Explicit:**

* Converting large data type to small data type
* Must be done manually

**Ex:** double d = 7.5;

int a = (int) d; // double to int (explicit)

System.out.println(a); // prints 7

1. What is the default value of each primitive data type in java?

**Type Default Value**

byte 0

short 0

int 0

long 0L

float 0.0f

double 0.0d

char null character

boolean false

1. What are control statements in Java? List the types with examples?

Control Statements means control the flow of execution

**Types:**

* **Conditional:** if, if-else, if-else-if, switch
* **Looping:** for, while, do-while
* **Branching:** break, continue, return

**If:**

public class If\_statement

{

public static void main(String[] args)

{

int age = 20;

if(age >= 18)

{

System.*out*.println("Eligible for voting");

}

System.*out*.println("End of program");

}

}

**Output:** Eligible for voting

**If-else:**

public class IfElse\_Statement {

public static void main(String[] args) {

int marks = 75;

if (marks >= 50) {

System.out.println("You passed the exam!");

} else {

System.out.println("You failed the exam.");

}

}

}

**Output:** You passed the exam!

**If-else-if:**

public class Student\_Marks{

public static void main(String[] args)

{

float marks = 95;

if(marks >= 90){

System.***out***.println("Distinction");

}

else if(marks >= 75 && marks < 90){

System.***out***.println("First Class");

}

else if(marks >= 60 && marks < 75){

System.***out***.println("Second Class");

}

else if(marks >= 35 && marks < 60){

System.***out***.println("Third Class");

}

else {

System.***out***.println("Fail");

}

}

}

**Output:** Distinction

**Switch:**

public class Switch\_Statement {

public static void main(String[] args) {

int menu = 3;

switch(menu){

case 1 : System.***out***.println("Tea");

break;

case 2 : System.***out***.println("Coffee");

break;

case 3 : System.***out***.println("Water Bottle");

break;

case 4 : System.***out***.println("Egg-Puff");

break;

case 5 : System.***out***.println("Veg-Puff");

break;

case 6 : System.***out***.println("Samosa");

break;

default : System.***out***.println("Invalid input");

}

}

}

**Output:** Water Bottle

**for:**

public class for\_loop {

public static void main(String[] args) {

for(int i = 1; i <= 5; i++)

{

System.***out***.println(i);

}

}

}

**Output:**

1

2

3

4

5

**while:**

public class while\_demo {

public static void main(String[] args) {

int i = 1;

while(i <= 5){

System.***out***.println(i);

i++;

}

}

}

**Output:**

1

2

3

4

5

**do-while:**

public class do\_while\_demo {

public static void main(String[] args)

{

int i = 1;

do{

System.***out***.println(i);

i++;

}

while(i <= 5);

}

}

**Output:**

1

2

3

4

5

1. Write a Java program to demonstrate the use of if-else and switch-case statements?

**Program:**

public class ControlDemo {

public static void main(String[] args) {

int num = 5;

if (num > 0) {

System.out.println("Positive");

} else {

System.out.println("Negative or Zero");

}

switch (num) {

case 1 : System.out.println("One");

case 5 : System.out.println("Five");

default : System.out.println("Other");

}

}

}

**Output:**

Positive

Five

1. What is the difference between break and continue statements?

**BREAK:**

* It will terminate the loop immediately
* It helps to exit the loop completely
* Often used when a certain condition is met and no further loop execution is needed
* Applicable in for, while, do-while and switch

**CONTINUE:**

* The current iteration will be skipped and moves to the next
* Loop continues running from the next iteration
* Used when certain iterations need to be skipped without stopping the loop
* Applicable in for, while and do-while

1. Write a java program to print even numbers between 1 to 50 using a for loop?

**Program:**

public class EvenNumbersBtw1to50 {

public static void main(String[] args) {

for (int i = 1; i <= 50; i++) {

if (i % 2 == 0) { // checks for the even number condition

System.***out***.print(i+" ");

}

}

}

}

**Output:**

2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50

1. Explain the differences between while and do-while loops with examples?

The while and do-while loops are two common looping constructs in Java used for repeating a block of code based on a condition.

**while loop:**

* The while loop tests the loop condition before executing the loop body
* If the condition is false initially, the loop body may never execute
* It is primarily used when the number of iterations is not known before and you want to ensure the condition is checked before each iteration.

**Program:**

public class while\_demo {

public static void main(String[] args)

{

int i = 1;

while(i <= 10)

{

System.***out***.print(i+" ");

i++;

}

}

}

**Output:**

1 2 3 4 5 6 7 8 9 10

**do-while:**

* The do-while loop executes the loop body at least once before testing the condition
* The condition is tested after the loop body executes
* This ensures that the loop body runs at least once irrespective of the condition
* It is used when you want the code inside the loop to execute before any condition check

**Program:**

public class do\_while\_demo {

public static void main(String[] args)

{

int i = 1;

do

{

System.***out***.print(i+" ");

i++;

}

while(i <= 10);

}

}

**Output:**

1 2 3 4 5 6 7 8 9 10

1. What are keywords in java? List 10 commonly used keywords.

Keywords are predefined, reserved words that have a specific meaning to the java compiler. Each keyword serves a special purpose in defining the structure, control flow and behaviour of a java program.

class

public

static

void

int

if

else

for

return

new

1. Explain the purpose of the following keywords: static, final, this, super.

* **static:**

The static keyword is used to declare class-level members (variables or methods) that belongs to the class itself rather than to any specific instance (object) of the class.

* **final:**

The final keyword is used to declare constants or variables/ methods/ classes that are not allowed to be changed or overridden once initialized or defined.

* **this:**

The this keyword refers to the current instance of the class where it is used.

1. What are the types of operators in java?

Java supports several types of operators:

* **Arithmetic:** Used to perform basic mathematical calculations like addition, subtraction, multiplication, division and modulus.
* **Unary:** Operator on a single operand to perform operations such as increment, decrement, negation and logical complement.
* **Assignment:** Used to assign values to variables. Compound assignment operators combine arithmetic or bitwise operations with assignment.
* **Relational:** Compare two values and return a Boolean result (true or false).
* **Logical:** Combine multiple Boolean expressions to form complex conditions.
* **Bitwise:** Operate on the individual bits of integer types to perform bit-level manipulations.
* **Shift:** Shift the bits of a number to the left or right.
* **Ternary:** A conditional operator that takes three operands, acting as a shorthand for if-else statements.
* **Instance Of:** Tests whether an object is an instance of a specific class or implements an interface.

1. Write a java program demonstrating the use of arithmetic, relational and logical operators?

**Program:**

import java.util.Scanner;

public class AllOperators {

public static void main(String[] args) {

Scanner in = new Scanner(System.in);

System.out.println("Enter Value Of A: ");

int a = in.nextInt();

System.out.println("Enter Value Of B: ");

int b = in.nextInt();

// Arithmetic Operators

System.out.println("Arithmetic Operators:");

System.out.println("a + b = " + (a + b));

System.out.println("a - b = " + (a - b));

System.out.println("a \* b = " + (a \* b));

System.out.println("a / b = " + (a / b));

System.out.println("a % b = " + (a % b));

// Relational Operators

System.out.println("\nRelational Operators:");

System.out.println("a == b: " + (a == b));

System.out.println("a != b: " + (a != b));

System.out.println("a > b: " + (a > b));

System.out.println("a < b: " + (a < b));

System.out.println("a >= b: " + (a >= b));

System.out.println("a <= b: " + (a <= b));

// Logical Operators

boolean x = true, y = false;

System.out.println("\nLogical Operators:");

System.out.println("x && y: " + (x && y));

System.out.println("x || y: " + (x || y));

System.out.println("!x: " + (!x));

}

}

**Output:**

Enter Value Of A:

5

Enter Value Of B:

4

Arithmetic Operators:

a + b = 9

a - b = 1

a \* b = 20

a / b = 1

a % b = 1

Relational Operators:

a == b: false

a != b: true

a > b: true

a < b: false

a >= b: true

a <= b: false

Logical Operators:

x && y: false

x || y: true

!x: false

1. What is operator precedence? How does it affect the outcome of expressions?

Operator precedence in Java (and most programming languages) refers to the rules that determine the order in which operators are evaluated in an expression when there are multiple operators involved and expressions can contain multiple operators, understanding and following operator precedence ensures that the expression is evaluated correctly and produces the intended result. Ignoring precedence rules or misunderstanding them can lead to unexpected outcomes or logical errors.

**Ex:** int result = (10 + 20) \* 3;

System.out.println(result);

Output:

90