# Java Programming Assignment

## Section 1: Java Data Types

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

a) Java has 8 primitive types: byte, short, int, long, float, double, char, boolean.

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

a) Primitive: Stores value directly, predefined by Java, no methods.  
Non-Primitive: Stores object reference, created by programmer, has methods.

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

a)

public class PrimitiveExample {

public static void main(String[] args) {

byte b = 100;

short s = 30000;

int i = 100000;

long l = 15000000000L;

float f = 5.75f;

double d = 19.99;

char c = 'A';

boolean bool = true;

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

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

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

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

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

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

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

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

}

}

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

a) Converting one type to another.  
Implicit: int x = 5; double y = x;  
Explicit: double a = 9.7; int b = (int) a;

5. What is the default value of each primitive data type in Java?

a) byte: 0, short: 0, int: 0, long: 0L, float: 0.0f, double: 0.0d, char: '\u0000', boolean: false

## Section 2: Java Control Statements

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

a) Control statements decide flow: Conditional (if-else, switch), Loops (for, while, do-while), Jump (break, continue, return).

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

a) if - else :

public class demo{  
 public static void main(String[] args) {  
 int num = 5;  
 if (num > 0){  
 System.out.println("Positive");

} else {

System.out.println("Non-positive");  
 }

}

Switch case:

public class demo{

public static void main(String[] args) {

int day = 3;

switch (day) {

case 1:

System.out.println("Monday");

break;

case 2:

System.out.println("Tuesday");

break;

case 3:

System.out.println("Wednesday");

break;

default:

System.out.println("Invalid day");

}

}

}

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

a) break exits loop/switch and continue skips current iteration.

4. Write a Java program to print even numbers between 1 to 50 using a for loop.

a) public class EvenNum {

public static void main(String[] args) {

System.out.println("Even numbers between 1 and 50:");

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

if (i % 2 == 0) {

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

}

}

}

}

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

a) while checks the condition before executing and do-while executes at least once.

Ex : while:

public class demowhile{

public static void main(String[] args) {

int i = 1;

while (i <= 5) {

System.out.println(i);

i++;

}

}

}

Do-while:

public class DoWhiledemo {

public static void main(String[] args) {

int i = 1;

do {

System.out.println(i);

i++;

} while (i <= 5);

}

}

## Section 3: Java Keywords and Operators

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

a) Java has a set of keywords that are reserved words that cannot be used as variables, methods, classes, or any other identifiers

Ex : class, public, static, void, if, else, for, while, return, new

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

a) static = class-level, final = constant/cannot be overrided, this = current object   
super = parent object

3. What are the types of operators in Java?

a) Arithmetic, Relational, Logical, Assignment, Unary, Bitwise, Ternary.

4. Write a Java program demonstrating the use of arithmetic, relational, and logical operators.

public class Operatorsdemo{ public static void main(String[] args) { int a = 10, b = 5;

// 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));  
  
 // 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));  
}  
}

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

a) Operator precedence is the order in which Java evaluates different operators in an expression.

Ex: int result = 10 + 5 \* 2;(Multiplication has high precedence than addition)

result = 10 + (5 \* 2) = 20

# Additional Questions

## Java Data Types

6. What is the size and range of each primitive data type in Java?

a) byte: 8 bits, range -128 to 127

short: 16 bits, range -32,768 to 32,767

int: 32 bits, range -2,147,483,648 to 2,147,483,647

long: 64 bits, range -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807

float: 32 bits, range ~±3.40282347E+38 (6–7 decimal digits)

double: 64 bits, range ~±1.79769313486231570E+308 (15 decimal digits)

char: 16 bits, range 0 to 65,535 (Unicode)

boolean: size JVM-dependent, values are true or false

7. How does Java handle overflow and underflow with numeric types?

a) Overflow happens when a calculation exceeds the maximum value of a type. Underflow happens when it goes below the minimum value. Java wraps around instead of throwing an error.

8. Write a program to convert a double value to an int without data loss.

a) public class DoubleToInt {

public static void main(String[] args) {

double d = 45.75;

int i = (int) Math.round(d); // Rounds to nearest integer

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

System.out.println("Converted int: " + i);

}

}

9. What is the difference between char and String in Java?

a) char is a primitive type that stores a single character. String is a class that stores a sequence of characters. char is enclosed in single quotes (e.g., 'A'), while String is enclosed in double quotes (e.g., "Java").

10. Explain wrapper classes and their use in Java.

a) Wrapper classes are object representations of primitive types in Java. They allow primitives to be used in collections, provide utility methods, and support autoboxing/unboxing.

Ex: public class WrapperExample {

public static void main(String[] args) {

int num = 10;

Integer obj = num; // Autoboxing

int val = obj; // Unboxing

System.out.println("Wrapper Object: " + obj);

System.out.println("Primitive Value: " + val);

}

}

## Java Control Statements

6. Write a Java program using nested if statements.

a) public class NestedIfdemo {

public static void main (String [] args) {

int num = 25;

if (num > 0) {

if (num % 2 == 0) {

System.out.println("The number is positive and even.");

} else {

System.out.println("The number is positive and odd.");

}

} else {

System.out.println("The number is not positive.");

}

}

}

7. Write a Java program to display the multiplication table of a number using a loop.

a) public class MultiplicationTable {

public static void main(String[] args) {

int num = 5;

for (int i = 1; i <= 10; i++) {

System.out.println(num + " x " + i + " = " + (num \* i));

}

}

}

8. How do you exit from nested loops in Java?

a) By assigning a label before the outer loop and use break label Name to exit all loops.

9. Compare and contrast for, while, and do-while loops.

a) for loop is best for a known number of iterations, initialization, condition, and increment in one place.

while loop is best when the number of iterations is unknown and checks condition before execution.

do-while loop Executes at least once because condition is checked after the loop body.

10. Write a program that uses a switch-case to simulate a basic calculator.

a) import java.util.Scanner;

public class Calculatordemo {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("Enter first number: ");

double num1 = sc.nextDouble();

System.out.print("Enter second number: ");

double num2 = sc.nextDouble();

System.out.print("Enter operator (+, -, \*, /): ");

char op = sc.next().charAt(0);

switch (op) {

case '+':

System.out.println("Result: " + (num1 + num2));

break;

case '-':

System.out.println("Result: " + (num1 - num2));

break;

case '\*':

System.out.println("Result: " + (num1 \* num2));

break;

case '/':

if (num2 != 0) {

System.out.println("Result: " + (num1 / num2));

} else {

System.out.println("Error: Division by zero");

}

break;

default:

System.out.println("Invalid operator");

}

sc.close();

}

}

## Java Keywords and Operators

6. What is the use of the `instanceof` keyword in Java?

a) Instanceof is used to check whether an object is an instance of a specific class or a subclass and returns boolean value true or false.

7. Explain the difference between `==` and `.equals()` in Java.

a) == Compares references for objects or values for primitives and .equals() Compares content of objects.

8. Write a program using the ternary operator.

a) public class Ternarydemo{

public static void main(String[] args) {

int num = 10;

String result = (num % 2 == 0) ? "Even" : "Odd";

System.out.println(num + " is " + result);

}

}

9. What is the use of `this` and `super` in method overriding?

a) this refers to the current class’s object, can be used to call another method in the same class and super refers to the parent class, used to call the overridden method of the parent inside the child class.

10. Explain bitwise operators with examples.

a) Works on individual bits of integer values. Common operators are :

& (AND) =1 if both bits are 1.

| (OR) = 1 if any bit is 1.

^ (XOR) = 1 if bits are different.

~ (NOT) = Inverts bits.

<< (Left shift) = Shifts bits left, fills with 0.

>> (Right shift) =Shifts bits right, keeps sign bit.

>>> (Unsigned right shift) = Shifts bits right, fills with 0.