Java Programming Assignment

Section 1: Java Data Types

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

   Java has 8 primitive data types:
   byte – It's the smallest integer type. It takes 1 byte of memory.
   short – Slightly bigger than byte, takes 2 bytes of memory.
   int – This is the default integer type. It takes 4 bytes.
   long – Used for very large integer values. It takes 8 bytes.
   float – A decimal type with less precision. It takes 4 bytes.
   double – More precise decimal type. It takes 8 bytes.
   char – Used to store a single character like 'A'. It takes 2 bytes.
   boolean – Stores either true or false. It's just 1 bit, but handled in 1 byte internally.

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

   Primitive data types are the basic types like int, float, char, etc. They store simple values and are already predefined in Java.
   Non-primitive data types include arrays, classes, interfaces, and Strings. They can store multiple values and you can call methods on them. Also, they are created by the programmer or Java libraries, not built-in like primitives.

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

```
public class PrimitiveTypesDemo {
  public static void main(String[] args) {
    byte b = 100;
    short s = 10000;
    int i = 100000;
    long l = 10000000000L;
    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.

   Type casting means converting one data type into another.
   There are two types:
   Implicit Casting (Widening, automatically):
   This happens automatically when you convert a smaller data type to a larger one.
   Example for implicit typecasting:
   int x = 10;
   double y = x;
   System.out.println(y);   Output:10.0
   Explicit Casting (Narrowing):
   This happens manually when converting a larger type to a smaller one.
   Example for explicit typecasting:
   double a = 9.78;
   int b = (int) a;
   System.out.println(b);   Output: 9

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

   Default values of primitive data type like byte, int, float, double, long, short is 0 and for char it is null and for Boolean it is false.

Section 2: Java Control Statements

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

   Control statements are used in Java to control the flow of execution in a program like deciding, repeating, or jumping to a certain part of code based on some conditions.
   They are mainly of three types:
   1. Decision-making statements:
   Used to make choices in code:

I. if
II. if-else
III. if-else-if
IV. switch

   2. Looping statements:
   Used to repeat a block of code:

I. for
II. while
III. do-while

3. Jumping statements:

Used to jump out of or skip code:

I. break
II. continue
III. return

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

If-else example:

```
public class IfElseSwitchDemo {
   public static void main(String[] args) {
        int age = 25;
       if (age < 18) {
          System.out.println("Eligible for voting.");
       } else if (age >= 18 {
          System.out.println("Not Eligible for voting.");
       }
}
```

Switch-Case Example:

```
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;

 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;
          default:
            System.out.println("Invalid day number");
        }
    }
```

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

Break and continue are used inside loops, but they do different things:

- break completely exits the loop.
- continue skips only the current iteration and goes to the next one.

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

```
public class Task1 {
public static void main(String[] args) {
for(int i=2;i<=50;i++)
{
        if(i%2==0)
        System.out.println(i);
        }
}
}
```

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

while loop:

Condition is checked first, then the block runs.

If the condition is false at the beginning, the loop might not run at all.

Java program example:

```
int i = 5;
while (i < 5) {
   System.out.println("Inside while loop");
   i++;
}
```

do-while loop:

Block runs first, then the condition is checked.

It runs at least once, even if the condition is false.

```
int i = 5;
do {
   System.out.println("Inside do-while loop");
   i++;
} while (i < 5);
```

Section 3: Java Keywords and Operators

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

Keywords in Java are reserved words that have a special meaning in the language. You can't use them as variable names, class names, or method names because Java already uses them.
Here are 10 commonly used keywords:
Int, class, public, static, void, if, else, return final, new

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

   static makes members belong to a class, not instances, useful for shared data.
   final makes members (variables, methods, classes) unchangeable after initialization/definition.
   this refers to the current object instance, and
   super refers to the parent class, often for accessing overridden methods or constructors.

3. What are the types of operators in Java?

   Java supports several types of operators:

Arithmetic Operators: These operators perform mathematical calculations like addition (+), subtraction (-), multiplication (*), division (/), and modulus (%).

Assignment Operators: Used to assign values to variables, with the most basic being the equals sign (=).

Relational Operators: Used to compare values and determine the relationship between them. Examples include equals (==), not equals (!=), greater than (>), less than (<), greater than or equal to (>=), and less than or equal to (<=).

Logical Operators: Used to combine or modify conditional statements. Common logical operators are AND (&&), OR (||), and NOT (!)

Bitwise Operators: These operators work on the individual bits of their operands, including AND (&), OR (|), XOR (^), NOT (~), left shift (<<), right shift (>>), and unsigned right shift (>>>).

Unary Operators: These operators require only one operand and include increment (++), decrement (--), plus (+), minus (-), and logical NOT (!).

Ternary Operator: This is a shorthand way of writing an if-else statement, using the syntax condition ? expression1 : expression2

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

Arithemetic Operation Example:

```
public class ArithmeticDemo {
  public static void main(String[] args) {
    int a = 10, b = 5;
    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 Operator Example:

```
public class RelationalDemo {
  public static void main(String[] args) {
    int a = 10, b = 5;
    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 Operator Example:

```
public class LogicalDemo {
  public static void main(String[] args) {
    boolean x = true, y = false;

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

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

    System.out.println("!x    : " + (!x));    // NOT

  }

}
```

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

Operator precedence means the **order in which operators are evaluated** when there are multiple operators in an expression.

Example:
In the expression 5 + 3 * 2, multiplication happens first, then addition.
So result is 5 + 6 = 11, **not** 16.

Because * has higher precedence than +.