**JDK:**

In a top-down model of JDK,JRE, and JVM, we can visualise them as layers, where each higher layer contains the lower one.

1. **JDK (Java Development Kit)** - The topmost layer:

- This is the full package used for developing Java applications. It includes the tools for both writing and running Java code.

- **Contains:** JRE and additional development tools (like the compiler).

2. **JRE (Java Runtime Environment)** - The middle layer:

- This is what you need to run Java programs. It doesn't have tools for writing or compiling Java code, just for running it.

- **Contains:** JVM and libraries required to run Java programs.

3. **JVM (Java Virtual Machine)** - The bottommost layer:

- This is where the actual execution of Java bytecode happens. The JVM is responsible for converting the bytecode into machine-readable instructions.

- **Contained within:** The JRE.

The JDK contains the JRE, which in turn contains the JVM.

**UNICODE:**

Unicode is a global character encoding standard that assigns a unique code (called a code point) to every character, symbol, and emoji from all languages and scripts. It ensures consistent text representation across platforms and devices. Unicode supports multiple encoding formats, like UTF-8, UTF-16, and UTF-32, each using different byte sizes for characters.

**TYPE-CASTING:**

Typecasting in Java means converting one data type into another. There are two kinds:

**1. Automatic Typecasting (Widening):**

- Happens when a smaller type is converted to a larger type automatically.

- Example: Converting an int to a double:

java

int num = 5;

double decimal = num;

**2. Manual Typecasting (Narrowing):**

- You do it manually when converting a larger type to a smaller type, as it may cause data loss.

- Example: Converting a double to an int:

java

double decimal = 5.7;

int num = (int) decimal;

**PROGRAM-1:**

Program for hello world and datatypes.

class firstprogram {

public void display() {

int a = 10;

System.out.println(6+a);

}

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

System.out.println("Hello World");

firstprogram obj = new firstprogram();

obj.display();

int a = 10;

int d = 7;

int e = a+d;

byte b = 127;

short c = 128;

double g = 12.34;

float f = 12.34f;

char ch = 'A';

System.out.println("The sum of "+a+" and "+d+" is "+e);

System.out.println(b+" "+c+" "+g+" "+f+" "+ch);

}

}

**PROGRAM-2:**

Program for static variable and constructors and objects.

public class StaticExample {

static int count = 0;

public **StaticExample**(){

count++;

}

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

StaticExample obj1 = new StaticExample();

System.out.println(count);

StaticExample obj2 = new StaticExample();

System.out.println(count);

count = 0;

System.out.println(count);

StaticExample obj3 = new StaticExample();

System.out.println(count);

StaticExample obj4 = new StaticExample();

System.out.println(count);

}

}

**PROGRAM-3:**

Program for Objects and parameterized constructors and un-parameterized constructors and set values and printing values.

public class Student {

public String name;

public String id;

public int age;

public long phoneNumber;

public String department;

public void **printDetails**() {

System.out.println("Name: " + name);

System.out.println("ID: " + id);

System.out.println("Age: " + age);

System.out.println("Phone Number: " + phoneNumber);

System.out.println("Department: " + department);

}

public void **setDetails**(String *name*, String *id*,int *age*, long *phoneNumber*,String *department*) {

this.name = *name*;

this.id = *id*;

this.age = *age*;

this.phoneNumber = *phoneNumber*;

this.department = *department*;

}

public **Student**() {

name = "Christopher";

id = "1234567890";

age = 20;

phoneNumber = 1234567890;

department = "IT";

printDetails();

}

public **Student**(String *name*, String *id*,int *age*, long *phoneNumber*,String *department*) {

this.name = *name*;

this.id = *id*;

this.age = *age*;

this.phoneNumber = *phoneNumber*;

this.department = *department*;

printDetails();

}

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

Student obj = new Student();

System.out.println();

Student obj2 = new Student("Daniel", "9087654321", 21, 9087654321L, "IT");

}}

**PROGRAM-4:**

Program to understand instanceOf() built-in function.

class InstanceOfFunction {

}

public class Instance extends InstanceOfFunction{

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

Instance obj = new Instance();

System.out.println(obj instanceof InstanceOfFunction);

}

}

**PROGRAM-5:**

Program to understand typecasting.

public class WideningTypeCasting {

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

byte originalByte = 10;

System.out.println("Original byte value: " + originalByte);

short byteToShort = (short)originalByte;

System.out.println("Short value: " + byteToShort);

char shortToChar = (char)byteToShort;

System.out.println("Char value: " + shortToChar);

int charToInt = (int)byteToShort;

System.out.println("Int value: " + charToInt);

long intToLong = (long)charToInt;

System.out.println("Long value: " + intToLong);

float longToFloat = (float)intToLong;

System.out.println("Float value: " + longToFloat);

double floatToDouble = (double)longToFloat;

System.out.println("Double value: " + floatToDouble);

double intToDouble = (double)charToInt;

System.out.println("Double value: " + intToDouble);

int bigInt = 1234567890;

double bigIntToDouble = (double)bigInt;

System.out.println("Double value: " + bigIntToDouble);

byte small = 10;

short medium = 100;

int res = small + medium;

System.out.println("Result: " + res);

}

}

**PROGRAM-6:**

Program to understand Logical Operators.

public class LogicalOperators {

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

*// Basic boolean variables*

System.out.println("1. Basic Logical Operations:");

System.out.println(" AND: true && true = " + (true && true));

System.out.println(" AND: true && false = " + (true && false));

System.out.println(" AND: false && true = " + (false && true));

System.out.println(" AND: false && false = " + (false && false));

System.out.println(" OR: true || true = " + (true || true));

System.out.println(" OR: true || false = " + (true || false));

System.out.println(" OR: false || true = " + (false || true));

System.out.println(" OR: false || false = " + (false || false));

System.out.println(" NOT: !true = " + (!true));

System.out.println(" NOT: !false = " + (!false));

System.out.println("\n2. Short-circuit Evaluation:");

System.out.println(" false && (1/0 > 0) = " + (false && (1/0 > 0)));

System.out.println(" true || (1/0 > 0) = " + (true || (1/0 > 0)));

System.out.println("\n3. Precedence:");

System.out.println(" true || false && false = " + (true || false && false));

System.out.println(" (true || false) && false = " + ((true || false) && false));

System.out.println("\n4. With Comparison Operators:");

int x = 7, y = 11;

System.out.println(" (x < y) && (y > 0) = " + ((x < y) && (y > 0)));

System.out.println(" (x > y) || (y < 20) = " + ((x > y) || (y < 20)));

System.out.println("\n5. Complex Conditions:");

boolean a = true, b = false, c = true;

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

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

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

System.out.println(" !a || (b && !c) = " + (!a || (b && !c)));

System.out.println("\n6. Bitwise vs. Logical Operators:");

System.out.println(" true & false = " + (true & false));

System.out.println(" true | false = " + (true | false));

System.out.println(" true ^ false = " + (true ^ false));

System.out.println("\n7. Short-circuit vs. Non-short-circuit:");

int i = 0;

boolean result1 = (false && (++i > 0)); *// i is not incremented*

boolean result2 = (false & (++i > 0)); *// i is incremented*

System.out.println(" Short-circuit AND result: " + result1 + ", i = " + i);

System.out.println(" Non-short-circuit AND result: " + result2 + ", i = " + i);

System.out.println("\n8. Logical Operators with Non-boolean Operands:");

System.out.println(" (1 < 2) && (3 < 4) = " + ((1 < 2) && (3 < 4)));

System.out.println(" ('a' < 'b') || ('c' > 'd') = " + (('a' < 'b') || ('c' > 'd')));

System.out.println("\n9. Logical Operators in Control Structures:");

if (true && !false) {

System.out.println(" This will be printed.");

}

int j = 0;

while (j < 3 && true) {

System.out.println(" j = " + j);

j++;

}

System.out.println("\n10. Logical Operators with Method Calls:");

System.out.println(" isPositive(5) && isEven(4) = " + (isPositive(5) && isEven(4)));

System.out.println(" isPositive(-3) || isEven(7) = " + (isPositive(-3) || isEven(7)));

System.out.println("\n11. Logical Operators with Null Checks:");

String str = null;

System.out.println(" (str != null) && (str.length() > 0) = " + ((str != null) && (str.length() > 0))); *// Safe null check*

*// System.out.println(" (str.length() > 0) && (str != null) = " + ((str.length() > 0) && (str != null))); // This would throw NullPointerException*

System.out.println("\n12. Using Logical Operators for Conditional Assignment:");

int value = true ? 1 : 0;

System.out.println(" value = " + value);

System.out.println("\n13. Logical Operators in Lambda Expressions:");

java.util.function.Predicate<Integer> isPositiveAndEven = *n* -> *n* > 0 && *n* % 2 == 0;

System.out.println(" Is 6 positive and even? " + isPositiveAndEven.test(6));

System.out.println(" Is 5 positive and even? " + isPositiveAndEven.test(5));

}

private static boolean **isPositive**(int *n*) {

return *n* > 0;

}

private static boolean **isEven**(int *n*) {

return *n* % 2 == 0;

}

}

**PRACTICE APPLICATION:**

Program during practice session.

import java.util.\*;

public class PayRollCalculator {

public static double monthlySalary=0;

public static int **detectSalary**(int *hours*){

return (8-*hours*)\*100;*//detecting 100/- per hour*

}

public static int **overTime**(int *hours*){

return (*hours*-8)\*400;*//adding 400/- per hour*

}

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

Scanner sc=new Scanner(System.in);

for(int i=1;i<=22;i++){

double salary=2500;

System.out.println("Enter the number of hours worked in "+i+" day:");

int hours=sc.nextInt();

if(hours==8){

System.out.println("Salary is:"+String.valueOf(salary));

monthlySalary+=salary;

}

else if(hours>=0 && hours<8){

int res=detectSalary(hours);

System.out.println("Amount detected is:"+String.valueOf(res));

salary-=res;

System.out.println("Salary is:"+String.valueOf(salary));

monthlySalary+=salary;

}

else if(hours>8){

int res=overTime(hours);

System.out.println("Amount added is:"+String.valueOf(res));

salary+=res;

System.out.println("Salary is:"+String.valueOf(salary));

monthlySalary+=salary;

}

}

sc.close();

System.out.println("Monthly Salary is:"+monthlySalary);

}

}