**Core Java -Assignment1**

### 1. Swap Two Variables Without Using a Third Variable

public class SwapWithoutThirdVariable {

public static void main(String[] args) {

int a = 5, b = 10;

// Swap using arithmetic operations

a = a + b; // a becomes 15

b = a - b; // b becomes 5

a = a - b; // a becomes 10

System.out.println("a = " + a); // 10

System.out.println("b = " + b); // 5

}

}

### 2. Type Promotion in Expressions Involving Mixed Data Types

public class TypePromotion {

public static void main(String[] args) {

int intVar = 5;

float floatVar = 5.5f;

double doubleVar = 6.7;

char charVar = 'A'; // ASCII value 65

// Type promotion in mixed expressions

double result = intVar + floatVar + doubleVar + charVar;

System.out.println("Result of type promotion: " + result); // Result will be a double

}

}

### 3. Reverse the Bits of an Integer

public class ReverseBits {

public static void main(String[] args) {

int num = 5; // Binary: 101

int reversed = 0;

while (num > 0) {

reversed = (reversed << 1) | (num & 1);

num >>= 1;

}

System.out.println("Reversed bits: " + reversed);

}

}

### 4. Check Whether a Number is Power of Two Using Bitwise Operators

public class PowerOfTwo {

public static void main(String[] args) {

int num = 16;

if (num > 0 && (num & (num - 1)) == 0) {

System.out.println(num + " is a power of 2.");

} else {

System.out.println(num + " is not a power of 2.");

}

}

}

### 5. Evaluate a Mathematical Expression Given as a String and Return the Result

import java.util.Stack;

public class ExpressionEvaluator {

public static void main(String[] args) {

String expression = "3 + 2 \* ( 2 + 5 )";

System.out.println("Result: " + evaluate(expression));

}

public static int evaluate(String expression) {

Stack<Integer> values = new Stack<>();

Stack<Character> ops = new Stack<>();

for (int i = 0; i < expression.length(); i++) {

char ch = expression.charAt(i);

if (ch == ' ') continue;

if (Character.isDigit(ch)) {

int num = 0;

while (i < expression.length() && Character.isDigit(expression.charAt(i))) {

num = num \* 10 + (expression.charAt(i) - '0');

i++;

}

values.push(num);

i--;

} else if (ch == '(') {

ops.push(ch);

} else if (ch == ')') {

while (ops.peek() != '(') {

values.push(applyOp(ops.pop(), values.pop(), values.pop()));

}

ops.pop();

} else if (ch == '+' || ch == '-' || ch == '\*' || ch == '/') {

while (!ops.isEmpty() && precedence(ops.peek()) >= precedence(ch)) {

values.push(applyOp(ops.pop(), values.pop(), values.pop()));

}

ops.push(ch);

}

}

while (!ops.isEmpty()) {

values.push(applyOp(ops.pop(), values.pop(), values.pop()));

}

return values.pop();

}

public static int applyOp(char op, int b, int a) {

switch (op) {

case '+': return a + b;

case '-': return a - b;

case '\*': return a \* b;

case '/': return a / b;

}

return 0;

}

public static int precedence(char op) {

if (op == '+' || op == '-') return 1;

if (op == '\*' || op == '/') return 2;

return 0;

}

}

### 6. Sum of Digits of a Given Number Using a While Loop

public class SumOfDigits {

public static void main(String[] args) {

int number = 12345;

int sum = 0;

while (number != 0) {

sum += number % 10;

number /= 10;

}

System.out.println("Sum of digits: " + sum);

}

}

### 7. Check Whether a Given Number is Prime Using a For Loop

public class PrimeNumber {

public static void main(String[] args) {

int num = 29;

boolean isPrime = true;

for (int i = 2; i <= Math.sqrt(num); i++) {

if (num % i == 0) {

isPrime = false;

break;

}

}

if (isPrime && num > 1) {

System.out.println(num + " is a prime number.");

} else {

System.out.println(num + " is not a prime number.");

}

}

}

### 8. Generate the First N Fibonacci Numbers Using a Do-While Loop

public class FibonacciDoWhile {

public static void main(String[] args) {

int N = 10;

int a = 0, b = 1;

int count = 0;

do {

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

int next = a + b;

a = b;

b = next;

count++;

} while (count < N);

}

}

### 9. Print a Pattern Using Nested Loop

public class PrintPattern {

public static void main(String[] args) {

int n = 5;

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

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

System.out.print("\* ");

}

System.out.println();

}

}

}

### 10. Custom Immutable Class to Store Configurations

public final class Config {

private final String host;

private final int port;

public Config(String host, int port) {

this.host = host;

this.port = port;

}

public String getHost() {

return host;

}

public int getPort() {

return port;

}

@Override

public String toString() {

return "Host: " + host + ", Port: " + port;

}

}

### 11. Convert RGB to Grayscale Using Bitwise Operations

public class RGBtoGrayscale {

public static int convertToGrayscale(int r, int g, int b) {

int gray = (int)(0.3 \* r + 0.59 \* g + 0.11 \* b);

return (gray << 16) | (gray << 8) | gray; // Grayscale color

}

public static void main(String[] args) {

int r = 255, g = 0, b = 0; // Red color

int grayscale = convertToGrayscale(r, g, b);

System.out.println("Grayscale: " + Integer.toHexString(grayscale));

}

}

### 12. Encode an Integer Using Bitwise Operations

public class IntegerCompression {

public static int encode(int number) {

return number ^ (number >> 16); // XOR with shifted version

}

public static void main(String[] args) {

int number = 123456;

int encoded = encode(number);

System.out.println("Encoded: " + encoded);

}

}

### 13. Discount Calculator

import java.util.Scanner;

public class DiscountCalculator {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Enter the original price:");

double originalPrice = scanner.nextDouble();

System.out.println("Enter the discount percentage:");

double discount = scanner.nextDouble();

double finalPrice = originalPrice - (originalPrice \* discount / 100);

System.out.println("The final price after discount is: " + finalPrice);

}

}

### 14. ATM Withdrawal Function

import java.util.Scanner;

public class ATMWithdrawal {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Enter your current balance:");

double balance = scanner.nextDouble();

System.out.println("Enter the withdrawal amount:");

double withdrawalAmount = scanner.nextDouble();

if (withdrawalAmount <= balance) {

balance -= withdrawalAmount;

System.out.println("Withdrawal successful! New balance: " + balance);

} else {

System.out.println("Insufficient funds!");

}

}

}

### 15. Multiplication Table of N Using All Three Loops

import java.util.Scanner;

public class MultiplicationTable {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Enter the number for multiplication table:");

int N = scanner.nextInt();

// Using for loop

System.out.println("Using for loop:");

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

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

}

// Using while loop

System.out.println("Using while loop:");

int i = 1;

while (i <= 10) {

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

i++;

}

// Using do-while loop

System.out.println("Using do-while loop:");

i = 1;

do {

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

i++;

} while (i <= 10);

}

}