1. Arithmetic & Assignment Operators

Q1: Write a program to swap two numbers **without using a third variable** and without using arithmetic operators like + or - **.**
Hint : Use bitwise XOR ^ operator.

```
public class Q01{
	public static void main(String[] args){
		int a = 4;
		int b = 6;
		System.out.println("Before Swapping: \nA = " + a + "\tB = " + b);
		a = a ^ b;
		b = a ^ b;
		a = a ^ b;
		System.out.println("After Swapping: \nA = " + a + "\tB = " + b);
	}
}
```

Q2: Write a program to check whether a given number **is even or odd** using only **bitwise operators .**
Hint : Use n & 1 to check.

```
public class Q02{
	public static void main(String[] args){
		int a = 9, b = 10;
		check(a);
		check(b);
	}

	public static void check(int a){
		if((a&1) == 0){
			System.out.println(a + " is even");
		} else {
			System.out.println(a + " is odd");
		}
	}
}
```

Q3: Implement a program that calculates the **sum of digits of an integer** using **modulus (%) and division (/) operators .**

```
public class Q03{
	public static void main(String[] args){
		int x = 123456;
		int res = 0;
		while(x!=0){
			res = res + (x%10);
			x = x/10;
		}
		System.out.println("Sum: " + res);
	}
}
```

Q4: Write a program to find whether a given number is **divisible by 3** without using the modulus (%) or division (/) operators.
Hint : Use **subtraction and bitwise shifts**

```
public class Q04{
	public static void main(String[] args){
		int n = 15;

		if(n<0){
			n = -n;
		}

		while(n>3){
			n = (n & 3) + (n >> 2);
		}

		if(n==0 || n==3)
			System.out.println("Division by 3");
		else
			System.out.println("Not division by 3");

	}
}
```

Q5: Write a Java program to **swap two numbers** using the += and -= operators only.

```
public class Q05{
	public static void main(String[] args){
		int x = 3, y = 9;
		System.out.println("Before Swapping: \nX = " + x + "\nY = " + y);
		x += y;
		y -= x;
		y = -y;
		x -= y;
		System.out.println("After Swapping: \nX = " + x + "\nY = " + y);
	}
}
```

2. Relational & Logical Operators

Q6: Write a program to find the **largest of three numbers** using only the **ternary operator (? :) .**

```
public class Q06{
	public static void main(String[] args){
		int x=8, y=4, z=1;
		int res = ((x>y && x>z)? x : (y>x && y>z)? y : z);
		System.out.println(res + " is the largest number.");
	}
}
```

Q7: Implement a Java program that checks whether a given year is a **leap year or not** using **logical (&& , ||) operators .**

```
import java.util.Scanner;
public class Q07{
	public static void main(String[] args){
		Scanner sc = new Scanner (System.in);
		int y = sc.nextInt();
		if(y%400==0 || (y%4==0 && y%100!=0))
			System.out.println(y + " is a leap year.");
		else
			System.out.println(y + " is not a leap year.");
	}
}
```

Q8: Write a program that **takes three boolean inputs** and prints true if at least two of them are true . **Hint :** Use logical operators (&& , ||)

```
public class Q08{
	public static void main(String[] args){
		boolean x = true, y = false, z = true;
		check(x,y,z);
		x = true; y = false; z = false;
		check(x,y,z);
	}

	public static void check(boolean a, boolean b, boolean c){
		if((a && (b||c)) || (c && (a||b)) || (b && (a||c))){
			System.out.println("true");
		} else {
			System.out.println("false");
		}
	}
}
```

Q9: Implement a Java program that checks if a number is **within a specific range (20 to 50)** without using if-else .
Hint : Use **logical AND (&&) in a print statement .**

```
public class Q09{
	public static void main(String[] args){
		int x = 40, y = 52;
		check(x);
		check(y);
	}

	public static void check(int a){
		System.out.println((a>=20 && a<=50)? a + " lies in the range." : a + " doesn't
lie within the range");
	}
}
```

Q10: Write a program to determine if a **character is a vowel** or a consonant using the ternary operator.

```
public class Q10{
	public static void main(String[] args){
		char x = 'a';
		check(x);
		x = 'q';
		check(x);
		x = 'A';
		check(x);
		x = 'R';
		check(x);
	}

	public static void check(char a){
		char ch = Character.toLowerCase(a);
		String str = (ch=='a' || ch=='e' || ch=='i' || ch=='o' || ch =='u')? a + " is a vowel."
: a + " is a consonant";
		System.out.println(str);
	}
}
```

3. Bitwise Operators

Q11: Write a program to check if a given number is a **power of 2** using bitwise operators.
Hint : n & (n - 1) == 0 for positive numbers.

```
public class Q11{
	public static void main(String[] args){
		check(8);
		check(7);
	}

	public static void check(int a){
		if(a<0) a = -a;
		if((a & (a-1))==0)
			System.out.println(a + " is a power of two");
		else
			System.out.println("Oops! " + a + " is not a power of two");
	}
}
```

Q12: Write a Java program to **multiply a number by 8** without using * or / operators**.**
Hint : Use bitwise left shift (<<).

```
public class Q12{
	public static void main(String[] args){
		check(7);
		check(20);
		check(-9);
	}

	public static void check(int a){
		//Use bitwise left shift (  <<  )
		int result = a << 3;
		System.out.println(result);
	}
}
```

Q13: Implement a Java program to find the **absolute value** of an integer using bitwise operators.
Hint : mask = num >> 31; abs = (num + mask) ^ mask;

```
public class Q13{
	public static void main(String[] args){
		abs(7);
		abs(-20);
	}

	public static void abs(int num){
		int mask = num >> 31;
		int abs = (num + mask) ^ mask;
		System.out.println(abs);
	}
}

// 00000000 00000000 00000000 00010100
// 11111111 11111111 11111111 11101011
// 11111111 11111111 11111111 11101100
// 11111111 11111111 11111111 11111111
// 00000000 00000000 00000000 00010101
// 11111111 11111111 11111111 11101010

// 11111111 11111111 11111111 11101011
// 11111111 11111111 11111111 11111111
// 00000000 00000000 00000000 00010100
```

Q14: Write a program to count the **number of 1s (set bits)** in a binary representation of a number using bitwise operations.
Hint : Use n & (n - 1)

```
public class Q14{
        public static void main(String[] args){
                setbits(7);
                setbits(12);
        }

        public static void setbits(int num){
                int count = 0;
                while(num > 0){
        num = (num & (num - 1));
        count++;
      }
                System.out.println(count);
        }
}

// 0111 0110
// 0110 1
// 0110 0101
// 0100 2
// 0100 0011
// 0000 3
```

4. Ternary Operator Challenges

Q15: Implement a program to swap **odd and even bits** of a number using bitwise operators**.**
Hint : Use masks: (x & 0xAAAAAAAA) >> 1 | (x & 0x55555555) << 1

```
public class Q15{
        public static void main(String[] args){
                swapper(10);
                swapper(9);
        }

        public static void swapper(int x){
           int evenBits = (x & 0xAAAAAAAA) >> 1; // Get even bits and shift right
      int oddBits = (x & 0x55555555) << 1;   // Get odd bits and shift left
      int res = (evenBits | oddBits);
                System.out.println(res);
```

```
	}
}

// 1001
// 10101010 10101010 10101010 10101010
//                         1001
// 00000000 00000000 00000000 00001000
// 00000000 00000000 00000000 00000100

// 01010101 01010101 01010101 01010101
//                         1001
// 00000000 00000000 00000000 00000001
// 00000000 00000000 00000000 00000010
// 00000000 00000000 00000000 00000010
// 00000000 00000000 00000000 00000100
// 00000000 00000000 00000000 00000110
```

Q16: Write a program that determines whether a given number is **positive, negative, or zero** using only the **ternary operator .**

```
public class Q16{
	public static void main(String[] args){
		pnz(12);
		pnz(-11);
		pnz(0);
	}
	public static void pnz(int x){
		String res = (x==0)? "zero" : (x>0)? "positive" : "negative";
		System.out.println(res);
	}
}
```

Q17: Implement a Java program that finds the **minimum of four numbers** using nested ternary operators.

```
public class Q17{
	public static void main(String[] args){
		minfour(1,2,3,4);
	}

	public static void minfour(int x, int y, int z, int m){
		int res = (x<y && x<z && x<m)? x:(z<y && z<x && z<m)? z:(y<z && y<x &&
y<m)? y : m;
		System.out.println(res);
	}
}
```

Q18: Given a student's percentage, print "Pass" if the percentage is 40 or above; otherwise, print "Fail" , using only the ternary operator.

```
public class Q18{
	public static void main(String args[]){
		check(50);
		check(39);
	}

	public static void check(int a){
		String result = (a>=40)? "Pass" : "Fail";
		System.out.println(result);
	}
}
```

Q19: Write a Java program that checks whether a character is uppercase, lowercase, or not a letter using only the ternary operator.

```
public class Q19{
	public static void main(String args[]){
		check('a');
		check('A');
		check('1');
	}

	public static void check(char A){
		int a = (int) A;
		String result = (a>64 && a<91)? "Uppercase" : (a>96 && a<123)?
"Lowercase" : "Not a character";
		System.out.println(result);
	}
}}
```

Q20: Implement a Java program that returns the absolute value of a given number using the ternary operator (without using Math.abs()

```
public class Q20{
	public static void main(String args[]){
		check(20);
		check(-120);
	}

	public static void check(int A){
		int result = (A>0)? A : -A;
		System.out.println("absolute value: " + result);
	}
}
```

5. Miscellaneous Operator Questions

Q21: Write a program that increments a number without using + or ++ operators. Hint : Use bitwise - (~x)

```
public class Q21{
	static public void main(String me[]){
		int x = 5;
		System.out.println(-~x);
	}
}
```

Q22: Implement a calculator that takes two numbers and an operator (+ , - , * , /) as input and prints the result using only switch-case .

```
import java.util.Scanner;
public class Q22{
	public static void main(String[] args){
		Scanner sc = new Scanner (System.in);
		System.out.print("Enter first number: ");
		int a = sc.nextInt();
		System.out.print("Enter second number: ");
		int b = sc.nextInt();
		System.out.println("MENU: ");
		System.out.println("1. ADDITION ");
		System.out.println("2. SUBTRACTION ");
		System.out.println("3. MULTIPLICATION ");
		System.out.println("4. DIVISION ");
		System.out.println("5. FIND REMAINDER ");
		System.out.print("Enter your Option: ");
		int c = sc.nextInt();
		while(c!=6){
			switch(c){
				case 1: System.out.println("Addition of " + a + " + " + b + " = " +
(a+b));
					break;
				case 2: System.out.println("Difference between " + a + " - " + b
+ " = " + (a-b));
					break;
				case 3: System.out.println("Multiplication of " + a + " x " + b + "
= " + (a*b));
					break;
```

```
                                        case 4: System.out.println("Quotient in division of " + a + " / " +
b + " = " + ((float)a/(float)b));
                                                    break;
                                        case 5: System.out.println("Remainder in division of " + a + "
and " + b + " = " + ((float)a%(float)b));
                                                    break;
                                        default:System.out.println("INVALID OPTION");
                                                    break;
                              }
                              System.out.print("Enter your Option: ");
                              c = sc.nextInt();
                    }
                    System.out.println("EXITING PROGRAMMING!");
          }
}
```

Q23: Given a number, find whether it is odd or even using the & bitwise operator and print the result without using if-else

```
public class Q23{
          public static void main(String[] args){
                    int a = 9, b = 10;
                    check(a);
                    check(b);
          }

          public static void check(int a){
                    if((a&1) == 0){
                              System.out.println(a + " is even");
                    } else {
                              System.out.println(a + " is odd");
                    }
          }
}
```

Q24: Write a program that prints all even numbers from 1 to 100 using only bitwise AND (&) and for loop.

```
public class Q24{
   public static void main(String[] args) {
      for (int i = 1; i <= 100; i++) {
         if ((i & 1) == 0) {
            System.out.print(i + " ");
         }
      }
   }
}
```

Q25: Implement a program that reverses an integer number without using string conversion (StringBuilder or toCharArray). Hint : Use while(n!=0) { rev = rev * 10 + n % 10; n /= 10; }

```
public class Q25{
	static public void main(String me[]){
		int x = 1534, rev = 0;
		int temp = x;
		while(x!=0){
			rev = rev * 10 + x % 10;
			x = x/10;
		}
		System.out.println("Reverse of " + temp + " is " + rev);
	}
}
```