Set 1: Programs 1-10

1. Even or Odd Number

```
fun main() {
    val num = 10
    if (num % 2 == 0) println("$num is Even") else println("$num is Odd")
}
```

2. Largest of Two Numbers

```
fun main() {
    val a = 10
    val b = 20
    if (a > b) {
        println("$a is larger")
    } else {
        println("$b is larger")
    }
}
```

3. Largest of Three Numbers

```
fun main() {
    val a = 10
    val b = 20
    val c = 30
    val largest = if (a > b && a > c) a else if (b > c) b else c
    println("Largest number is $largest")
}
```

4. Check Leap Year

```
fun main() {
    val year = 2024
    val isLeap = (year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)
    if (isLeap) {
        println("$year is a Leap Year")
    } else {
        println("$year is not a Leap Year")
    }
}
```

5. Check Positive, Negative, or Zero

```
fun main() {
    val num = -5
    when {
        num > 0 -> println("$num is Positive")
        num < 0 -> println("$num is Negative")
        else -> println("$num is Zero")
    }
}
```

6. Print 1 to 10

```
fun main() {
    for (i in 1..10) {
        print("$i ")
    }
}
```

7. Sum of First 10 Natural Numbers

```
fun main() {
    var sum = 0
    for (i in 1..10) {
        sum += i
    }
    println("Sum of first 10 natural numbers is $sum")
}
```

8. Factorial of a Number

```
fun main() {
    val num = 5
    var fact = 1
    for (i in 1..num) {
        fact *= i
    }
    println("Factorial of $num is $fact")
}
```

9. Fibonacci Series

```
fun main() {
    val n = 10
    var t1 = 0
    var t2 = 1
    print("Fibonacci Series: $t1, $t2")
    for (i in 3..n) {
        val sum = t1 + t2
        print(", $sum")
        t1 = t2
        t2 = sum
    }
}
```

10. Check Prime Number

```
fun main() {
    val num = 29
    var isPrime = true

    if (num <= 1) isPrime = false
    else {
        for (i in 2..num / 2) {
            if (num % i == 0) {
                isPrime = false
                break
            }
        }
    }

    if (isPrime) {
        println("$num is a Prime Number")
    } else {
        println("$num is not a Prime Number")
    }
}
```

Set 2: Programs 11-20

11. Reverse a Number

```
fun main() {
    var num = 1234
    var reversed = 0
    while (num != 0) {
        val digit = num % 10
        reversed = reversed * 10 + digit
        num /= 10
    }
    println("Reversed Number: $reversed")
}
```

12. Palindrome Check

```
fun main() {
    val num = 121
    val original = num.toString()
    val reversed = original.reversed()
    if (original == reversed) {
        println("$num is a Palindrome")
    } else {
        println("$num is not a Palindrome")
    }
}
```

13. Armstrong Number

```
fun main() {
    val num = 153
    val digits = num.toString().map { it.toString().toInt() }
    val sum = digits.sumOf { it * it * it }
    if (sum == num) {
        println("$num is an Armstrong Number")
    } else {
        println("$num is not an Armstrong Number")
    }
}
```

14. Multiplication Table

```
fun main() {
    val num = 5
    for (i in 1..10) {
        println("$num x $i = ${num * i}")
    }
}
```

15. Star Pattern

```
fun main() {
    val n = 5
    for (i in 1..n) {
        for (j in 1..i) {
            print("* ")
        }
        println()
    }
}
```

16. Number Pyramid

```
fun main() {
    val n = 5
    for (i in 1..n) {
        for (j in 1..i) {
            print("$j ")
        }
        println()
    }
}
```

17. Floyd's Triangle

```
fun main() {
    val rows = 5
    var num = 1
    for (i in 1..rows) {
        for (j in 1..i) {
            print("$num ")
            num++
        }
        println()
    }
}
```

18. Diamond Pattern

```
fun main() {

    val n = 5

    for (i in 1..n) {

        print(" ".repeat(n - i))

        println("*".repeat(2 * i - 1))

    }

    for (i in n - 1 downTo 1) {

        print(" ".repeat(n - i))

        println("*".repeat(2 * i - 1))

    }

}
```

19. Find GCD of Two Numbers

```
fun main() {

    val a = 56

    val b = 98

    var gcd = 1

    for (i in 1..minOf(a, b)) {

        if (a % i == 0 && b % i == 0) {

            gcd = i

        }

    }

    println("GCD of $a and $b is $gcd")

}
```

20. LCM of Two Numbers

```
fun main() {

    val a = 12

    val b = 15

    val lcm = (a * b) / gcd(a, b)

    println("LCM of $a and $b is $lcm")

}


fun gcd(x: Int, y: Int): Int {

    var a = x

    var b = y

    while (b != 0) {

        val temp = b

        b = a % temp

        a = temp

    }

    return a

}
```

Set 3: Programs 21-30

21. Bubble Sort

```
fun main() {
    val arr = intArrayOf(5, 2, 9, 1, 5, 6)
    for (i in arr.indices) {
        for (j in 0 until arr.size - i - 1) {
            if (arr[j] > arr[j + 1]) {
                val temp = arr[j]
                arr[j] = arr[j + 1]
                arr[j + 1] = temp
            }
        }
    }
    println("Sorted Array: ${arr.joinToString(", ")}")
}
```

22. Selection Sort

```
fun main() {
    val arr = intArrayOf(64, 25, 12, 22, 11)
    for (i in arr.indices) {
        var minIdx = i
        for (j in i + 1 until arr.size) {
            if (arr[j] < arr[minIdx]) minIdx = j
        }
        val temp = arr[minIdx]
        arr[minIdx] = arr[i]
        arr[i] = temp
    }
    println("Sorted Array: ${arr.joinToString(", ")}")
}
```

23. Binary Search

```
fun main() {
    val arr = intArrayOf(1, 3, 5, 7, 9, 11)
    val target = 5
    val result = binarySearch(arr, target)
    println(if (result != -1) "Element found at index $result" else "Element not found")
}

fun binarySearch(arr: IntArray, target: Int): Int {
    var left = 0
    var right = arr.size - 1
    while (left <= right) {
        val mid = left + (right - left) / 2
        when {
            arr[mid] == target -> return mid
            arr[mid] < target -> left = mid + 1
            else -> right = mid - 1
        }
    }
    return -1
}
```

24. Matrix Addition

```
fun main() {
    val a = arrayOf(intArrayOf(1, 2, 3), intArrayOf(4, 5, 6))
    val b = arrayOf(intArrayOf(7, 8, 9), intArrayOf(1, 2, 3))
    val rows = a.size
    val cols = a[0].size
    val result = Array(rows) { IntArray(cols) }

    for (i in 0 until rows) {
        for (j in 0 until cols) {
            result[i][j] = a[i][j] + b[i][j]
        }
    }

    println("Matrix Addition Result:")
    result.forEach { println(it.joinToString(" ")) }
}
```

25. Matrix Multiplication

```
fun main() {
    val a = arrayOf(intArrayOf(1, 2), intArrayOf(3, 4))
    val b = arrayOf(intArrayOf(5, 6), intArrayOf(7, 8))
    val rowsA = a.size
    val colsA = a[0].size
    val colsB = b[0].size
    val result = Array(rowsA) { IntArray(colsB) }
    for (i in 0 until rowsA) {
        for (j in 0 until colsB) {
            for (k in 0 until colsA) {
                result[i][j] += a[i][k] * b[k][j]
            }
        }
    }

    println("Matrix Multiplication Result:")
    result.forEach { println(it.joinToString(" ")) }
}
```

26. Simple Interest Calculator

```
fun main() {
    val principal = 1000.0
    val rate = 5.0
    val time = 2.0
    val simpleInterest = (principal * rate * time) / 100
    println("Simple Interest: $simpleInterest")
}
```

27. ATM Simulation

```
fun main() {
    val balance = 5000.0
    val withdrawAmount = 1500.0
    if (withdrawAmount <= balance) {
        println("Transaction Successful! Remaining Balance: ${balance - withdrawAmount}")
    } else {
        println("Insufficient Balance!")
    }
}
```

28. Electricity Bill Calculator

```
fun main() {
    val units = 250
    val bill = when {
        units <= 100 -> units * 1.5
        units <= 200 -> 100 * 1.5 + (units - 100) * 2.5
        else -> 100 * 1.5 + 100 * 2.5 + (units - 200) * 3.5
    }
    println("Electricity Bill: $bill")
}
```

29. Check Voter Eligibility

```
fun main() {
    val age = 17
    if (age >= 18) {
        println("Eligible to Vote")
    } else {
        println("Not Eligible to Vote")
    }
}
```

30. BMI Calculator

```
fun main() {

    val weight = 70.0 // in kilograms

    val height = 1.75 // in meters

    val bmi = weight / (height * height)

    println("Your BMI is $bmi")

    when {

        bmi < 18.5 -> println("Underweight")

        bmi in 18.5..24.9 -> println("Normal weight")

        bmi in 25.0..29.9 -> println("Overweight")

        else -> println("Obesity")

    }

}
```