Практическая работа 2

1.

2.1.20-Бета1 JVM Аргументы программы

```kotlin
fun main() {
    val number = 27 // Замените 27 на любое двузначное число

    val десятки = number / 10
    val единицы = number % 10
    val суммаЦифр = десятки + единицы
    val произведениеЦифр = десятки * единицы

    println("Число десятков: $десятки")
    println("Число единиц: $единицы")
    println("Сумма цифр: $суммаЦифр")
    println("Произведение цифр: $произведениеЦифр")
}
```

```
Число десятков: 2
Число единиц: 7
Сумма цифр: 9
Произведение цифр: 14
```

2.

```kotlin
fun main() {
    val number = 345 // Замените 345 на любое трехзначное число

    val единицы = number % 10
    val десятки = (number / 10) % 10
    val сотни = number / 100
    val суммаЦифр = единицы + десятки + сотни
    val произведениеЦифр = единицы * десятки * сотни

    println("Число единиц: $единицы")
    println("Число десятков: $десятки")
    println("Число сотен: $сотни")
    println("Сумма цифр: $суммаЦифр")
    println("Произведение цифр: $произведениеЦифр")
}
```

```
Число единиц: 5
Число десятков: 4
Число сотен: 3
Сумма цифр: 12
Произведение цифр: 60
```

3.

```
fun main() {
    val делимое = 10.0
    val делитель = 2.0

    if (делитель != 0.0) {
        val результат = делимое / делитель
        println("Результат деления $делимое на $делитель: $результат")
    } else {
        println("Ошибка: Делитель не может быть нулем.")
    }
}
```

```
Результат деления 10,0 на 2,0: 5,0
```

4.

```
fun main() {
    val число = 2.0
    val степень = 3

    val результат = возвестиВСтепень(число, степень)
    println("$число в степени $степень равно $результат")
}

fun возвестиВСтепень(число: Double, степень: Int): Double {
    return Math.pow(число, степень.toDouble())
}
```

```
2.0 в степени 3 равно 8.0
```

5.

```
fun main() {
    val число = 16.0

    val результат = найтиКорень(число)
    println("Корень из $число равен $результат")
}

fun найтиКорень(число: Double): Double {
    if (число < 0) {
        throw IllegalArgumentException("Число должно быть неотрицательным.")
    }
    return Math.sqrt(число)
}
```

```
Корень из 16.0 равен 4.0
```

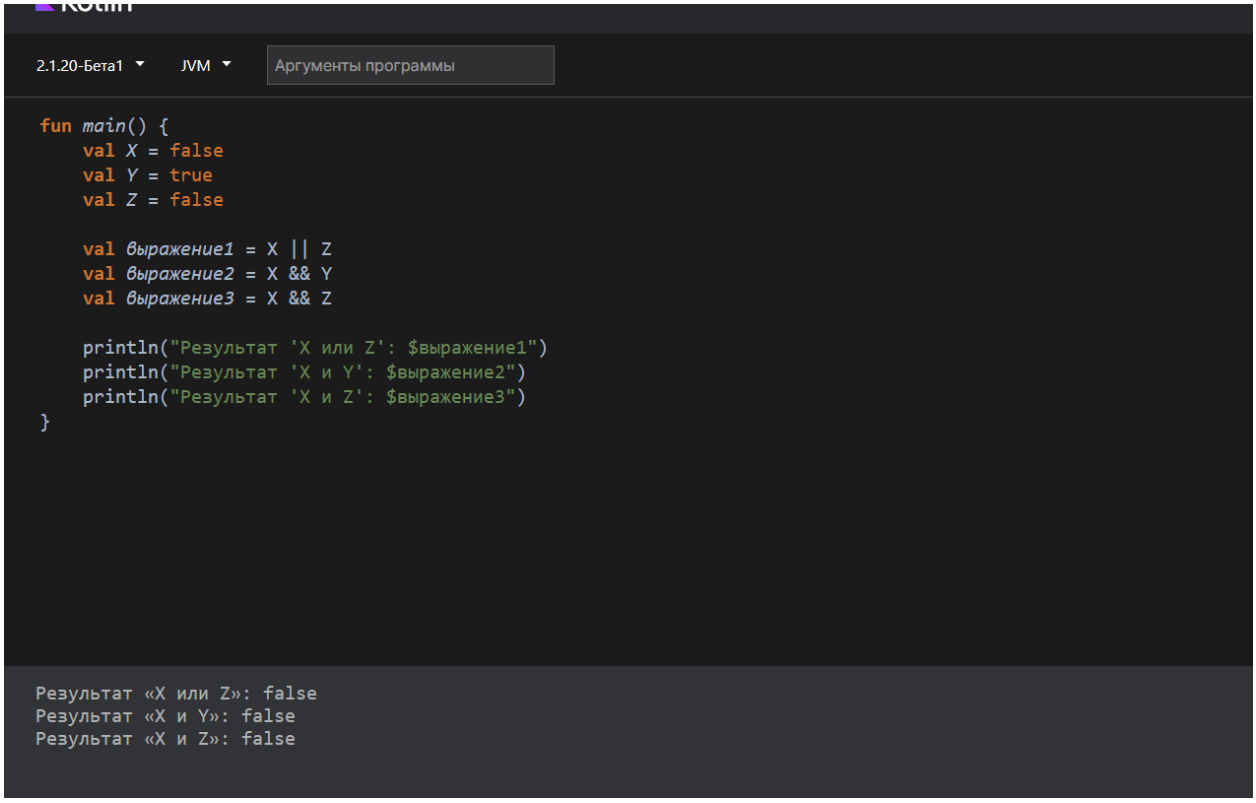
Вычисление логических выражений

1.

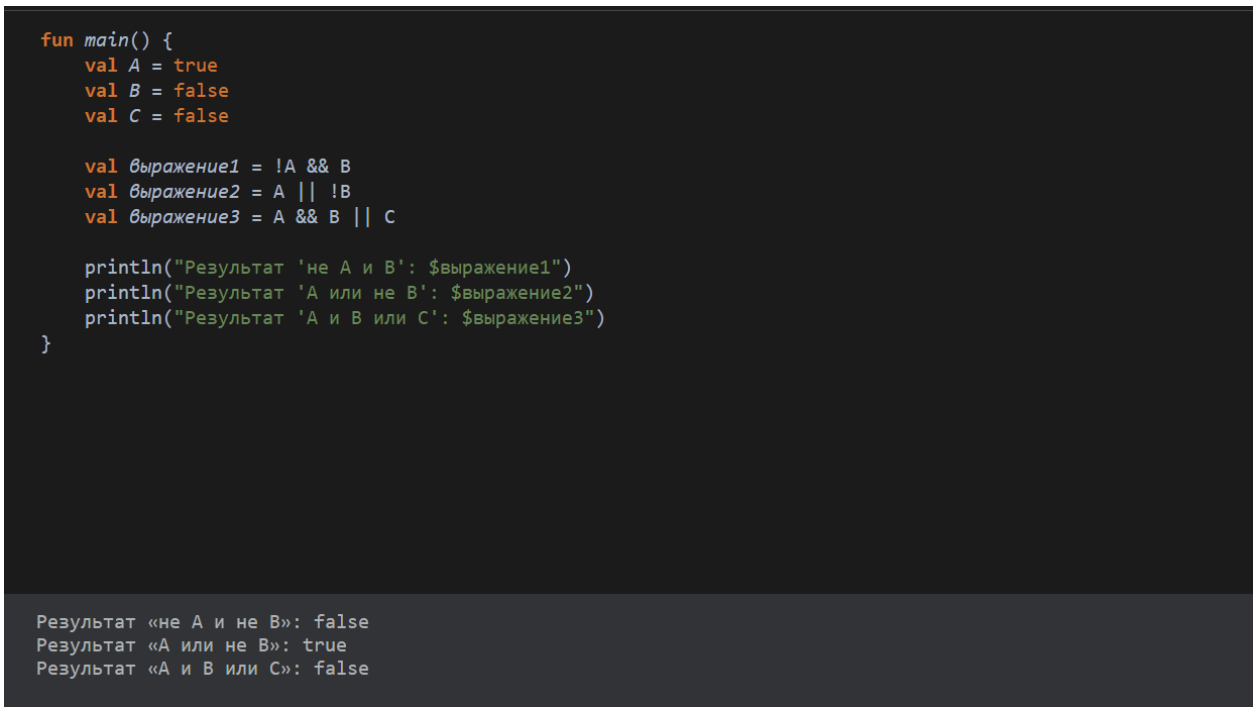
```
fun main() {
    val A = true
    val B = false
    val C = false

    val выражение1 = A || B
    val выражение2 = A && B
    val выражение3 = B || C

    println("Результат 'A или B': $выражение1")
    println("Результат 'A и B': $выражение2")
    println("Результат 'B или C': $выражение3")
}
```

```
Результат «A или B»: true
Результат «A и B»: false
Результат «B или C»: false
```

2.


2.1.20-Бета1
JVM
Аргументы программы
fun main() {
val X = false
val Y = true
val Z = false
val выражение1 = X || Z
val выражение2 = X && Y
val выражение3 = X && Z
println("Результат 'X или Z': $выражение1")
println("Результат 'X и Y': $выражение2")
println("Результат 'X и Z': $выражение3")
}
Результат «X или Z»: false
Результат «X и Y»: false
Результат «X и Z»: false

3.


fun main() {
val A = true
val B = false
val C = false
val выражение1 = !A && B
val выражение2 = A || !B
val выражение3 = A && B || C
println("Результат 'не A и B': $выражение1")
println("Результат 'A или не B': $выражение2")
println("Результат 'A и B или C': $выражение3")
}
Результат «не A и не B»: false
Результат «A или не B»: true
Результат «A и B или C»: false

4.

```
fun main() {
    val X = true
    val Y = true
    val Z = false

    val выражение1 = !X && Y
    val выражение2 = X || !Y
    val выражение3 = X || (Y && Z)

    println("Результат 'не X и Y': $выражение1")
    println("Результат 'X или не Y': $выражение2")
    println("Результат 'X или Y и Z': $выражение3")
}
```

```
Результат «не X и Y»: false
Результат «X или не Y»: true
Результат «X или Y и Z»: true
```

5.

```
fun main() {
    val X = true
    val Y = true
    val Z = false

    val выражение1 = !X && Y
    val выражение2 = X || !Y
    val выражение3 = X || (Y && Z)

    println("Результат 'не X и Y': $выражение1")
    println("Результат 'X или не Y': $выражение2")
    println("Результат 'X или Y и Z': $выражение3")
}
```

```
Результат «не X и Y»: false
Результат «X или не Y»: true
Результат «X или Y и Z»: true
```

6.

```
fun main() {
    val X = false
    val Y = false
    val Z = true

    val выражение1 = X || (Y && !Z)
    val выражение2 = X && !Y || Z
    val выражение3 = !X && !Y
    val выражение4 = X && (!Y || Z)
    val выражение5 = !(X && Z) || Y
    val выражение6 = X || !(Y || Z)

    println("Результат 'X или Y и не Z': $выражение1")
    println("Результат 'X и не Y или Z': $выражение2")
    println("Результат 'не X и не Y': $выражение3")
    println("Результат 'X и (не Y или Z)': $выражение4")
    println("Результат 'не (X и Z) или Y': $выражение5")
    println("Результат 'X или (не (Y или Z))': $выражение6")
}
```

```
Результат «X или Y и не Z»: false
Результат «X и не Y или Z»: true
Результат «не X и не Y»: true
Результат «X и (не Y или Z)»: false
Результат «не (X и Z) или Y»: true
Результат «X или (не (Y или Z))»: false
```

7.

2.1.20-Бета1 JVM Аргументы программы

```
fun main() {
    val A = true
    val B = false
    val C = false

    val выражение1 = A || !(A && B) || C
    val выражение2 = !A || A && (B || C)
    val выражение3 = (A || (B && !C)) && C

    println("Результат 'A или не (A и B) или C': $выражение1")
    println("Результат 'не A или A и (B или C)': $выражение2")
    println("Результат '(A или B и не C) и C': $выражение3")
}
```

```
Результат 'A или не (A и B) или C': true
Результат 'не A или A и (B или C)': false
Результат '(A или B и не C) и C': false
```