Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет Компьютерного Проектирования
Кафедра инженерной психологии и эргономики

Отчёт по предмету «Программирование мобильных информационных систем»
по лабораторной работе №1
на тему:

ОСНОВЫ ЯЗЫКА *KOTLIN*

Проверил _____ Усенко Ф.В.
(подпись)

Выполнил _____ Ахрамович С.П.,
(подпись) гр.214301

Минск 2024

Цель работы: Изучить основные конструкции языка *Kotlin*, научиться работать с типами данных, операциями ввода-вывода, а также условными операторами. Закрепить полученные знания через выполнение задач, требующих практического применения теории.

ВАРИАНТ 2. Разложение числа на простые множители.

КОД ПРОГРАММЫ

```
import java.util.Scanner

fun primeFactors(n: Int): List<Int> {
    var num = n
    val factors = mutableListOf<Int>()
    var i = 2

    while (i * i <= num) {
        if (num % i == 0) {
            factors.add(i)
            num /= i
        } else {
            i++
        }
    }

    if (num > 1) factors.add(num)
    return factors
}

fun main() {
    val scanner = Scanner(System.`in`)
    do {
        print("Введите число: ")
        val number = scanner.nextInt()

        if (number > 1) {
            val factors = primeFactors(number)
            if (factors.size == 1) {
                println("Число $number является простым.")
            } else {
                println("Простые множители числа $number: $factors")
            }
        } else {
            println("Введите число больше 1.")
        }

        print("Хотите ввести другое число? (y/n): ")
        val answer = scanner.next().lowercase()

    } while (answer == "y")

    println("Программа завершена.")
}
```

Результат работы программы представлен на рисунке 1.

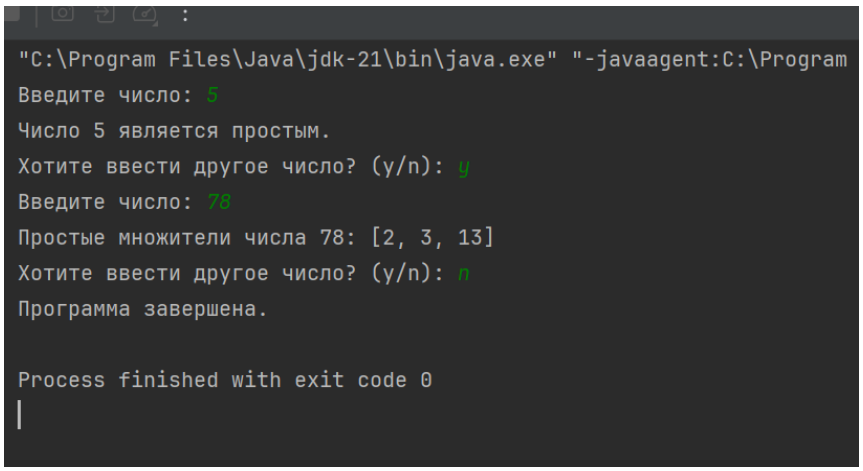

Рисунок 1 – Результат работы программы

КОНТРОЛЬНЫЕ ВОПРОСЫ

1 Как организован ввод данных с консоли в *Kotlin*?

Ввод данных от пользователя в *Kotlin* осуществляется с помощью функции *readLine*(), которая возвращает введенную строку. Для преобразования строки в число используются методы *toInt*(), *toDouble*(), и т.д.

2 Какие типы данных существуют в *Kotlin* для целых чисел?

Byte (8 бит), *Short* (16 бит), *Int* (32 бита), *Long* (64 бита) – типы данных для хранения целых чисел.

3 Какие операторы используются для сравнения чисел в *Kotlin*?

==, != – проверка на равенство и неравенство;
>, < – больше и меньше;
>=, <= – больше или равно, меньше или равно.

4 Чем отличаются переменные, объявленные с использованием *val* и *var*?

val – это неизменяемая переменная (аналог константы). Её значение нельзя изменить после присвоения.

var – это изменяемая переменная, её значение можно изменять в процессе работы программы.

5 Как объявить строковую переменную в *Kotlin*? Можно ли её изменить после объявления?

Строковая переменная объявляется с использованием типа String. Пример:

```
val greeting: String = "Hello"
```

Строки в Kotlin неизменяемы. Это значит, что после создания строку изменить нельзя, но можно создать новую строку на основе существующей.

6 В чем разница между конструкциями *if...else* и *when*?

if...else используется для проверки условий и выполнения соответствующих действий.

when используется для выбора одного из множества действий в зависимости от значения переменной (аналог *switch*).

7 Как создать функцию в *Kotlin*, которая возвращает значение? Приведите пример.

Функция объявляется с использованием ключевого слова *fun* и указывает возвращаемый тип. Пример:

```
fun sum(a: Int, b: Int): Int {
    return a + b
}
```

8 Что такое параметры функции по умолчанию, и как они используются в *Kotlin*?

Параметры по умолчанию позволяют вызывать функцию без передачи всех аргументов. Пример:

```
fun greet(name: String = "Guest") {
    println("Hello, $name!")
}
```

9 Можно ли передать функцию в качестве аргумента другой функции? Приведите пример.

Да, функции в *Kotlin* могут быть переданы в качестве аргументов. Пример:

```
fun operate(a: Int, b: Int, operation: (Int, Int) -> Int): Int {
    return operation(a, b)
}
```

```
val result = operate(3, 4, ::sum)
```

10 Какую конструкцию следует использовать для выбора действия на основе множества значений переменной?

Для этого можно использовать конструкцию *when*, которая позволяет выбирать действие в зависимости от значения переменной.

ВЫВОД

В ходе выполнения лабораторной работы были изучены основные конструкции языка *Kotlin*, удалось научиться работать с типами данных, операциями ввода-вывода, а также условными операторами. Полученные знания были закреплены через выполнение задач, требующих практическое применение теории.