Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерного проектирования

Кафедра инженерной психологии и эргономики

Дисциплина: Программирование мобильных информационных систем

ОТЧЁТ по лабораторной работе №1 на тему:

**«Основы языка Kotlin»**

Выполнил: Проверил:

ст. гр. 214301 Усенко Ф.В.

Приходько Н.Н.

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 10**. **Проверка на анаграммы:** Напишите программу, которая проверяет, являются ли две введенные строки анаграммами друг друга (например, "listen" и "silent"). Программа должна игнорировать пробелы, регистр и знаки препинания.

# КОД ПРОГРАММЫ

fun areAnagrams(str1: String, str2: String): Boolean {  
 // Функция для очистки строки от пробелов, знаков препинания и перевода в нижний регистр  
 fun cleanString(str: String): String {  
 var result = ""  
 for (i in 0 *until* str.length) {  
 val char = str[i]  
 // Проверка, является ли символ буквой или цифрой  
 if ((char >= 'a' && char <= 'z') || (char >= 'A' && char <= 'Z') || (char >= '0' && char <= '9')) {  
 // Преобразование символа в нижний регистр, если это буква  
 if (char >= 'A' && char <= 'Z') {  
 result += (char + 32) // преобразование в нижний регистр  
 } else {  
 result += char  
 }  
 }  
 }  
 return result  
 }  
  
 // Функция для сортировки строки (простая сортировка пузырьком)  
 fun bubbleSort(str: String): String {  
 val charArray = str.*toCharArray*()  
 for (i in 0 *until* charArray.size) {  
 for (j in 0 *until* charArray.size - 1 - i) {  
 if (charArray[j] > charArray[j + 1]) {  
 val temp = charArray[j]  
 charArray[j] = charArray[j + 1]  
 charArray[j + 1] = temp  
 }  
 }  
 }  
 return *String*(charArray)  
 }  
  
 // Очистка обеих строк  
 val cleanedStr1 = cleanString(str1)  
 val cleanedStr2 = cleanString(str2)  
  
 // Сортировка очищенных строк  
 val sortedStr1 = bubbleSort(cleanedStr1)  
 val sortedStr2 = bubbleSort(cleanedStr2)  
  
 // Проверка, равны ли отсортированные строки  
 return sortedStr1 == sortedStr2  
}  
  
fun main() {  
 // Ввод строк пользователем  
 *println*("Введите первую строку:")  
 val input1 = *readLine*() ?: ""  
  
 *println*("Введите вторую строку:")  
 val input2 = *readLine*() ?: ""  
  
 // Проверка, являются ли строки анаграммами  
 if (*areAnagrams*(input1, input2)) {  
 *println*("Строки являются анаграммами.")  
 } else {  
 *println*("Строки не являются анаграммами.")  
 }  
}

# ОТВЕТЫ НА КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Какие типы циклов существуют в Kotlin, и как они используются?

В языке Kotlin используются циклы for, while, do…while.

1. Как работает цикл for для итерации по элементам массива? Приведите пример.

Цикл for используется для выполнения набора инструкций заданное количество раз или для итерации по элементам коллекции (например, массиву).

Пример:

val numbers = arrayOf(1, 2, 3, 4, 5)

for (number in numbers) {

println(number)

}

В этом примере цикл for используется для перебора элементов массива numbers.

1. Чем отличается цикл while от цикла for?

В отличие от цикла for, выполняющего блок кода определенное количество раз, цикл while выполняет блок кода до тех пор, пока условие истинно.

1. Какую роль выполняет цикл do...while, и в чем его отличие от while?

Цикл do...while отличается от цикла while тем, что сначала выполняется действие, и лишь потом проверяется условие.

1. Что такое диапазон в Kotlin, и как его создать?

В Kotlin диапазон (range) представляет собой последовательность значений, которая идет от начального значения до конечного. Диапазоны поддерживают как целые числа, так и символы, а также другие типы.

Для создания диапазона в Kotlin можно использовать оператор range.

Пример:

val range = 1..5

for (i in range) {

println(i)

}

# ВЫВОД

В ходы выполнения лабораторной работы были изучены основные конструкции языка Kotlin. Закреплены полученные знания через выполнение задач, требующих практического применения теории.