Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерного проектирования

Кафедра проектирования информационно-компьютерных систем

Отчёт

К лабораторной работе № 1

на тему:

**ОСНОВЫ ЯЗЫКА KOTLIN**

Проверил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Усенко Ф. В.

(подпись)

Выполнил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ П.А. Барбук

(подпись) 214302

Минск 2024

**Вариант 1. Калькулятор с поддержкой переменных**

Условие: Напишите программу, которая позволяет пользователю вводить выражения с переменными (например, "x + y \* 2") и сохранять переменные для последующего использования. Программа должна поддерживать операции присваивания и выводить результат вычисления.

**Код программы:**

import java.util.\*

fun parseExpression(expression: String): List<String> {

val tokens = mutableListOf<String>()

var num = StringBuilder()

for (char in expression) {

when {

char.isLetterOrDigit() -> num.append(char)

char in "+-\*/()" -> {

if (num.isNotEmpty()) {

tokens.add(num.toString())

num = StringBuilder()

}

tokens.add(char.toString())

}

}

}

if (num.isNotEmpty()) {

tokens.add(num.toString())

}

return tokens

}

fun precedence(op: String): Int {

return when (op) {

"+", "-" -> 1

"\*", "/" -> 2

else -> 0

}

}

fun infixToPostfix(expression: List<String>): List<String> {

val stack = Stack<String>()

val output = mutableListOf<String>()

for (token in expression) {

when {

token.all { it.isLetterOrDigit() } -> output.add(token)

token == "(" -> stack.push(token)

token == ")" -> {

while (stack.isNotEmpty() && stack.peek() != "(") {

output.add(stack.pop())

}

stack.pop()

}

else -> {

while (stack.isNotEmpty() && precedence(stack.peek()) >= precedence(token)) {

output.add(stack.pop())

}

stack.push(token)

}

}

}

while (stack.isNotEmpty()) {

output.add(stack.pop())

}

return output

}

fun evaluatePostfix(postfix: List<String>, variables: Map<String, Double>): Double {

val stack = Stack<Double>()

for (token in postfix) {

when {

token.toDoubleOrNull() != null -> stack.push(token.toDouble())

token in variables -> stack.push(variables[token])

else -> {

val b = stack.pop()

val a = stack.pop()

when (token) {

"+" -> stack.push(a + b)

"-" -> stack.push(a - b)

"\*" -> stack.push(a \* b)

"/" -> stack.push(a / b)

}

}

}

}

return stack.pop()

}

fun main() {

print("Введите выражение: ")

val expression = readlnOrNull() ?: return

val tokens = parseExpression(expression)

val postfix = infixToPostfix(tokens)

val variables = mutableMapOf<String, Double>()

for (token in postfix) {

if (token.first().isLetter() && token !in variables) {

print("Введите значение для $token: ")

val value = readlnOrNull()?.toDoubleOrNull() ?: return

variables[token] = value

}

}

val result = evaluatePostfix(postfix, variables)

println("Результат выражения: $result")

}

**Контрольные вопросы:**

1. Что такое функция *main* в *Kotlin*, и какова её роль в программе?

– Функция, являющаяся точкой входа в программу. В данной функции

выполняется основной код программы.

2. Чем отличаются переменные, объявленные с использованием *val* и

*var*?

– Обе конструкции используются для объявления переменных. Разница лишь в том, что *val* не дает переопределять значение переменной, в отличие от *var*.

3. Какие типы данных существуют в *Kotlin* для целых чисел? Приведите примеры.

– Для хранения целых данных есть *Int, Short, Byte, Long*. Для хранения дробных чисел есть *Float* и *Double*. Хранение символьных данных – *Char*. Логические значения – *Boolean*.

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

– Строковая переменная объявляется следующим образом: val greeting = "Hello". Для хранения строковых значений используется тип *String*. Данные такого типа изменять нельзя.

5. Что такое интерполяция строк, и как она работает в *Kotlin*?

– Интерполяция – это альтернатива конкатенации в *Java*. Вместо присоединения переменных к строке (конкатенация), используется встраивание переменных в строки, что делает код более читаемым.

Пример: val greeting = "Привет, меня зовут $name и мне $age лет."