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. 214302 Усенко Ф.В.

Петрусевич Д.А.

Минск 2024

Вариант 20.Напишите программу, которая принимает математическое выражение, оптимизирует его (например, объединяет константы, упрощает выражения) и выводит упрощенную версию. Программа должна корректно работать с вложенными выражениями и скобками.

Листинг кода:

import java.util.\*  
fun main() {  
 val scanner = Scanner(System.*`in`*)  
 while (true) {  
 *println*("Введите математическое выражение (или 'exit' для выхода):")  
 val input = scanner.nextLine()  
 if (input.*lowercase*() == "exit") break  
  
 try {  
 val optimizedExpression = *optimizeExpression*(input)  
 *println*("Упрощенное выражение: $optimizedExpression")  
 } catch (e: Exception) {  
 *println*("Ошибка: ${e.message}")  
 }  
 }  
}  
fun optimizeExpression(expression: String): String {  
 // Удаляем пробелы и обрабатываем скобки  
 val cleanedExpression = expression.*replace*("\\s+".*toRegex*(), "")  
 return *simplify*(cleanedExpression)  
}  
fun simplify(expression: String): String {  
 // Обрабатываем базовые арифметические операции  
 return try {  
 val result = *evaluateExpression*(expression)  
 result.toString()  
 } catch (e: Exception) {  
 throw IllegalArgumentException("Не удалось упростить выражение: $expression")  
 }  
}  
fun evaluateExpression(expr: String): Double {  
 val tokens = *tokenize*(expr)  
 val postfix = *toPostfix*(tokens)  
 return *evaluatePostfix*(postfix)  
}  
  
fun tokenize(expr: String): List<String> {  
 val regex = "([+\\-\*/()])|([0-9]+(?:\\.[0-9]+)?)".*toRegex*()  
 return regex.findAll(expr).*map* **{ it**.value **}**.*toList*()  
}  
fun toPostfix(tokens: List<String>): List<String> {  
 val output = *mutableListOf*<String>()  
 val stack = Stack<String>()  
  
 for (token in tokens) {  
 when {  
 token.*matches*("\\d+".*toRegex*()) -> output.add(token) // Число  
 token == "(" -> stack.push(token) // Открывающая скобка  
 token == ")" -> {  
 while (stack.*isNotEmpty*() && stack.peek() != "(") {  
 output.add(stack.pop())  
 }  
 stack.pop() // Удаляем открывающую скобку  
 }  
 else -> {  
 while (stack.*isNotEmpty*() && *precedence*(token) <= *precedence*(stack.peek())) {  
 output.add(stack.pop())  
 }  
 stack.push(token)  
 }  
 }  
 }  
  
 while (stack.*isNotEmpty*()) {  
 output.add(stack.pop())  
 }  
  
 return output  
}  
fun precedence(op: String): Int {  
 return when (op) {  
 "+", "-" -> 1  
 "\*", "/" -> 2  
 else -> 0  
 }  
}  
fun evaluatePostfix(tokens: List<String>): Double {  
 val stack = Stack<Double>()  
  
 for (token in tokens) {  
 when {  
 token.*matches*("\\d+(\\.\\d+)?".*toRegex*()) -> stack.push(token.*toDouble*())  
 else -> {  
 val right = stack.pop()  
 val left = stack.pop()  
 val result = when (token) {  
 "+" -> left + right  
 "-" -> left - right  
 "\*" -> left \* right  
 "/" -> left / right  
 else -> throw IllegalArgumentException("Неизвестная операция: $token")  
 }  
 stack.push(result)  
 }  
 }  
 }  
 return stack.pop()  
}

Контрольные вопросы:

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

Функция *main* является точкой входа в программу. В этой функции выполняется основной код программы.

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

Переменные объявленные с помощью *val* являются неизменяемыми переменными. То есть мы можем присвоить значение такой переменной только один раз, но изменить его после первого присвоения мы уже не сможем. В свою очередь переменные, объявленные с помощью *var* имеют возможность многократно менять значения.

1. Как объявить строковую переменную в Kotlin? Можно ли её изменить

после объявления?

В *Kotlin* строковую переменную можно объявить с помощью ключевых слов *val* или *var*.

1. **val** используется для объявления неизменяемой (финальной переменной. После объявления значение не может быть изменено.
2. **var** используется для объявления изменяемой переменной, значение которой можно изменять после объявления.