Министерство образования

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерного проектирования

Кафедра инженерной психологии и эргономики

Отчет по лабораторной работе №2

«Использование языка программирования Swift: функции, замыкания, перечисления»

Выполнил: Моров Н.В.

Студент группы 310902

Проверил: Усенко Ф.В.

Минск 2024

Цель работы – Выполнить разработку приложения с использованием языка программирования Swift: функции, замыкания, перечисления.

**Задание 1.** Дано натуральное число P. Найдите все делители числа P.

**Код на языке Swift:**

**Package.swift**

**// swift-tools-version:5.7**

**import PackageDescription**

**let package = Package(**

**name: "DivisorFinder",**

**targets: [**

**.executableTarget(**

**name: "MainModule",**

**dependencies: ["ValidationModule", "ProcessingModule"]**

**),**

**.target(**

**name: "ValidationModule",**

**dependencies: []**

**),**

**.target(**

**name: "ProcessingModule",**

**dependencies: []**

**),**

**]**

**)**

**MainModule\main.swift**

**import ProcessingModule**

**import ValidationModule**

**public struct Application {**

**/// Замыкание для запуска программы**

**public static let run: () -> Void = {**

**print("Введите натуральное число P:")**

**if let input = readLine(),**

**let number = InputValidation.validateNaturalNumber(input) {**

**let divisors = NumberOperations.findDivisors(number)**

**print("Делители числа \(number): \(divisors)")**

**} else {**

**print("Ошибка: Введите корректное натуральное число!")**

**}**

**}**

**}**

**Application.run()**

**ProcesingModule\NumberOperation.swift**

**public struct PrimeFinder {**

**public static func findPrimes(upTo p: Int) -> [Int] {**

**guard p >= 2 else { return [] }**

**var isPrime = [Bool](repeating: true, count: p + 1)**

**isPrime[0] = false**

**isPrime[1] = false**

**for i in 2...Int(Double(p).squareRoot()) {**

**if isPrime[i] {**

**for multiple in stride(from: i \* i, through: p, by: i) {**

**isPrime[multiple] = false**

**}**

**}**

**}**

**return (2...p).compactMap { isPrime[$0] ? $0 : nil }**

**}**

**}**

**ValidationModule\InputValidation.swift**

**public struct InputValidation {**

**public static let validateNaturalNumber: (String) -> Int? = { input in**

**guard let number = Int(input), number > 0 else {**

**return nil**

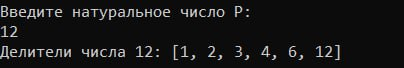
**}**

**return number**

**}**

**}**

**Скриншот работающей программы:**

****

**Рисунок 1 - Результат выполнения программы**

**Задание 2. Используя функциональные типы, создайте программу согласно варианту.**

а) для последовательности целых чисел находит среднее арифметическое;

б) для строки находит количество букв, содержащихся в ней.

**Код на языке Swift:**

**Package.swift**

**// swift-tools-version:5.7**

**import PackageDescription**

**let package = Package(**

**name: "FunctionalProgram",**

**targets: [**

**.executableTarget(**

**name: "MainModule",**

**dependencies: ["ValidationModule", "ProcessingModule"]**

**),**

**.target(**

**name: "ValidationModule",**

**dependencies: []**

**),**

**.target(**

**name: "ProcessingModule",**

**dependencies: []**

**),**

**]**

**)**

**ProcessingModule\Calculations.swift**

**import Foundation**

**public struct Calculations {**

**/// Вычисляет среднее арифметическое чисел**

**public static let calculateAverage: ([Int]) -> Double = { numbers in**

**guard !numbers.isEmpty else { return 0 }**

**let sum = numbers.reduce(0, +)**

**return Double(sum) / Double(numbers.count)**

**}**

**/// Подсчитывает количество букв в строке**

**public static let countLetters: (String) -> Int = { input in**

**input.filter { $0.isLetter }.count**

**}**

**}**

**MainModule\main.swift**

**import ValidationModule**

**import ProcessingModule**

**public struct Application {**

**public static let run: () -> Void = {**

**print("Выберите операцию:")**

**print("1. Найти среднее арифметическое последовательности чисел.")**

**print("2. Найти количество букв в строке.")**

**if let choice = readLine(), let option = Int(choice) {**

**switch option {**

**case 1:**

**handleAverageCalculation()**

**case 2:**

**handleLetterCounting()**

**default:**

**print("Ошибка: Неверный выбор. Завершение программы.")**

**}**

**} else {**

**print("Ошибка: Введите корректный номер операции.")**

**}**

**}**

**private static func handleAverageCalculation() {**

**print("Введите последовательность целых чисел, разделённых пробелами:")**

**if let input = readLine(),**

**let numbers = InputValidation.validateIntegerSequence(input) {**

**let average = Calculations.calculateAverage(numbers)**

**print("Среднее арифметическое: \(average)")**

**} else {**

**print("Ошибка: Введите корректную последовательность чисел.")**

**}**

**}**

**private static func handleLetterCounting() {**

**print("Введите строку:")**

**if let input = readLine(),**

**let validString = InputValidation.validateNonEmptyString(input) {**

**let letterCount = Calculations.countLetters(validString)**

**print("Количество букв в строке: \(letterCount)")**

**} else {**

**print("Ошибка: Введите непустую строку.")**

**}**

**}**

**}**

**Application.run()**

**ValidationModule\InputValidation.swift**

**import Foundation**

**public struct InputValidation {**

**/// Проверяет, что строка содержит только числа, разделённые пробелами**

**public static let validateIntegerSequence: (String) -> [Int]? = { input in**

**let numbers = input.split(separator: " ").compactMap { Int($0) }**

**return numbers.isEmpty ? nil : numbers**

**}**

**/// Проверяет, что строка непустая**

**public static let validateNonEmptyString: (String) -> String? = { input in**

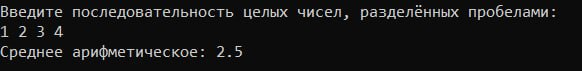
**let trimmedInput = input.components(separatedBy: .whitespacesAndNewlines).joined()**

**return trimmedInput.isEmpty ? nil : input**

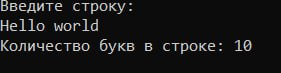
**}**

**}**

**Скриншот работающей программы:**

****

**Рисунок 2 - Результат выполнения программы**

****

**Рисунок 3 - Результат выполнения программы**

**Вывод: В этой лабораторной работе мы разработали два приложения с помощью языке программирования Swift. При разработке были использованы функции, замыкания и перечисления. Программа была разбита на несколько модулей с учетом логики работы.**