Министерство образования

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерного проектирования

Кафедра инженерной психологии и эргономики

Отчет по лабораторной работе №3

«Использование языка программирования Swift: структуры и классы, методы»

Выполнил: Моров Н.В.

Студент группы 310902

Проверил: Усенко Ф.В.

Минск 2024

Цель работы – Выполнить разработку приложения с использованием языка программирования Swift: структуры и классы, методы.

* 1. Задание. Николай – оригинальный человек. Он решил создать класс Nikola, принимающий при инициализации 2 параметра: имя и возраст. Но на этом он не успокоился. Не важно, какое имя передаст пользователь при создании экземпляра, оно всегда будет содержать «Николая». В частности – если пользователя на самом деле зовут Николаем, то с именем ничего не произойдет, а если его зовут, например, Максим, то оно преобразуется в “Я не Максим, а Николай”.

**Код на языке Swift:**

**Package.swift**

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

**import PackageDescription**

**let package = Package(**

**name: "Nikita3",**

**targets: [**

**.executableTarget(**

**name: "MainModule",**

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

**),**

**.target(**

**name: "ValidationModule",**

**dependencies: []**

**),**

**.target(**

**name: "NikolaModule",**

**dependencies: ["ValidationModule"]**

**),**

**]**

**)**

**MainModule\main.swift**

**import Foundation**

**import NikolaModule**

**import ValidationModule**

**do {**

**let nikolay1 = try Nikola(name: "Николай", age: 30)**

**let nikolay2 = try Nikola(name: "Максим", age: 25)**

**let nikolay3 = try Nikola(name: "Александр", age: 89)**

**let nikolaPeople = [nikolay1, nikolay2, nikolay3]**

**for person in nikolaPeople {**

**print(person.description())**

**}**

**} catch let error as ValidationError {**

**print("Ошибка валидации: \(error.description)")**

**} catch {**

**print("Неизвестная ошибка: \(error)")**

**}**

**NikolaModule\Nikola.swift**

**import Foundation**

**import ValidationModule**

**public class Nikola {**

**public let name: String**

**public let age: Int**

**public init(name: String, age: Int) throws {**

**try Validation.validateAge(age)**

**try Validation.validateName(name)**

**self.age = age**

**if name.contains("Николай") {**

**self.name = name**

**} else {**

**self.name = "Я не \(name), а Николай"**

**}**

**}**

**public func description() -> String {**

**return "Имя: \(name), Возраст: \(age)"**

**}**

**}**

**ValidationModule\InputValidation.swift**

**import Foundation**

**public enum ValidationError: Error, CustomStringConvertible {**

**case invalidAge**

**case invalidName**

**public var description: String {**

**switch self {**

**case .invalidAge:**

**return "Возраст должен быть положительным числом."**

**case .invalidName:**

**return "Имя не может быть пустым."**

**}**

**}**

**}**

**public class Validation {**

**public static func validateAge(\_ age: Int) throws {**

**if age <= 0 {**

**throw ValidationError.invalidAge**

**}**

**}**

**public static func validateName(\_ name: String) throws {**

**if name.trimmingCharacters(in: .whitespacesAndNewlines).isEmpty {**

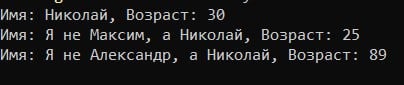
**throw ValidationError.invalidName**

**}**

**}**

**}**

**Скриншот работающей программы:**

****

**Рисунок 1 - Результат выполнения программы**

**Вывод: В этой лабораторной работе мы разработали приложение на языке программирования Swift, используя структуры, классы и методы. Функционал приложения был разбит на несколько пакетов, учитывая логику работы, и была сделана валидация вводимых значений.**