Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет

информатики и радиоэлектроники»

Учебная дисциплина «Современные языки программирования»

Отчёт

по лабораторной работе №2

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

Вариант 15

Выполнил студент группы 310101

Криницкий М.А.

Проверил:

Усенко Ф.В.

Минск 2024

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

Язык Swift предлагает достаточно гибкий единый синтаксис функций – от простых C-подобных функций без параметров до сложных методов в стиле Objective-C с локальными и внешними параметрами.

Замыкания – это самодостаточные блоки с определенным функционалом, которые могут быть переданы и использованы в вашем коде. Замыкания в Swift похожи на блоки в C и Objective-C, и лямбды в других языках программирования.

*Перечисления* определяют общий тип для группы ассоциативных значений и позволяют работать с этими значениями в типобезопасном режиме в вашем коде.

Задание 1

Дано натуральное число P. Проверьте, кратно ли P одиннадцати, используя признак делимости на 11 (Листинг 1).

Листинг 1

func DevEleven(value:Int) {

var countOne=0;

var countTwo=0;

var i=1;

var tempvalue=value;

while tempvalue != 0 {

if i % 2 == 0 {

countOne += tempvalue % 10

} else {

countTwo += tempvalue % 10

}

i += 1

tempvalue /= 10

}

if (countOne-countTwo)%11==0

{

print("Делится на 11")

}

else{

print(" НЕ Делится на 11")

}

}

DevEleven(value:1111)

Задание 2.

Используя функциональные типы, создать программу согласно варианту.

а) целые числа возводит в степень n;

б) из десятичных чисел извлекает корень степени n (Листинг 2).

Листинг 2

import Foundation

func ExtentInt(value: Int, extent: Int) -> Int {

return Int(pow(Double(value), Double(extent)))

}

func ExtentDouble(value: Double, extent: Int) -> Double {

return pow(value, 1.0 / Double(extent))

}

func chooseFunction(mathFunctionInt: (Int, Int) -> Int, mathFunctionDouble: (Double, Int) -> Double, a: Any, b: Any) -> Any? {

if let intValue = a as? Int, let intExtent = b as? Int {

return mathFunctionInt(intValue, intExtent)

} else if let doubleValue = a as? Double, let intExtent = b as? Int {

return mathFunctionDouble(doubleValue, intExtent)

}

return nil

}

let result = chooseFunction(mathFunctionInt: ExtentInt, mathFunctionDouble: ExtentDouble, a: 0.64, b: 2)

if let intResult = result as? Int {

print("Результат для Int: \(intResult)")

} else if let doubleResult = result as? Double {

print("Результат для Double: \(doubleResult)")

}

Скриншоты работы программы представлены на рисунках 1,2.

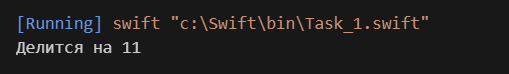


Рисунок 1 – Скриншот работы программы задания 1

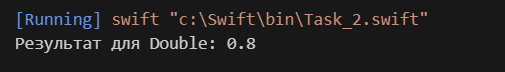


Рисунок 2 – Скриншот работы программы задания 2

**Вывод**

В ходе лабораторной работы были продемонстрированы основные принципы программирования на Swift, включая использование функций для выполнения задач, замыканий для обработки данных и перечислений для управления состояниями.