Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Кафедра инженерной психологии и эргономики

Современные языки программирования

Отчет по лабораторной работе №2

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

Выполнил: Чиж А. В.

Студент группы 310901

Преподаватель: Усенко Ф. В.

Минск 2024

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

1. **Вариант 2, Задание 1:** Дано натуральное число m. Укажите все тройки натуральных чисел x, y и z, удовлетворяющие следующему условию: m = x3 + y3 + z3.

**Код программы:**

let first: task1 = task1(m: validate.getForTask1())

for element in first.result() {

print("\(element.x), \(element.y), \(element.z)")

}

class task1 {

var m: Int

var list: [(x: Int, y: Int, z: Int)] = []

init(m: Int) {

self.m = m

}

public func result() -> [(x: Int, y: Int, z: Int)] {

var tmp: (x: Int, y: Int, z: Int)

for i in stride(from: 1, to: self.m - 1, by: 1) {

tmp.x = i

for j in stride(from: 1, to: self.m - i, by: 1) {

tmp.y = j

tmp.z = self.m - i - j

list.append(tmp)

}

}

return list

}

}

Результат работы программы представлен на рисунке 1

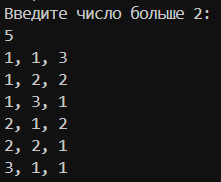


Рисунок 1 – Результат работы программы

**Вариант 2, Задание 2:** а) для сложения вещественных чисел;

б) для сложения комплексных чисел

**Код программы:**

let second: task2 = task2(input: validate.getForTask2())

let answer = second.result()

print("Ответ: \(answer.real)\(answer.imaginary == 0 ? "" : (answer.imaginary > 0 ? "+\(answer.imaginary)i" : "\(answer.imaginary)i"))")

class task2 {

struct Number {

var real: Int = 0

var imaginary: Int = 0

}

var a: Number = Number()

var b: Number = Number()

init (input: (real1: Int, imaginary1: Int, real2: Int, imaginary2: Int)) {

self.a.real = input.real1

self.a.imaginary = input.imaginary1

self.b.real = input.real2

self.b.imaginary = input.imaginary2

}

public func result() -> Number {

return sum(rule: chooseSum(real: (self.a.imaginary == 0 && self.b.imaginary == 0)), a: self.a, b: self.b)

}

private func sum(rule: (Number, Number) -> Number, a: Number, b: Number) -> Number {

return rule(a, b);

}

private func chooseSum(real: Bool) -> (Number, Number) -> Number {

return real ? sumReal : sumComplex

}

private func sumReal(a: Number, b: Number) -> Number {

var number: Number = Number()

number.real = a.real + b.real

number.imaginary = 0

return number

}

private func sumComplex(a: Number, b: Number) -> Number {

var number: Number = sumReal(a: a, b: b)

number.imaginary = a.imaginary + b.imaginary

return number

}

}

Результат работы программы представлен на рисунке 2

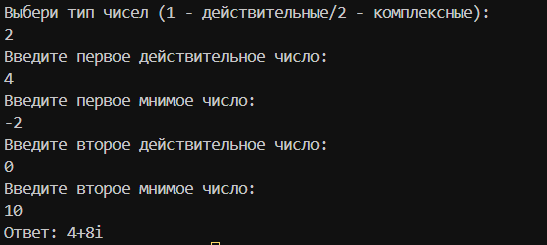


Рисунок 2 – Результат работы программы

**Ответы на вопросы к лабораторной работе:**

* 1. Как объявить функцию с несколькими параметрами?

Объявить функцию с несколькими параметрами, указав их типы в скобках после имени функции. Каждый параметр должен иметь уникальное имя и тип.

* 1. Для чего служат ярлыки аргументов и имен параметров функции?

Ярлыки аргументов — это имена, которые используются при вызове функции, чтобы указать, какой аргумент вы передаете. Они помогают сделать код более понятным, особенно когда функция принимает несколько параметров.

* 1. Для чего служат функциональные типы? Как создать свой функциональный тип?

Функциональные типы в Swift представляют собой типы, которые могут хранить ссылки на функции и использовать их как значения. Это позволяет вам передавать функции в качестве аргументов, возвращать их из других функций и хранить их в переменных или коллекциях.

* 1. Что такое замыкание? Для чего оно служит?

Замыкание (closure) в Swift — это самодостаточная единица кода, которая может захватывать и хранить свои окружения. Замыкания могут использоваться как функции, но они не имеют имени и могут быть присвоены переменным или переданы в качестве аргументов другим функциям.

**Вывод**: Изучена работа языка *Swift* и выполнена разработка приложения с использованием функций, замыканий и перечислений.