Министерство образования

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерного проектирования

Кафедра инженерной психологии и эргономики

Отчет по лабораторной работе №4

«Использование языка программирования Swift: наследование, протоколы»

Выполнил: Бельский М.Д.

Студент группы 310902

Проверил: Усенко Ф. В.

Минск 2024

Цель работы – Выполнить разработку приложения с использованием языка программирования Swift: наследование, протоколы.

1 Класс рациональных дробей. класс, представляющий рациональную дробь (num – числитель, den – знаменатель). Класс содержит методы умножения и деления (дроби на дробь и дроби на целое число). Создать протокол, который служит для создания случайной дроби из заданного диапазона целых. Добавить две различные реализации протокола.

Листинг кода:

protocol RandomFraction {

func getRandomFraction(minNum: Int, maxNum: Int, minDen: Int, maxDen: Int) -> Fraction

}

class Fraction {

var num: Int

var den: Int

init(num: Int, den: Int) {

self.num = num

self.den = den

}

func multiply(\_ other: Fraction) -> Fraction {

return Fraction(num: self.num \* other.num, den: self.den \* other.den)

}

func divide(\_ other: Fraction) -> Fraction {

return Fraction(num: self.num \* other.den, den: self.den \* other.num)

}

func multiply(\_ other: Int) -> Fraction {

return Fraction(num: self.num \* other, den: self.den)

}

func divide(\_ other: Int) -> Fraction {

return Fraction(num: self.num, den: self.den \* other)

}

}

class FirstGenerateRandomFraction: RandomFraction {

private let random = Int.random(in: 0...Int.max)

func getRandomFraction(minNum: Int, maxNum: Int, minDen: Int, maxDen: Int) -> Fraction{

let num = Int.random(in: minNum..<maxNum+1)

let den = Int.random(in: minDen..<maxDen+1)

return Fraction(num: num, den: den)

}

}

class SecondGenerateRandomFraction: RandomFraction {

func getRandomFraction(minNum: Int, maxNum: Int, minDen: Int, maxDen: Int) -> Fraction {

let num = Int.random(in: minNum...maxNum)

let den = Int.random(in: minDen...maxDen)

let nod = gcd(a: num, b: den)

let reducedNum = num / nod

let reducedDen = den / nod

return Fraction(num: reducedNum, den: reducedDen)

}

private func gcd(a: Int, b: Int) -> Int {

var a = a

var b = b

while b != 0 {

let temp = b

b = a % b

a = temp

}

return a

}

}

// Пример использования

let generator1 = FirstGenerateRandomFraction()

let f1 = generator1.getRandomFraction(minNum: 1, maxNum: 10, minDen: 2, maxDen: 12)

print("\(f1.num)/\(f1.den)")

let generator2 = SecondGenerateRandomFraction()

let f2 = generator2.getRandomFraction(minNum: 1, maxNum: 10, minDen: 2, maxDen: 12)

let resultMult = f1.multiply(f2)

let resultDiv = f1.divide(f2)

print("\(f2.num)/\(f2.den)")

print("\(f1.num)/\(f1.den) \* \(f2.num)/\(f2.den) = \(resultMult.num)/\(resultMult.den)")

print("\(f1.num)/\(f1.den) / \(f2.num)/\(f2.den) = \(resultDiv.num)/\(resultDiv.den)")

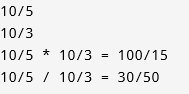


Рисунок 1 – Результат выполнения программы

Вывод: в результате выполнения лабораторной работы я выполнил разработку приложения с использованием языка программирования Swift: наследование, протоколы.