Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Дисциплина: современные языки программирования

**Лабораторная работа №4**

**«Использование языка программирования Swift: наследование, протоколы»**

Вариант 15

Выполнил: Шабанович А.Б.

студент гр. 310102

Проверил: Усенко Ф.В.

Минск 2024

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

Задание: Протокол Создания уникальных чисел. Реализовать метод, который генерирует уникальное число. Создать класс, который будет иметь в себе генератор. Добавить несколько различных алгоритмов создания уникальных чисел. Создать несколько уникальных последовательностей чисел.

Листинг кода:

import Foundation

// Протокол для генерации уникальных чисел

protocol UniqueNumberGenerator {

    func generateUniqueNumber() -> Int

}

// Реализация алгоритма генерации уникальных чисел на основе счетчика

class IncrementalNumberGenerator: UniqueNumberGenerator {

    private var counter: Int = 77

    func generateUniqueNumber() -> Int {

        counter += 1

        return counter

    }

}

// Реализация алгоритма генерации уникальных чисел на основе случайных чисел

class RandomNumberGenerator: UniqueNumberGenerator {

    private var generatedNumbers: Set<Int> = []

    func generateUniqueNumber() -> Int {

        var uniqueNumber: Int

        repeat {

            uniqueNumber = Int.random(in: 1...1\_000\_000)

        } while generatedNumbers.contains(uniqueNumber)

        generatedNumbers.insert(uniqueNumber)

        return uniqueNumber

    }

}

// Реализация алгоритма генерации уникальных чисел на основе временной метки

class TimestampNumberGenerator: UniqueNumberGenerator {

    private var lastTimestamp: Int = 0

    func generateUniqueNumber() -> Int {

        let timestamp = Int(Date().timeIntervalSince1970 \* 1\_000)

        if timestamp == lastTimestamp {

            lastTimestamp += 1

        } else {

            lastTimestamp = timestamp

        }

        return lastTimestamp

    }

}

// Класс для работы с генераторами

class UniqueNumberManager {

    private var generator: UniqueNumberGenerator

    init(generator: UniqueNumberGenerator) {

        self.generator = generator

    }

    func generate() -> Int {

        return generator.generateUniqueNumber()

    }

}

// Пример использования

func main() {

    print("Выберите алгоритм генерации уникальных чисел:")

    print("1. Инкрементальный генератор")

    print("2. Случайный генератор")

    print("3. Генератор на основе временной метки")

    guard let choice = readLine(), let option = Int(choice), (1...3).contains(option) else {

        print("Ошибка: введите число от 1 до 3.")

        return

    }

    let generator: UniqueNumberGenerator

    switch option {

    case 1:

        generator = IncrementalNumberGenerator()

    case 2:

        generator = RandomNumberGenerator()

    case 3:

        generator = TimestampNumberGenerator()

    default:

        print("Ошибка: неверный выбор.")

        return

    }

    let manager = UniqueNumberManager(generator: generator)

    print("Сколько уникальных чисел вы хотите сгенерировать?")

    guard let countInput = readLine(), let count = Int(countInput), count > 0 else {

        print("Ошибка: введите положительное число.")

        return

    }

    print("Сгенерированные уникальные числа:")

    for \_ in 1...count

    {

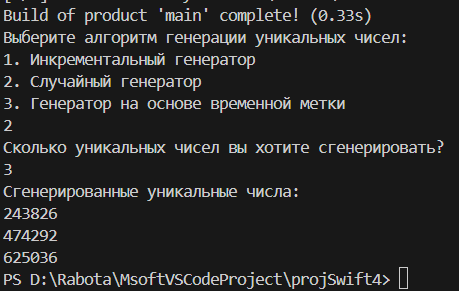
        print(manager.generate())

    }

}

main()

Результат выполнения кода:



Вывод: разработана программа на языке программирования Swift. В результате разработки программ на языке Swift была успешно реализована функциональность, позволяющая генерировать уникальные числа тремя разными способами в заданном количестве. Была проделана работа с протоколами.