Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет

информатики и радиоэлектроники»

Кафедра инженерной психологии и эргономики

­­­­­

Современные языки программирования

Отчет по лабораторной работе №4

«Использование языка программирования Swift: наследование, протоколы»

Выполнил: Пинчук Р.Д.

Студент группы 310901

Преподаватель: Усенко Ф. В.

Минск 2024

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

Задание 6: Класс Train. Пункт назначения, номер поезда, время отправления, число мест(общих, плацкарт, купе) создать список объектов, вывести: список поездов следующих до заданного пункта и количество свободных мест по категориям, список поездов, следующих до заданного пункта и отправляющихся после заданного времени. Создать протокол, который реализует метод по движению поезда.

Листинг кода:

Файл main.swift:

import Foundation

let trainService = TrainService()

// Пример создания поездов

if let train1 = Train(destination: "Москва", trainNumber: 101, departureTime: Date(), seats: (general: 50, reservedSeat: 40, coupe: 20)) {

trainService.addTrain(train: train1)

}

if let train2 = Train(destination: "Санкт-Петербург", trainNumber: 102, departureTime: Date().addingTimeInterval(3600), seats: (general: 60, reservedSeat: 50, coupe: 30)) {

trainService.addTrain(train: train2)

}

// Фильтрация поездов по пункту назначения

let filteredTrains = trainService.filterTrains(destination: "Москва")

for train in filteredTrains {

print(train.moveTrain()) // Выводим информацию о поезде

}

Результат работы программы представлен на рисунке 1.

Файл Models.swift:

import Foundation

class Train: TrainMovement {

let destination: String

let trainNumber: Int

let departureTime: Date

var seats: (general: Int, reservedSeat: Int, coupe: Int)

// Конструктор с валидацией

init?(destination: String, trainNumber: Int, departureTime: Date, seats: (general: Int, reservedSeat: Int, coupe: Int)) {

guard !destination.isEmpty, trainNumber > 0, seats.general >= 0, seats.reservedSeat >= 0, seats.coupe >= 0 else {

return nil // Валидация не пройдена

}

self.destination = destination

self.trainNumber = trainNumber

self.departureTime = departureTime

self.seats = seats

}

// Метод из протокола

func moveTrain() -> String {

return "Поезд №\(trainNumber) направляется в \(destination)"

}

}

Файл Protocols.swift:

protocol TrainMovement {

func moveTrain() -> String

}

Файл Services.swift:

import Foundation

class TrainService {

// Список поездов

var trains: [TrainMovement] = []

// Добавляем поезд (с валидацией)

func addTrain(train: TrainMovement) {

trains.append(train)

}

// Вывод списка поездов по пункту назначения и времени отправления

func filterTrains(destination: String, afterTime: Date? = nil) -> [Train] {

return trains.compactMap { $0 as? Train }.filter { train in

train.destination == destination && (afterTime == nil || train.departureTime > afterTime!)

}

}

}

Результат работы программы представлен на рисунке 1.

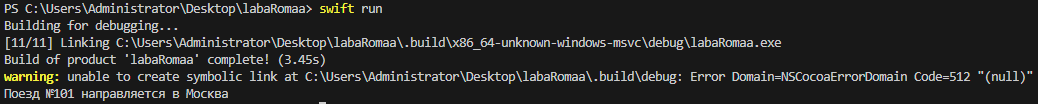


Рисунок 1 – Результат выполнения программы

Вывод: в ходе выполнения лабораторной работы номер 4 по языку программирования swift мы познакомились с методами проектирования и связей такие как наследование, реализация абстракций и их отношения. В ходе работы познакомились с необходимыми конструкциями языка для реализации абстракций.