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: Суровый И. А.

Студент группы 310901

Преподаватель: Усенко Ф. В.

Минск 2024

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

**Вариант** **12** Протокол Задача. Реализовать метод, который возвращает информацию о задаче (например, «Задача: Подготовить презентацию, выполнено: нет») и метод, который меняет статус задачи. Создать список задач и вывести их информацию и поменять статус.

**Код программы:**

import Foundation

// Список задач, работающий с экземплярами протокола TaskProtocol

class TaskManager {

var tasks: [TaskProtocol] = []

// Метод для добавления задачи

func addTask(title: String) {

do {

let newTask = try Task(title: title)

tasks.append(newTask)

print("Задача добавлена: \(newTask.getInfo())")

} catch ValidationError.invalidTitle {

print("Ошибка: заголовок задачи должен содержать не менее 3 символов.")

} catch {

print("Неизвестная ошибка.")

}

}

// Метод для вывода информации о задачах

func showTasks() {

for task in tasks {

print(task.getInfo())

}

}

// Метод для изменения статуса задачи

func toggleTaskStatus(at index: Int) {

guard index >= 0 && index < tasks.count else {

print("Ошибка: некорректный индекс задачи.")

return

}

tasks[index].toggleStatus()

}

}

// Пример использования

let taskManager = TaskManager()

// Добавление задач

taskManager.addTask(title: "Подготовить презентацию")

taskManager.addTask(title: "Сделать ДЗ")

taskManager.addTask(title: "") // Неверный ввод

// Вывод всех задач

taskManager.showTasks()

// Изменение статуса первой задачи

taskManager.toggleTaskStatus(at: 0)

// Вывод всех задач после изменения статуса

taskManager.showTasks()

protocol TaskProtocol {

var title: String { get set }

var isCompleted: Bool { get set }

func getInfo() -> String

mutating func toggleStatus()

}

// Протокол для валидации

protocol ValidatorProtocol {

func validateTitle(\_ title: String) throws

}

// Ошибки валидации

enum ValidationError: Error {

case invalidTitle

}

// Класс задачи, реализующий протокол TaskProtocol

class Task: TaskProtocol {

var title: String

var isCompleted: Bool = false

init(title: String) throws {

// Валидация заголовка

let validator = TaskValidator()

try validator.validateTitle(title)

self.title = title

}

// Метод для получения информации о задаче

func getInfo() -> String {

return "Задача: \(title), выполнено: \(isCompleted ? "да" : "нет")"

}

// Метод для изменения статуса задачи

func toggleStatus() {

isCompleted.toggle()

}

}

// Класс валидации задачи, реализующий протокол ValidatorProtocol

class TaskValidator: ValidatorProtocol {

func validateTitle(\_ title: String) throws {

if title.isEmpty || title.count < 3 {

throw ValidationError.invalidTitle

}

}

}

Результат работы программы представлен на рисунке 1

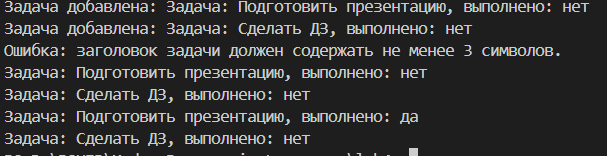


Рисунок 1 – результат работы программы

**Ответы на вопросы к лабораторной работе:**

1. Что такое наследование?

Наследование позволяет одному классу (подклассу) унаследовать свойства, методы и др характеристики др класса (суперкласса). Это позволяет создать новые классы на основе существующих, повторно используя код и добавляя новые функциональности.

1. Что такое переопределение? Что можно переопределять?

Переопределение позволяет подклассу предоставить свою собственную реализацию метода, свойства или сабскрипта, который уже определен в его суперклассе. В Swift можно переопределять методы, свойства (как вычисляемые, так и хранимые) и сабскрипты.

1. Что такое протоколы?

Протоколы определяют набор методов, свойств и других требований, которые должны быть реализованы классами, структурами или перечислениями, соответствующими этому протоколу. Протоколы позволяют описывать интерфейсы, которые должны быть реализованы.

1. Для чего служат протоколы?

Протоколы исп для определения общих интерфейсов, которые могут быть реализованы различными типами. Это позволяет создать гибкий и модульный код, где различные типы могут взаимодействовать друг с другом через общие интерфейсы.

1. Что такое расширение протоколов?

Расширение протоколов позволяет добавлять реализацию методов и свойств к протоколам. Это позволяет предоставлять стандартную реализацию для методов и свойств, которые могут быть переопределены в конкретных реализациях протоколов.

1. Какой синтаксис у протоколов?

protocol SomeProtocol

{

var someProperty: String { get set }

func someMethod()

}

**Вывод**: Изучена работа языка *Swift* и выполнена разработка приложения с использованием наследования и протоколов.