Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерного проектирования

Кафедра инженерной психологии и эргономики

Дисциплина: Современные языки программирования

Отчёт

По лабораторной работе №4

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

Вариант 13

Выполнила:

студентка группы 310101

Сердюк У. В.

Проверил:

Усенко Ф. В.

Минск 2024

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

Создать протокол для выполнения математических операций. Реализовать методы (умножение, сложение, вычитание, деление, возведение в степень).

Создаём *MathOperations*, который включает методы для выполнения базовых математических операций, таких как сложение, вычитание, умножение, деление и возведение в степень. Затем реализуем этот протокол для целых чисел в структуре *IntMathOperations*, а также для чисел с плавающей точкой в структуре *DoubleMathOperations*. В каждой из этих структур определим логику для выполнения операций, с проверкой на деление на ноль и отрицательные показатели степени. Ты продемонстрировала использование этих структур, вызывая методы для чисел разных типов. В результате, ты показала, как можно использовать протоколы для реализации обобщенной логики работы с числами.

Результат выполнения работы программы при разных вводимых значениях представлен на рисунке 1.

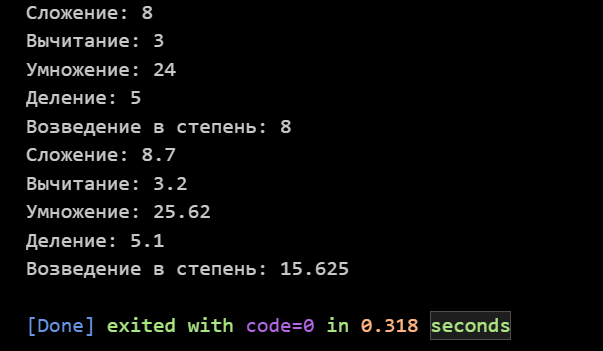


Рисунок 1 – Результат выполнения работы программы

**Вывод:** В ходе лабораторной работы разработано приложение на языке программирования *Swift*, выполнены задания на создание протокола, принимаемый классом. Осуществлена валидация вводимых значений.

**Ответы на вопросы**

1. Класс может наследовать методы, свойства и другие характеристики другого класса. Когда один класс наследует у другого класса, то наследующий класс называется подклассом, класс у которого наследуют - суперклассом. Наследование - фундаментальное поведение, которое отделяет классы от других типов Swift.

Классы в Swift могут вызывать или получать доступ к методам, свойствам, индексам, принадлежащим их суперклассам и могут предоставлять свои собственные переписанные версии этих методов, свойств, индексов для усовершенствования или изменения их поведения.

3. *Протокол* определяет образец методов, свойств или другие требования, которые соответствуют определенному конкретному заданию или какой-то функциональности. Протокол фактически не предоставляет реализацию для любого из этих требований, он только описывает как реализация должна выглядеть. Протокол может быть *принят* классом, структурой или перечислением для обеспечения фактической реализации этих требований. Любой тип, который удовлетворяет требованиям протокола, имеет указание *соответствовать* этому протоколу или другими словами *реализовать* данный протокол.

4. Определение протокола очень похоже на то, как определяются классы, структуры и перечисления:

protocol SomeProtocol {

// определение протокола…

}

Пользовательские типы утверждают, что они принимают протокол, когда они помещают имя протокола после имени типа и разделяются с этим именем двоеточием, то есть указывают эти протоколы как часть их определения. После двоеточия можно указывать множество протоколов, перечисляя их имена через запятую:

struct SomeStructure: FirstProtocol, AnotherProtocol {

// определение структуры…

}

Если у класса есть суперкласс, то вписывается имя суперкласса до списка протоколов, которые он принимает, также разделяется имя суперкласса и имя протокола запятой:

class SomeClass: SomeSuperclass, FirstProtocol, AnotherProtocol {

// определение класса…

}