#5. Принципы ООП: наследование, инкапсуляция, полиморфизм

Задание:

1. Необходимо разработать иерархию наследования, основываясь на базовом классе из примера. При этом:
    a. Иерархия должна включать **МИНИМУМ 2** класса-наследника;
    b. Иерархия должна включать хотя бы один интерфейс, описывающий какие-то общие методы и/или свойства всех наследников;
    c. Иерархия должна включать абстрактный класс;
    d. Один из классов обязан быть **sealed**;
    e. Хотя бы один из классов-наследников должен переопределять поведение базовых методов.
2. Необходимо разработать класс-коллекцию, работающую с вашей иерархией. При этом:
    a. Коллекция должна иметь возможность хранить любой объект-наследник;
    b. Коллекция должна включать в себя следующие методы:
        i. Добавление элемента / удаление элемента;
        ii. Индексатор;
        iii. Метод поиска элемента, принимающий параметром предикат и возвращающий первый элемент, удовлетворяющий условию, либо **null**.
        iv. Удаление элемента должно быть реализовано явно ☺
    c. Реализовать в коллекции методы вывода информации о хранящихся объектах.
3. Разработать демонстрацию, в которой:
    a. Создать объект коллекции;
    b. Добавить в коллекцию несколько объектов разных типов;
    c. Продемонстрировавать работу с объектами через базовый/абстрактный класс или интерфейс. Использовать **as** и **is** операторы.

Повышенный уровень:

1. Реализовать коллекцию как синглтон.
2. Ознакомиться с методами класса **File** https://docs.microsoft.com/en-us/dotnet/api/system.io.file?view=netframework-4.8
3. Реализовать в коллекции метод, сохраняющий информацию об объектах в текстовый файл в следующем виде:
   index, *objType*
   где *index* – индекс элемента в коллекции, *objType* – тип объекта (можно использовать **GetType**)

Подсказка:

Пример иерархии можно посмотреть здесь: https://github.com/demeshchik/oop-example
Однако этот пример **НЕЛЬЗЯ** считать базовым, т.к. некоторые моменты в нем сознательно приведены неверно с архитектурной точки зрения (это не влияет на понимание общей концепции наследования и полиморфизма)

Вопросы:

1. Что такое полиморфизм? Какие виды полиморфизма существуют в C#?
2. Для чего служит ключевое слово **base**?
3. В чем назначение виртуальных функций?
4. Для чего доступны ключевые слова **is** и **as**?
5. Кому доступны переменные с модификатором **protected**?
6. Можно ли запретить наследование от класса?
7. Что может содержать интерфейс?
8. Что такое «**абстрактный класс**»?
9. Назовите отличия между интерфейсом и абстрактным классом.
10. Как используются ограничения при наследовании? Приведите пример.
11. Какие существуют ограничения?

Варианты заданий:

1. Автомобиль.
2. Самолет
3. Книга
4. Журнал
5. Растение
6. Животное
7. Геометрическая фигура
8. Фильм
9. Класс-логгер
10. Семья
11. Персонал (школы, университета, любая административная структура)
12. Товар (в магазине)
13. Телефон (стационарный, таксофон, мобильный и т.д.)
14. Дом
15. Водоем
16. Спортивный инвентарь
17. Испытание (тест, экзамен и т.д.)
18. Программное обеспечение
19. Звезда (небесное тело)
20. Социальная иерархия в Древнем Египте ☺