**Лабораторная работа №11. Часть 2**

*«Классы. Наследование»*

**Рекомендации по программированию**

* Поля предпочтительнее делать private.
* Интерфейс должен быть полным, т.е. предоставлять возможность выполнять любые разумные действия с классом. И одновременно минимально необходимым, т.е. без дублирования и пересечения возможностей методов.
* Каждый метод класса должен решать только одну задачу. Не стоит объединять два коротких независимых фрагмента кода в один метод.
* Если действия встречаются хотя бы дважды, стоит оформить их в отдельные функции.
* Все заданные методы, поля и экземпляры классов должны иметь осмысленные имена.

**Требования к лабораторной**

* Каждый разрабатываемый класс должен, как правило, содержать следующие элементы: поля с различными спецификаторами, конструкторы с параметрами и без параметров, методы, свойства.
* Методы и свойства должны обеспечивать непротиворечивый, полный, минимальный и удобный интерфейс класса.
* При возникновении ошибок должны выбрасываться исключения
* К каждому отчету, помимо формулировки задания и скрина консоли, нужно прикрепить еще диаграмму классов
* Каждый студент должен реализовать свой класс со своими полями и методами. **Любое совпадение не случайно и карается по закону джунглей**.
* Проделанную работу необходимо показать на паре.

**Постановка задачи**

Реализовать иерархию классов.

1. К классу из лабораторной работы №10 прописать его базовый (класс-родитель) **абстрактный класс** (должно быть слово **abstract**), в котором поля и методы будут использоваться для последующих наследников.

Пример: *Класс Покемон с полями имя, пол, вес, рост и др.*

(Соответственно в классе Псайдак указанные поля для прошлой лабораторной, что будут едины для всех наследников класса Покемон, определять не нужно)

1. От абстрактного класса породить еще два новых класса (помимо уже существующего). Определить для них методы абстрактного класса, добавить конструкторы, использующие конструктор абстрактного класса-родителя (через ключевое слово **base**), еще два-три новых поля и метода, один из которых **виртуальный**. Продемонстрировать работу каждого метода в консоли. Можно с использованием массивов персонажей.

Пример: *Класс Пикачу с дополнительными полями удар молнии, прыжок и др.*

*Класс Слоупок с дополнительными полями замедление, вызов дождя (это оказывается канон :с) и др.*

*Класс Псайдак (из прошлого задания с соответствующими полями).*

1. От каждого созданного производного класса построить еще как минимум по два наследника, расширяющие возможности этих классов. Определить конструкторы, использующие конструктор класса-родителя (через ключевое слово **base**), добавить новые поля и реализовать виртуальные методы класса-родителя (через ключевое слово **override**). Продемонстрировать работу каждого метода в консоли. Можно с использованием массивов персонажей.

Пример: *Наследники класса Пикачу: Пичу (с дополнительным полем умиление) и Райчу (с дополнительным полем передвижение на хвосте).*

*Наследники класса Слоупок: Слоубро (с дополнительным полем… ну… командная медлительность?) и Слоукинг (с дополнительным полем медлительности на всех участников боя (включая и союзников :с)).*

*+ наследники класса Псайдак.*

Псы: Совет. Чаще всего виртуальным методом полезно делать метод вывода на экран каких-либо значений или результата для персонажей в иерархии. Но вы можете делать и какой-нибудь еще :)

Псы 2: Написание кода на русском все еще забавно)

Псы 3: Вообще, можно построить более ветвистое дерево наследования. Фантазия, упоротость, труды и мучения в этом направлении могут быть вознаграждены оценкой :3

Псы 4: И не забудьте **схему иерархии** ваших классов!





