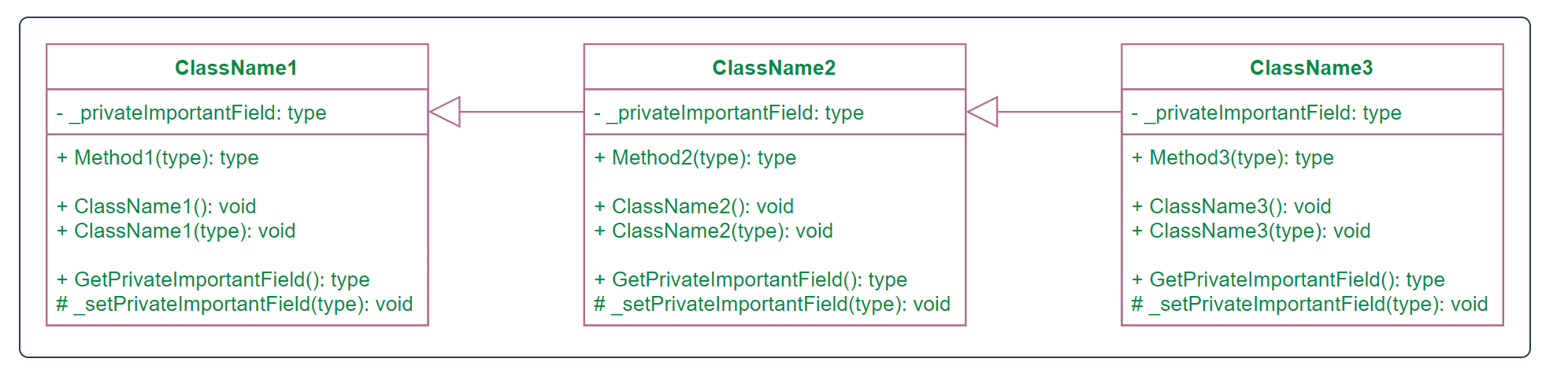
**Лабораторная работа №1 - Классы, объекты, инкапсуляция, наследование**



Дана UML диаграмма классов. У каждого класса есть приватное поле (privateImportantField), конструктор без параметров и один перегруженный конструктор с параметром, значение которого передается в упомянутое поле.

Необходимо придумать осмысленные поле и метод для каждого из классов (осмысленные и для его наследников). По желанию можно сделать классы более конкретными.

**Задание:**

* В draw.io https://app.diagrams.net/, изменить согласно варианту названия классов, полей и методов (ClassName1 – Генератор, ClassName2 - Генератор случайных чисел, ClassName3 - Генератор случайных чисел с нормальным распределением)
* Реализовать иерархию классов в коде. Геттер и сеттер для приватного поля реализовать:
  + для ClassName1 через методы (как на диаграмме);
  + для ClassName2 через свойство;
  + для ClassName3 через автосвойство.
* В основном теле программы создать экземпляры классов разными способами, продемонстрировать доступные и недоступные способы работы с ними.

**Лабораторная работа №2 – Полиморфизм**

**Задание:**

Измените, расширьте и опишите иерархию классов из предыдущей лабораторной, используя:

* Описание и наследование классами как минимум трех интерфейсов.
* Абстрактный класс (с абстрактным методом и абстрактным свойством) в качестве основы полиморфизма.

**Лабораторная работа №3 - Объектно-ориентированное проектирование**

**Задание:**

* По словесному описанию составить UML диаграмму классов (все существительные должны превратиться в классы или поля классов, глаголы - в методы).
* Реализовать в коде спроектированные классы.
* Реализовать описанный текстом сценарий взаимодействия объектов в основном теле программы.

**Словесное описание:**

Возьми густой вишнёвый сок

И белый мамин плащ.

Лей аккуратно сок на плащ –

Появится пятно.

Теперь, чтоб не было пятна

На мамином плаще,

Плащ надо сунуть целиком

В густой вишнёвый сок.

Возьми вишнёвый мамин плащ

И кружку молока.

Лей аккуратно молоко –

Появится пятно.

Теперь, чтоб не было пятна

На мамином плаще,

Плащ надо сунуть целиком

В кастрюлю с молоком.

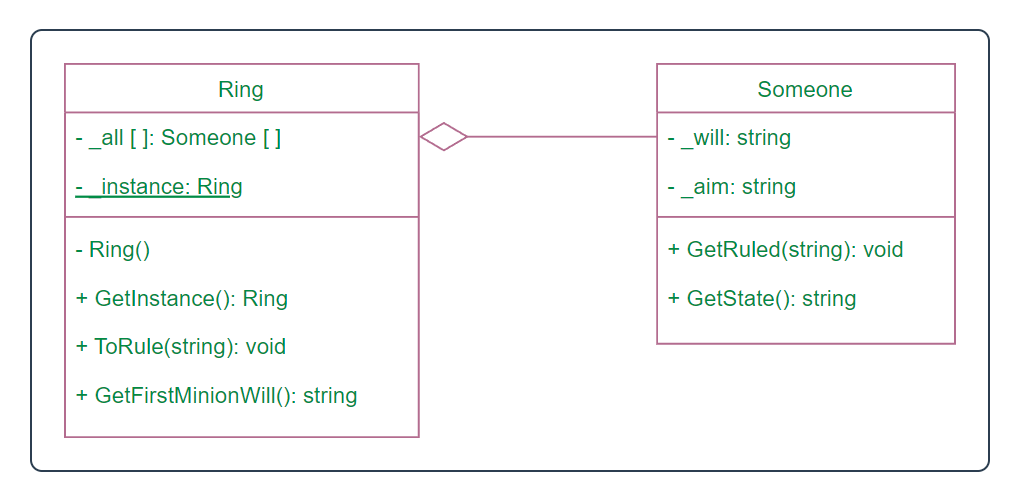
Возьми густой вишнёвый сок

И белый мамин плащ.

Лей аккуратно…

**Пример выполнения работы:**

**Описание**: One Ring to rule them all.

**UML:** 

**Class Ring:**

**class** **Ring**

{

**private** Someone[] \_all;

**private** **static** Ring \_instance;

**private** Ring()

{

\_all = **new** Someone[]{**new** Someone(), **new** Someone(), **new** Someone(), **new** Someone(), **new** Someone(), **new** Someone(), **new** Someone(), **new** Someone(), **new** Someone(),

**new** Someone(), **new** Someone(), **new** Someone(), **new** Someone(), **new** Someone(), **new** Someone(), **new** Someone(),

**new** Someone(), **new** Someone(), **new** Someone()};

}

**public** **static** Ring GetInstance()

{

**if** (\_instance == **null**)

\_instance = **new** Ring();

**return** \_instance;

}

**public** **void** ToRule(**string** s)

{

**foreach** (**var** person **in** \_all)

{

person.GetRuled(s.Split(':')[1]);

}

}

**public** **string** GetFirstMinionWill()

{

**return** \_all[0].GetState();

}

}

**Class Someone:**

**class** **Someone**

{

**private** **string** \_will = "free will";

**private** **string** \_aim = "doing something neutral";

**public** **void** GetRuled(**string** s)

{

\_will = "not free will";

\_aim = s;

}

**public** **string** GetState()

{

**return** "This Someone have " + \_will + " and will " + \_aim;

}

}

**Main:**

Ring ring = Ring.GetInstance();

Console.WriteLine(ring.GetFirstMinionWill());

ring.ToRule("Dark Lord's will: retrieve the Ring!");

Console.WriteLine(ring.GetFirstMinionWill());

**Результат:**

This Someone have free will and will doing something neutral

This Someone have not free will and will retrieve the Ring!