Создать библиотеку классов **Elections,** симулирующую выборы. В билиотеке объявить:

Перечисление(**enum**) Party, включающее в себя следующие элементы:

Liberals,

Communists

Democrats,

Conservatives

**class Candidate** реализующий интерфейс  **IComparable:**

* Целочисленное поле **Id**(уникальное для каждого объекта класса)
* Строковое автосвойство **Name** с приватный сеттером
* Публичное автосвойсво с приватным сеттером **Party party**
* Целочисленное автосвойство с **internal** сеттером **Votes**
* Приватное вещественное поле **percentageofvotes**
* Публичное свойство с **internal** сеттером **PercentageOfVotes,** выдающее ошибку если **value** не лежит в промежутке [0,100].
* Переопределить **Equals(object obj)**  от **Id**
* Переопределить **ToString()**

**Class PollingStation:**

* Список типа **Candidate**
* Свойство типа **Candidate,** выдающее ошибку если **value==null**
* Целочисленное поле **CountOfVotes**
* Метод **GetVote(Candidate candidate),** добавляющее ко всем голосам и к голосам кандидата единицу
* Метод **Candidate FinishElections(),** высчитывающий процент голосов для каждого кандидата, сортирующий список кандидатов по убыванию. Метод возвращает кандидата с наибольшим количеством голосов
* Переопределить **ToString()** , чтобы он выдавал всех кандидатов, участвующий в выборах

**static class Randomize:**

* **Static Random rnd**
* **Static string RandomName()**
* **Static List<Candidate> CreateNewCandidates()**

В основной программе создайте экземпляр класса **PullingStation,** случайное количество раз вызовете метод **GetVote.** Выведеие на консоль победителя выборов, а после и всех участников в порядке убывания их процента голосов.

После сериализуйте объект класса **PullingStation** c помощью JsonSerialization.

**Hint.** Json очень легко читаемый формат и проверять его работу можно просто открыв файл, куда мы сериализовали объект класса и проверит на глаз