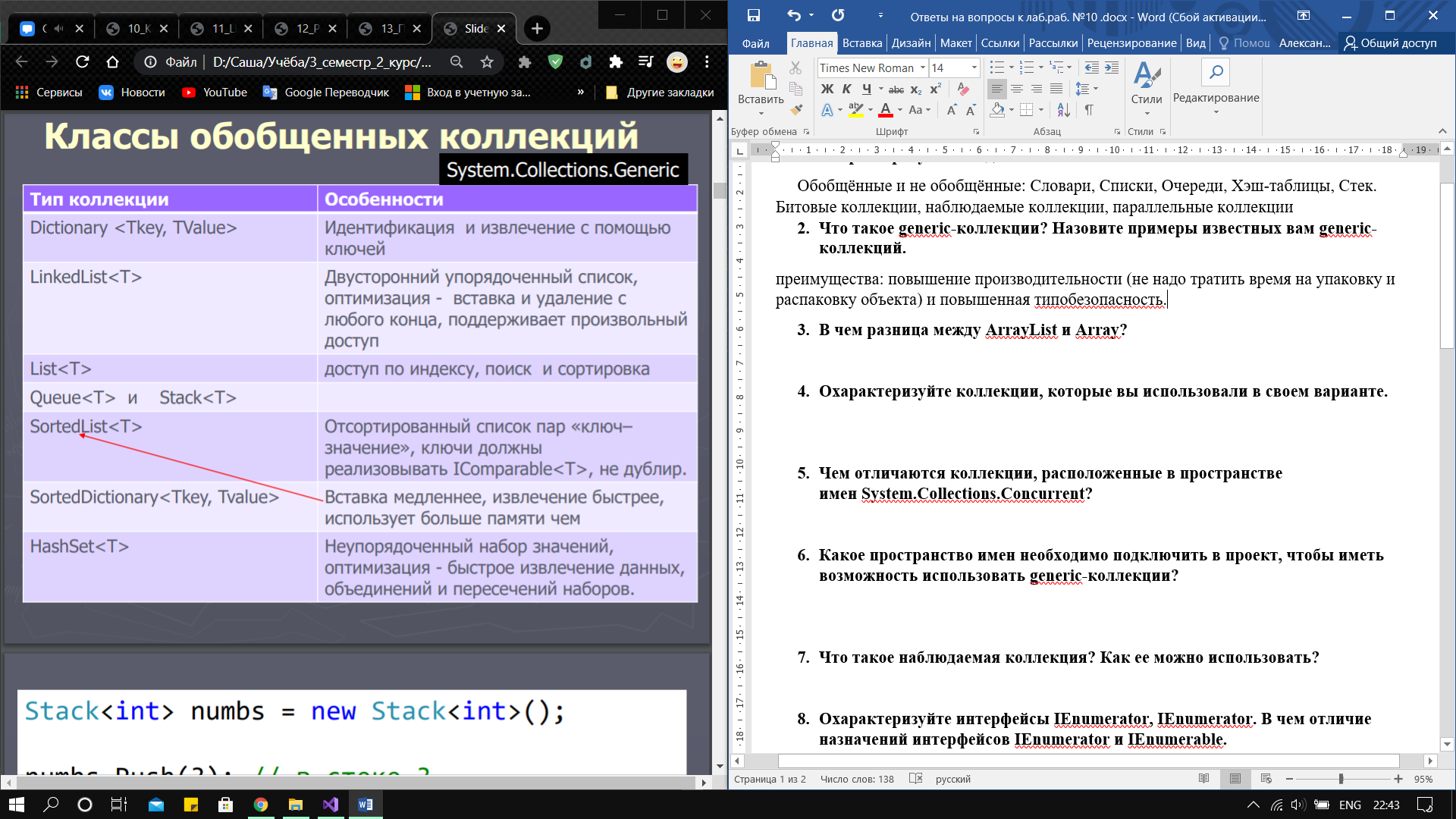
1. **На какие основные виды/типы делятся все коллекции .NET? Охарактеризуйте каждый из них.**

Обобщённые и не обобщённые: Словари, Списки, Очереди, Хэш-таблицы, Стек.

Битовые коллекции, наблюдаемые коллекции, параллельные коллекции

1. **Что такое generic-коллекции? Назовите примеры известных вам generic-коллекций.**

преимущества: повышение производительности (не надо тратить время на упаковку и распаковку объекта) и повышенная типобезопасность.



1. **В чем разница между ArrayList и Array?**

Класс ArrayList определяется массив переменной длины, который состоит из ссылок на объекты и может динамически увеличивать и уменьшать свой размер.

Массив представляет собой низкоуровневую структуру данных, которая по существу отображает область в памяти. ArrayList - это список переменной длины, реализованный как массив object который перераспределяется по мере роста списка.

1. **Охарактеризуйте коллекции, которые вы использовали в своем варианте.**

**BlockingCollection<T>**

Коллекция, которая осуществляет блокировку и ожидает, пока не появится возможность выполнить действие по добавлению или извлечению элемента. BlockingCollection<T> предлагает интерфейс для добавления и извлечения элементов методами Add() и Take(). Эти методы блокируют поток и затем ожидают, пока не появится возможность выполнить задачу.

1. **Чем отличаются коллекции, расположенные в пространстве имен System.Collections.Concurrent?**

Параллельные коллекции - коллекции классов, предназначенные для безопасной работы в многопоточной среде, которыми можно воспользоваться при создании многопоточных приложений.

1. **Какое пространство имен необходимо подключить в проект, чтобы иметь возможность использовать generic-коллекции?**

System.Collections.Concurrent

1. **Что такое наблюдаемая коллекция? Как ее можно использовать?**

**ObservableCollection<T>**

* пользовательский интерфейс получает информацию об изменениях коллекции
* унаследован от Collection<T>, использует внутри себя List<T>, INotifyCollectionChanged

1. **Охарактеризуйте интерфейсы IEnumerator, IEnumerator. В чем отличие назначений интерфейсов IEnumerator и IEnumerable.**

Интерфейс IEnumerable имеет метод, возвращающий ссылку на другой интерфейс – перечислитель. А интерфейс IEnumerator определяет функционал для перебора внутренних объектов в контейнере:

public interface IEnumerator

{

    bool MoveNext(); // перемещение на одну позицию вперед в контейнере элементов

    object Current {get;}  // текущий элемент в контейнере

    void Reset(); // перемещение в начало контейнера

}

1. **Поясните принцип работы коллекций:**
2. **LinkedList <T>**

Сохраняет элементы в двунаправленном списке.

1. **HashSet <T>**

Сохраняет ряд уникальных значений, используя хештаблицу.

1. **Dictionary <Tkey, TValue>**

Сохраняет пары "ключ-значение". Обеспечивает такие же функциональные возможности, как и необобщенный класс Hashtable.

1. **ConcurrentBag <Tkey, TValue>**

Этот класс не определяет никакого порядка для добавления или извлечения элементов. Он реализует концепцию отображения потоков на используемые внутренне массивы, и старается избежать блокировок. Для доступа к элементам применяются методы Add(), TryPeek() и TryTake().

1. **Stack<t>, Queue<T>**

Создает стек. Обеспечивает такие же функциональные возможности, как и необобщенный класс Stack.

Создает очередь. Обеспечивает такие же функциональные возможности, как и необобщенный класс Queue.

1. **SortedList, SortedList.**

Определяет отсортированный список пар "ключ-значение".