1. [На какие основные виды/типы делятся все коллекции .NET? Охарактеризуйте каждый из них.](#_1._Основные_виды/типы)
2. [Что такое generic-коллекции? Назовите примеры известных вам generic-коллекций.](#_2._Generic-коллекции:)
3. [В чем разница между ArrayList и Array?](#_3._Разница_между)
4. [Охарактеризуйте коллекции, которые вы использовали в своем варианте.](#_4._Описание_коллекций)

1. [Чем отличаются коллекции, расположенные в пространстве имен System.Collections.Concurrent?](#_5._Коллекции_в)
2. [Какое пространство имен необходимо подключить в проект, чтобы иметь возможность использовать generic-коллекции?](#_6._Пространство_имен)
3. [Что такое наблюдаемая коллекция? Как ее можно использовать?](#_7._Наблюдаемая_коллекция)
4. [Охарактеризуйте интерфейсы IEnumerator, IEnumerator. В чем отличие назначений интерфейсов IEnumerator и IEnumerable.](#_8._Интерфейсы_IEnumerator)
5. [Поясните принцип работы коллекций:](#_9._Принцип_работы)

a. LinkedList <T>

b. HashSet <T>

c. Dictionary <Tkey, TValue>

d. ConcurrentBag <Tkey, TValue>

e. Stack<t>, Queue<T>

f. SortedList, SortedList.

# 1. Основные виды/типы коллекций в .NET:

- Non-generic коллекции (System.Collections): Предоставляют базовые структуры данных (например, ArrayList, Queue, Stack) и основные интерфейсы (например, ICollection, IEnumerable).

- Generic коллекции (System.Collections.Generic): Предоставляют типобезопасные версии структур данных и интерфейсов (например, List<T>, Dictionary<TKey, TValue>, HashSet<T>).

- Concurrent коллекции (System.Collections.Concurrent): Разработаны для поддержки многопоточности и предоставляют безопасный доступ к коллекциям из нескольких потоков.

# 2. Generic-коллекции:

- Generic-коллекции предоставляют типобезопасные структуры данных.

- Примеры:

- List<T> - список.

- Dictionary<TKey, TValue> - словарь.

- HashSet<T> - множество.

# 3. Разница между ArrayList и Array:

- Array: Фиксированный размер, элементы должны быть одного типа.

- ArrayList: Динамический размер, может содержать элементы разных типов, но теряет преимущества типобезопасности.

# 4. Описание коллекций в варианте (по заданию):

- List<Car>: Универсальный список объектов класса Car.

- Dictionary<string, Car>: Словарь, где ключ - строка (название модели автомобиля), значение - объект класса Car.

# 5. Коллекции в System.Collections.Concurrent:

- Предоставляют коллекции, которые безопасны для использования в многопоточных сценариях.

- Примеры:

- ConcurrentBag<T> - мешок для хранения элементов.

- ConcurrentQueue<T> - потокобезопасная очередь.

- ConcurrentDictionary<TKey, TValue> - потокобезопасный словарь.

# 6. Пространство имен для generic-коллекций:

- Для использования generic-коллекций необходимо подключить пространство имен System.Collections.Generic.

# 7. Наблюдаемая коллекция (ObservableCollection<T>):

- Реализует интерфейс INotifyCollectionChanged, предоставляя события об изменениях в коллекции.

- Можно использовать для автоматического обновления пользовательского интерфейса при изменении данных в коллекции.

# 8. Интерфейсы IEnumerator и IEnumerable:

- IEnumerator: Предоставляет методы для перебора элементов в коллекции.

- IEnumerable: Предоставляет метод GetEnumerator, возвращающий объект IEnumerator. Используется для поддержки foreach.

# 9. Принцип работы коллекций:

- LinkedList<T>: Связанный список, где каждый элемент хранит ссылку на предыдущий и следующий элемент.

- HashSet<T>: Множество, не содержащее дубликатов элементов.

- Dictionary<TKey, TValue>: Словарь, где каждому ключу соответствует значение.

- ConcurrentBag<T>: Коллекция, поддерживающая добавление элементов из нескольких потоков.

- Stack<T> и Queue<T>: Стек и очередь, соответственно.

- SortedList и SortedDictionary: Коллекции, автоматически сортирующие элементы по ключу. Разница в реализации динамического изменения размера и производительности.