1. На какие основные виды/типы делятся все коллекции .NET? Охарактеризуйте

каждый из них.

**Краткий обзор коллекций**

Главное преимущество коллекций заключается в том, что они стандартизируют обработку групп объектов в программе. Все коллекции разработаны на основе набора четко определенных интерфейсов. Некоторые встроенные реализации таких интерфейсов, в том числе ArrayList, Hashtable, Stack и Queue, могут применяться в исходном виде и без каких-либо изменений. Имеется также возможность реализовать собственную коллекцию.

**В среде .NET Framework поддерживаются пять типов коллекций: обобщенные,** **необобщенные, специальные, с поразрядной организацией , параллельные.**

**Обобщенные коллекции**

Обеспечивают обобщенную реализацию нескольких стандартных структур данных, включая связные списки, стеки, очереди и словари. Такие коллекции являются типизированными в силу их обобщенного характера. Это означает, что в обобщенной коллекции могут храниться только такие элементы данных, которые совместимы по типу с данной коллекцией. Благодаря этому исключается случайное несовпадение типов. Обобщенные коллекции объявляются в пространстве имен System.Collections.Generic.

**Необобщенные коллекции**

Реализуют ряд основных структур данных, включая динамический массив, стек, очередь, а также словари, в которых можно хранить пары "ключ-значение". В отношении необобщенных коллекций важно иметь в виду следующее: они оперируют данными типа object. Таким образом, необобщенные коллекции могут служить для хранения данных любого типа, причем в одной коллекции допускается наличие разнотипных данных. Очевидно, что такие коллекции не типизированы, поскольку в них хранятся ссылки на данные типа object. Классы и интерфейсы необобщенных коллекций находятся в пространстве имен System.Collections.

**Специальные коллекции**

Оперируют данными конкретного типа или же делают это каким-то особым образом. Например, имеются специальные коллекции для символьных строк, а также специальные коллекции, в которых используется однонаправленный список. Специальные коллекции объявляются в пространстве имен System.Collections.Specialized.

**Поразрядная коллекция**

В прикладном интерфейсе Collections API определена одна коллекция с поразрядной организацией — это BitArray. Коллекция типа BitArray поддерживает поразрядные операции, т.е. операции над отдельными двоичными разрядами, например И, ИЛИ, исключающее ИЛИ, а следовательно, она существенно отличается своими возможностями от остальных типов коллекций. Коллекция типа BitArray объявляется в пространстве имен System.Collections.

**Параллельные коллекции**

Поддерживают многопоточный доступ к коллекции. Это обобщенные коллекции, определенные в пространстве имен System.Collections.Concurrent.

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

Их использование перед необобщенными коллекциями имеет те же преимущества: повышение производительности (не надо тратить время на упаковку и распаковку объекта) и повышенная типобезопасность

**List<T>**

1. В чем разница между ArrayList и Array?

массив может содержать как примитивные типы данных, так и объекты. Но ArrayList не может содержать примитивные типы данных. Он содержит только объекты

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

**BlockingCollection<T>** — это потокобезопасный класс коллекции, обеспечивающий следующие возможности:

реализует шаблон "производитель-получатель";

поддерживает параллельное добавление и извлечение элементов из нескольких потоков;

допускает указание максимальной емкости;

поддерживает операции вставки и удаления, блокирующиеся при опустошении или заполнении коллекции;

поддерживает условные операции вставки и удалении, не блокирующиеся или блокирующиеся лишь на определенное время;

инкапсулирует все типы коллекций, реализующие интерфейс IProducerConsumerCollection<T>;

поддерживает отмену с помощью токенов отмены;

5. Чем отличаются коллекции, расположенные в пространстве имен

System.Collections.Concurrent?

Однако доступ к элементам объекта коллекции через методы расширения или явные реализации интерфейса не обязательно будет потокобезопасным и может потребовать синхронизацию с вызывающим объектом.

6. Какое пространство имен необходимо подключить в проект, чтобы иметь

возможность использовать generic-коллекции?

using System.Collections.Generic;

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

ObservableCollection – класс определенный в пространстве имен System

bservableCollection<Game<string>> ObsColl = new ObservableCollection<Game<string>>()

{

new Game<string>("Игра1"),

new Game<string>("Игра2")

};

ObsColl.CollectionChanged += ObsColl\_CollectionChanged;

ObsColl.Add(new Game<string>("Killer\_Bob"));

void ObsColl\_CollectionChanged(object? sender, NotifyCollectionChangedEventArgs e)

8. Охарактеризуйте интерфейсы IEnumerator, IEnumerable. В чем отличие

назначений интерфейсов IEnumerator и IEnumerable.   
IEnumerator

Поддерживает простой перебор элементов универсальной коллекции.

IEnumerable

Предоставляет перечислитель, который поддерживает простой перебор элементов неуниверсальной коллекции.

1. Поясните принцип работы коллекций:

**Dictionary<Tkey, TValue>**

Сохраняет пары "ключ-значение". Обеспечивает такие же функциональные

возможности, как и необобщенный класс Hashtable

**HashSet<T>**

Сохраняет ряд уникальных значений, используя хештаблицу

**LinkedList<T>**

Сохраняет элементы в двунаправленном списке

**List<T>**

Создает динамический массив. Обеспечивает такие же функциональные

возможности, как и необобщенный класс ArrayList

**Queue<T>**

Создает очередь. Обеспечивает такие же функциональные возможности, как и

необобщенный класс Queue

**SortedDictionary<TKey, TValue>**

Создает отсортированный список из пар "ключ-значение"

**SortedList<TKey, TValue>**

Создает отсортированный список из пар "ключ-значение". Обеспечивает такие

же функциональные возможности, как и необобщенный класс SortedList

**SortedSet<T>**

Создает отсортированное множество

**Stack<T>**

Создает стек. Обеспечивает такие же функциональные возможности, как и

необобщенный класс Stack