1. Перечислите стандартные коллекции NET Framework.

хэш-таблицы, очереди, стеки, контейнеры, словари и списки.

2. Поясните принцип работы коллекции:

a) Stack<T>

Создает стек.

b) Queue<T>

Создает очередь.

c) HashSet<T>

Сохраняет ряд уникальных значений, используя хештаблицу

d) List<T>

Создает динамический массив.

e) Dictionary<Tkey, TValue>

Сохраняет пары "ключ-значение".

f) LinkedList<T>

Сохраняет элементы в двунаправленном списке

g) SortedDictionary<TKey, TValue>

Создает отсортированный список из пар "ключ-значение"

h) SortedList<TKey, TValue>

Создает отсортированный список из пар "ключ-значение".

i) SortedSet<T>

Создает отсортированное множество

3. Охарактеризуйте необобщенные, специальные, с поразрядной организацией,

обобщенные и параллельные коллекции.

*Необобщенные коллекции*

Реализуют ряд основных структур данных. 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.

*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 вобобщенной коллекции могут храниться только такие элементы данных, которыесовместимы по типу с данной коллекцией. Благодаря этому исключается случайноенесовпадение типов. Обобщенные коллекции объявляются в пространстве имен**System.Collections.Generic**.

*Параллельные коллекции*

Поддерживают многопоточный доступ к коллекции. Это обобщенные

коллекции, определенные в пространстве имен **System.Collections.Concurrent**.

В пространстве имен System.Collections.ObjectModel находится также ряд

классов, поддерживающих создание пользователями собственных обобщенных коллекций.

4. Какие интерфейсы используются в коллекциях C#?

**ICollection<T>**

Определяет основополагающие свойства обобщенных коллекций

**IComparer<T>**

Определяет обобщенный метод Compare() для сравнения объектов, хранящихся в коллекции

**IDictionary<Tkey, TValue>**

Определяет обобщенную коллекцию, состоящую из пар "ключ-значение"

**IEnumerable<T>**

Определяет обобщенный метод GetEnumerator(), предоставляющий

перечислитель для любого класса коллекции

**IEnumerator<T>**

Предоставляет методы, позволяющие получать содержимое коллекции по

очереди

**IEqualityComparer<T>**

Сравнивает два объекта на предмет равенства

**IList<T>**

Определяет обобщенную коллекцию, доступ к которой можно получить с

помощью индексатора

5. Для чего используется интерфейс IComparable?

Определяет обобщенный метод сравнения для типа, который реализуется типом значения или классом, чтобы упорядочить или сортировать его экземпляры.

6. Что содержит интерфейс IEnumerator или обобщенный интерфейс

IEnumerator<T>? Где и как его можно использовать?

Содержит перечеслители, которые обеспечивают стандартный способ поочередного доступа к элементам коллекции. Интерфейс IEnumerator определяет функционал для перебора внутренних объектов в коллекции. (например, цикл foreach)

7. Что такое наблюдаемая коллекция? Где и каким образом ее можно

использовать?

Представляет коллекцию динамических данных, обеспечивающих выдачу уведомлений при получении и удалении элементов или при обновлении всего списка.

Коллекция предназначена для того, чтобы пользовательский интерфейс мог получать информацию об изменениях коллекции. Используется в случае если нужна информация о том, когда элементы коллекции удаляются или добавляются.