**Класс Stack**

В классе Stack определены следующие конструкторы:

public Stack(); //создает пустой стек, начальная вместимость которого равна 10

public Stack(int capacity); // создает пустой стек, начальная вместимость которого равна capacity

public Stack(ICollection c); //создает стек, который содержит элементы коллекции, заданной

//параметром с, и аналогичной (аналогичной - с чем?) вместимостью

Кроме методов, определенных в интерфейсах, реализуемых классом Stack, в этом классе определены собственные методы:

|  |  |
| --- | --- |
| **Метод** | **Описание** |
| public virtual bool Contains(object v) | Возвращает значение true, если объект v содержится в вызывающем стеке, в противном случае возвращает значение false. |
| public virtual void Clear() | Устанавливает свойство Сount равным нулю, тем самым очищая стек. |
| public virtual object Peek() | Возвращает элемент, расположенный в вершине стека, но не извлекая его из стека |
| public virtual object Pop() | Возвращает элемент, расположенный в вершине стека, и извлекает его из стека |
| public virtual void Push(object v) | Помещает объект v в стек |
| public virtual object[] ToArray() | Возвращает массив, который содержит копии элементов вызывающего стека |

**Класс Queue**

В классе Queue определены следующие конструкторы:

public Queue(); //создает пустую очередь, начальная вместимость которой равна 32

public Queue (int capacity); // создает пустую очередь, начальная вместимость которой равна capacity

//создает пустую очередь, начальная вместимость которой равна capacity, и коэффициент роста

//устанавливается параметром n

public Queue (int capacity, float n);

//создает очередь, которая содержит элементы коллекции, заданной параметром с, и аналогичной

//вместимостью

public Queue (ICollection c);

Кроме методов, определенных в интерфейсах, реализуемых классом Queue, в этом классе определены собственные методы:

|  |  |
| --- | --- |
| **Метод** | **Описание** |
| public virtual bool Contains (object v) | Возвращает значение true, если объект v содержится в вызывающей очереди, в противном случае возвращает значение false |
| public virtual void clear () | Устанавливает свойство Count равным нулю, тем самым очищая очередь |
| public virtual object Dequeue () | Возвращает объект из начала вызывающей очереди, удаляя его из очереди |
| public virtual object Peek () | Возвращает объект из начала вызывающей очереди, не удаляя его из очереди |
| public virtual void Enqueue(object v) | Добавляет объект v в конец очереди |
| public virtual object [ ] ToArrау () | Возвращает массив, который содержит копии элементов из вызывающей очереди |
| public virtual void TrimToSizeO | Устанавливает свойство Capacity равным значению свойства Count |

**Класс ArrayList**

Класс ArrayList реализует интерфейсы ICollection, IList, IEnumerable и ICloneable. В классе ArrayList определены следующие конструкторы:

//создает пустой массив с максимальной емкостью равной 16 элементам, при текущей размерности 0

public ArrayList()

public ArrayList(int capacity) //создает массив с заданной емкостью capacity, при текущей размерности 0

public ArrayList(ICollection с) //строит массив, который инициализируется элементами коллекции с

Помимо методов, определенных в интерфейсах, которые реализует класс ArrayList, в нем определены и собственные методы:

|  |  |
| --- | --- |
| **Метод** | **Описание** |
| public virtual void AddRange (ICollection с) | Добавляет элементы из коллекции с в конец вызывающей коллекции |
| public virtual int BinarySearch (object v) | В вызывающей отсортированной коллекции выполняет поиск значения, заданного параметром v. Возвращает индекс найденного элемента. Если искомое значение не обнаружено, возвращает отрицательное значение. |
| public virtual int BinarySearch (object v, IComparer comp) | В вызывающей отсортированной коллекции выполняет поиск значения, заданного параметром v, на основе метода сравнения объектов, заданного параметром соmp. Возвращает индекс найденного элемента. Если искомое значение не обнаружено, возвращает отрицательное значение. |
| public virtual int BinarySearch (int startldx, int count, object v, IComparer comp) | В вызывающей отсортированной коллекции выполняет поиск значения, заданного параметром v, на основе метода сравнения объектов, заданного параметром comp. Поиск начинается с элемента, индекс которого равен значению startIdx, и включает count элементов. Метод возвращает индекс найденного элемента. Если искомое значение не обнаружено, возвращает отрицательное значение. |
| public virtual void CopyTo(Array ar, int startIdx) | Копирует содержимое вызывающей коллекции, начиная с элемента, индекс которого равен значению startIdx, в массив, заданный параметром аr. Приемный массив должен быть одномерным и совместимым по типу с элементами коллекции. |
| public virtual void CopyTo(int srcldx, Array ar, int destIdx, int count) | Копирует count элементов вызывающей коллекции, начиная с элемента, индекс которого равен значению srcIdx, в массив, заданный параметром аr, начиная с элемента, индекс которого равен значению destIdx. Приемный массив должен быть одномерным и совместимым по типу с элементами коллекции |
| public virtual ArrayList GetRange(int idx, int count) | Возвращает часть вызывающей коллекции типа ArrayList. Диапазон возвращаемой коллекции начинается с индекса idx и включает count элементов. Возвращаемый объект ссылается на те же элементы, что и вызывающий объект |
| public static ArrayList FixedSize(ArrayList ar) | Превращает коллекцию ar в ArrayList-массив с фиксированным размером и возвращает результат |
| public virtual void InsertRange(int startldx, ICollection c) | Вставляет элементы коллекции, заданной параметром с, в вызывающую коллекцию, начиная с индекса, заданного параметром startIdx |
| public virtual int LastlndexOf(object v) | Возвращает индекс последнего вхождения объекта v в вызывающей коллекции. Если искомый объект не обнаружен, возвращает отрицательное значение |
| public static ArrayList Readonly(ArrayList ar) | Превращает коллекцию ar в ArrayList-массив, предназначенный только для чтения |
| public virtual void RemoveRange(int idx, int count) | Удаляет count элементов из вызывающей коллекции, начиная с элемента, индекс которого равен значению idx |
| public virtual void Reverse() | Располагает элементы вызывающей коллекции в обратном порядке |
| public virtual void Reverse(int startldx, int count) | Располагает в обратном порядке count элементов вызывающей коллекции, начиная с индекса startldx |
| public virtual void SetRange(int startldx, ICollection c) | Заменяет элементы вызывающей коллекции, начиная с индекса startldx, элементами коллекции, заданной параметром с |
| public virtual void Sort() | Сортирует коллекцию по возрастанию |
| public virtual void Sort(IComparer comp) | Сортирует вызывающую коллекцию на основе метода сравнения объектов, заданного параметром comp. Если параметр comp имеет нулевое значение, для каждого объекта используется стандартный метод сравнения |
| public virtual void Sort ( int startidx, int endidx, icomparer comp) | Сортирует часть вызывающей коллекции на основе метода сравнения объектов, заданного параметром comp. Сортировка начинается с индекса startidx и заканчивается индексом endidx. Если параметр comp имеет нулевое значение, для каждого объекта используется стандартный метод сравнения |
| public virtual object [ ] ToArray () | Возвращает массив, который содержит копии элементов вызывающего объекта |
| public virtual Array ToArray (Type type) | Возвращает массив, который содержит копии элементов вызывающего объекта. Тип элементов в этом массиве задается параметром type |
| public virtual void TrimToSize() | Устанавливает свойство Capacity равным значению свойства Count |

**Класс Hashtable**

Класс Hashtable предназначен для создания коллекции, в которой для хранения объектов используется хеш-таблица. В хеш-таблице для хранения информации используется механизм, именуемый хешированием (hashing). Суть хеширования состоит в том, что для определения уникального значения, которое называется хеш-кодом, используется информационное содержимое соответствующего ему ключа. Хеш-код затем используется в качестве индекса, по которому в таблице отыскиваются данные, соответствующие этому ключу. Преобразование ключа в хеш-код выполняется автоматически, т.е. сам хеш-код вы даже не увидите. Но преимущество хеширования - в том, что оно позволяет сокращать время выполнения таких операций, как поиск, считывание и запись данных, даже для больших объемов информации.

В классе Hashtable определено несколько конструкторов:

public Hashtable() //создает пустую хеш-таблицу

// строит хеш-таблицу, которая инициализируется элементами коллекции с

public Hashtable(IDictionary с)

public Hashtable(int capacity) //создает хеш-таблицу с вместимостью capacity

//создает хеш-таблицу вместимостью capacity и коэффициентом заполнения n

public Hashtable(int capacity, float n)

Помимо методов, определенных в интерфейсах, которые реализует класс Hashtable, в нем определены и собственные методы:

|  |  |
| --- | --- |
| **Метод** | **Описание** |
| public virtual bool ContainsKey (object k) | Возвращает значение true , если в вызывающей хеш-таблице содержится ключ, заданный параметром k. В противном случае возвращает значение false |
| public virtual bool ContainsValue (object v) | Возвращает значение true, если в вызывающей хеш-таблице содержится значение, заданное параметром v. В противном случае возвращает значение false |
| public virtual IDictionaryEnumerator GetEnumerator() | Возвращает для вызывающей хеш-таблицы нумератор типа IDictionaryEnumerator |