РК №1 Ванеев Артем ИУ5-34Б

«Как можно реализовать класс однонаправленного списка без применения стандартных коллекций?»

Рассмотрим реализацию с нуля простого однонаправленного списка, где в качестве наследника выступает класс стека, а сами классы будут созданы без использования стандартных коллекций.

```
using System;
namespace FigureCollections {
        /// <summary>
         /// Элемент списка
         /// </summary>
         public class SimpleListItem<T> {
                 /// <summary>
                 /// Данные
                /// </summary>
                public T data { get; set; }
                        /// <summary>
                       /// Следующий элемент
                        /// </summary>
                public SimpleListItem<T> next { get; set; }
                 ///конструктор
                public SimpleListItem(T param) {
                        this.data = param;
                }
        }
}
```

Тип-обобщение T – тип данных списка. Класс SimpleListItem содержит свойство data обобщенного типа, предназначенное для хранения данных. В классе SimpleListItem также содержится свойство next типа SimpleListItem, являющееся аналогом указателя на следующий элемент.

При объявлении переменной типа объект класса в
данной переменной данные будут храниться именно как ссылка на объект класса, то есть фактически как указатель.

Класс SimpleList реализует список. Поля first и last содержат ссылки на для первого и последнего элемента списка. Свойство Count содержит количество элементов.

Метод добавления нового элемента в конец списка Add создает новый контейнер элемента на основе переданных данных (объект класса SimpleListItem), увеличивает количество элементов списка на единицу, а затем добавляет контейнер к цепочке контейнеров. В результате этого поле first будет содержать ссылку на первый, а last – на последний элементы списка.

Метод GetItem используется для получения контейнера по его порядковому номеру, в частности для контейнера с номером N метод перебирает в цикле N-1 контейнеров и возвращает N-ый контейнер. Метод Get возвращает данные, находящиеся в контейнере.

```
using System;
namespace FigureCollections
{
        /// <summary>
        /// Класс стек
        /// </summary>
        class SimpleStack<T> : SimpleList<T> where T : IComparable {
                /// <summary>
                /// Добавление в стек
                /// </summary>
                public void Push(T element) {
                        Add(element);
                }
                /// <summary>
```

```
        /// Удаление и чтение из стека
        /// </summary>
        public T Pop() {
            T Result = default(T);
            if (this.Count == 0) return Result;
            if (this.Count == 1) {
                Result = this.first.data;
                //обнуляются указатели начала и конца списка
                 this.first = null;
                 this.last = null;
            }
            Else {
                SimpleListItem<T> newLast = this.GetItem(this.Count - 2);
                 //Чтение значения из последнего элемента
                 Result = newLast.next.data;
                //предпоследний элемент считается последним
                this.last = newLast;
                //последний элемент удаляется из списка
                newLast.next = null;
            }
            this.Count--;
            return Result;
        }
    }
}
```

Класс стека наследует от класса списка все необходимые методы, за исключением собственно методов стека – Push (запись в стек) и Pop (чтение с удалением из стека).

При реализации стека на основе списка необходимо решить, что будет вершиной стека – начало или конец списка.

Метод Push просто вызывает метод Add для добавления данных в конец списка.

Метод Pop возвращает последний элемент списка, а затем удаляет его из списка. Реализация метода зависит от количества элементов в списке. Если список не содержит элементов, то возвращается значение по умолчанию для обобщенного типа default(T). Если список содержит один элемент, то он возвращается, а список переводится в состояние пустого списка. Если список содержит более одного элемента, то возвращается и удаляется из списка последний элемент, а предпоследний элемент устанавливается в качестве последнего.