Упражнения: Имплементация на разтеглив масив

1. Имплементация на разтеглив масив ArrayList<T>

Имплементирайте структура от данни **`ArrayList<T>`** която съдържа поредица от елементи от шаблонен тип **`T`**. Структурата трябва да пази поредица от елементи в масив. Структурата трябва да има капацитет, който расте двойно, когато се препълни, като в началото винаги има 2 елемента. Масивът трябва да поддържа следните операции:

- **`int Count`** → дава **броя** на елементите в структурата
- **`T this[int index]`** → индексатор, който служи за достъпване на елементите по техния **индекс** (в интервал **`0`** … **`Count-1`**)
- **`void Add(T item)`** → добавя елемент към поредицата и **удвоява** капацитета на масива, ако е запълнен
- **`T RemoveAt(int index)`** → премахва елемента по неговия **индекс** (в интервала **`0`** … **`Count-1`**) и връща елемента

Винаги тествайте имплементираните операции преди да продължите с реализацията на следващата

Примери

```
static void Main(string[] args)
{
    ArrayList<int> list = new ArrayList<int>();
    list.Add(5);
    list[0] = list[0] + 1;
    int element = list.RemoveAt(0);
}
```

Решение

Създайте класа **`ArrayList<T>`**

```
public class ArrayList<T>
{
    private const int Initial_Capacity = 2;

    private T[] items;

    public ArrayList()
    {
        this.items = new T[Initial_Capacity];
    }
}
```

Добавете **`Count`** и **`индексатор`**

```
public int Count { get; private set; }

public T this[int index]
{
    get...

    set...
}
```

Имплементирайте **get** операцията по индекс

```
get
{
    if (index >= this.Count)
    {
        throw new ArgumentOutOfRangeException();
    }

    return this.items[index];
}
```

А след това и **set** операцията по индекс

```
set
{
    if (index >= this.Count)
    {
        throw new ArgumentOutOfRangeException();
    }

    this.items[index] = value;
}
```

Реализирайте **Add** и **Resize** методите

```
public void Add(T item)
{
    if (this.Count == this.items.Length)
    {
        this.Resize();
    }

    this.items[this.Count++] = item;
}
```

```
private void Resize()
{
    T[] copy = new T[this.items.Length * 2];
    for (int i = 0; i < this.items.Length; i++)
    {
        copy[i] = this.items[i];
    }

    this.items = copy;
}
```

Накрая, реализирайте **RemoveAt**, **Shrink** и **Shift** методите

```
public T RemoveAt(int index)
{
    if (index >= this.Count)
    {
        throw new ArgumentOutOfRangeException();
    }

    T element = this.items[index];
    this.items[index] = default(T);
    this.Shift(index);
    this.Count--;

    if (this.Count <= this.items.Length / 4)
    {
        this.Shrink();
    }

    return element;
}
```

```
private void Shift(int index)
{
    for (int i = index; i < this.Count; i++)
    {
        this.items[i] = this.items[i + 1];
    }
}
```

```
private void Shrink()
{
    T[] copy = new T[this.items.Length / 2];
    for (int i = 0; i < this.Count; i++)
    {
        copy[i] = this.items[i];
    }

    this.items = copy;
}
```

Министерство на образованието и науката (МОН)

- Настоящият курс (презентации, примери, задачи, упражнения и др.) е разработен за нуждите на Национална програма "**Обучение за ИТ кариера**" на МОН за подготовка по професия "Приложен програмист".

- Курсът е базиран на учебно съдържание и методика, предоставени от **фондация "Софтуерен университет"** и се разпространява под **свободен лиценз CC-BY-NC-SA** (Creative Commons Attribution-Non-Commercial-Share-Alike 4.0 International).