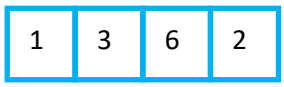
PWr Spotkanie 8
Aplikacje webowe na platformie .NET
Laboratorium – **Lista 8**

Wstęp.

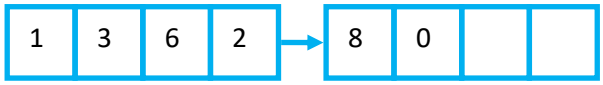
Wykorzystując gotowe klasy generyczne, które zostały zaimplementowane efektywnie, można równie efektywnie zaimplementować własne klasy generyczne. Jedna z takich klas może być wykorzystana do pamiętania danych w dużym pliku podzielonym na bloki odpowiadające jednemu klastrowi i operacji odczytu/zapisu na dysku. Pewnym wstępem do implementacji takiej klasy jest stworzenie klasy do pamiętania danych w postaci listy tablic o określonym stałym rozmiarze. Gdy podczas operacji dodawania pierwsza tablica zostanie wypełniona, przy dodawaniu nowego elementu, na końcu listy tworzona jest nowa tablica (o tym samym rozmiarze) do wykorzystania. Podczas usuwania elementu może się okazać, że na końcu listy występują niewykorzystane tablice. Z założenia nie są one automatyczne usuwane z listy tablic, ale może istnieć operacja odcinająca je na żądanie. Jeśli nazwiemy taką klasę `ListOfArrayList`, to przykładowe działanie może wyglądać jak na poniższych diagramach:

```
var col = new ListOfArrayList<int>(4);
```

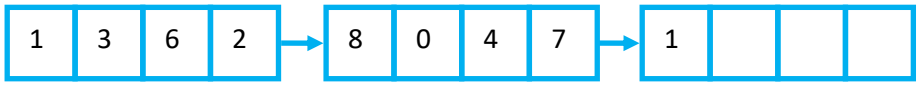
```
col.Add(1);
col.Add(3);
col.Add(6);
col.Add(2);
```



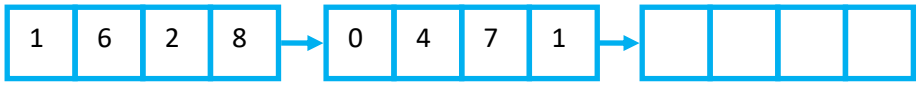

```
col.Add(8);
col.Add(0);
```



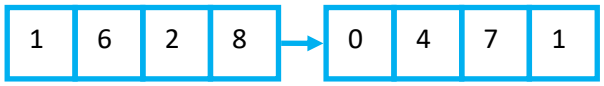

```
col.Add(4);
col.Add(7);
col.Add(1);
```



```
col.Remove(3);
```




```
col.Trim();
```



Lista zadań

Przygotować program demonstrujący użycie zaimplementowanych rozwiązań dla różnych przypadków użycia, dla równych typów (`int`, `string`, `Student`, `MixedNumber`).
Wykorzystać kolekcje i operacje na nich obecne w standardowej bibliotece C#.

1. Napisać klasę `ListOfArrayList`, która jest generyczną kolekcją zaimplementowaną wewnętrznie jako lista tablic określonej maksymalnej długości. Klasa ma implementować interfejs `IEnumerable<T>` oraz `IList<T>` (ale ten drugi nie wszystko). Wewnętrznie można użyć gotowych klas generycznych typu `List<T>` i `ArrayList<T>`. Z interfejsu `IList<T>` zaimplementować co najmniej metody `Count`, `Add`, `Clear`, `Contains`, `IndexOf`, `Remove`, `RemoveAt`. Dodatkowo niech będzie operacja Trim(), która odetnie te końcowe tablice, które puste (ale nie ma przycinać pozostałych tablicy.
2. Stworzyć operator dodawania do klasy `ListOfArrayList<T>` elementów z innej kolekcji implementującej interfejs `IEnumerable<T>`.

Data I: Spotkanie 9 (max 100 punktów)
Data II: Spotkanie 10 (max 80 punktów)
Data III: Spotkanie 11 (max 50 punktów)