Collection og Generics (stort set som i Java)

1) Collections
2) Generics

IEnumerable

- IEnumerable
  - Interface svarende til Iterable i Java.
  - har metoden GetEnumerator(),
  - som returnerer et IEnumerator-objekt

IEnumerator

IEnumerator

- Interface svarende til Iterator i Java
- har propertien
  - Current
- og metoderne
  - MoveNext();
  - Reset();

ICollection

Syntax:

```
public interface ICollection : IEnumerable {
   int Count { get; }
   IEnumerator GetEnumerator() ;
   void CopyTo(Array array; int index);
   …
}
```

Eksempel

Eksempel (List implementerer IEnumerable):

```
List<int> list = new List<int>();
list.Add(7);
list.Add(9);
list.Add(13);

int sum = 0;
IEnumerator<int> iter = list.GetEnumerator();
while (iter.MoveNext())
{
    int n = iter.Current;
    sum += n;
}  // man kan også bruge en foreach loop i stedet for.
```

System.Collections

- ArrayList
- BitArray
- List (generic)
- Hashtable
- Directory
- Queue
- SortedList
- Stack

Brug af en generic class

- Klassen er initialiseret til at bruge en bestemt type (og subklasser af denne type) under erklæringen:

  Stack<int>

Stack

Eksempel:

```
Stack<int> stak = new Stack<int>();

stak.Push(1);
stak.Push(2);
stak.Push(3);
e=stak.Peek(); // aflæser øverste uden at fjerne
Console.WriteLine("Efter Peek, Count: {0}",
                  stak.Count);
e=stak.Pop(); //aflæser og fjerner øverste element
```

Queue

Eksempel:

```
Queue<int> q = new Queue<int>();

q.Enqueue(10);
Console.WriteLine(”Count: {0}", q.Count);

f = q.First();
l = q.Last();
n = q.Dequeue(); // kaster exception ved tom kø
```

Konstruktion af en generic class

```
public Stack<T>
   {
        Push(T object) { … }
        T Pop() { … }
   }
….
Stack<String> s;
s.Push(8);  //compile-time type error !!!
```

Sub-classing generics

```
public CarStack : Stack<Car>
{  … }

Public SortedStack<T> : Stack<T>
{  … }
```