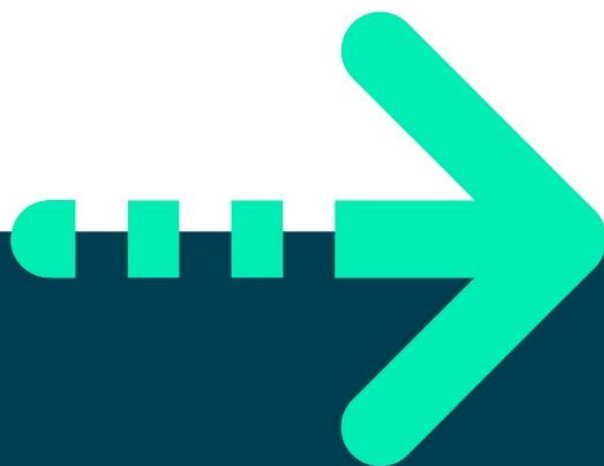




# **EXERCISE 7, GENERIC INTERFACES**





## Exercise 7, Generic interfaces

### Objective

To be able to use and understand the role of generic interfaces.

### Overview

Initially you will be working in a project which declares, creates and manipulates a list of 'person' references. You will use generic collection class **ArrayList<E>** and a generic interface **Comparable<E>** to sort the entries in the List.

In Part two you will take your work to date and enhance it using another interface **Comparer<E>**, which will enable you to sort your List in an alternative sequence.

### Part 1 – Using a Generic class and Comparer<E>

#### Step by step

#### Creating, filling and printing a list of person references

1. Create a new Console application called **Lab07**
2. Add a new class called **Person**
  - a. Add a following fields:  

```
string name;  
int age;
```
  - b. Create a constructor to set the name and age of a person
3. Override the **ToString()** method that prints out a padded **name** and **age** tab-separated.
4. Open the **Main()** method and create a **List<Person>** called **people**.
5. Add 6 Persons to the **people** ArrayList given the following ages and names:  

```
int[] ages = { 36, 53, 51, 21, 41, 19 };  
  
string[] names = {  
    "Holland", "Turner", "Powell", "Vicious", "Lyndon", "Spears"  
};
```

Tip: use a normal for loop using index of 0 to **people.Count**

#### Implementing IComparable<Person>

6. If you invoke **people.sort()** method in the **Main()** method, you'll get the following exception:



**System.InvalidOperationException:** 'Failed to compare two elements in the array.'

You get the above error because the system doesn't know how to compare one element of the people with another. To tell the system how to compare two elements, Person class has to implement the **IComparable<Person>**

7. Open the Person class and implement the **IComparable<Person>** to sort people by age.

**Tip:** Please see the slides if you're not sure how to do this.

8. Back in the Main() method, type *people.Sort();* to sort people by age.
9. Display every person's age and name in the *people* list.

**Tip:** better create a method to display people.

## Sorting by alternative criteria

### Part 2 – Sorting the list – Implementing **IComparer<Person>**

Now it's time to sort the **List<Person>** by Name without changing its code.

1. Let's create another class that enables us to sort people by name.
  - a. Add a class called **PersonNameComparer**
  - b. This class must implement **IComparer<Person>**

**Tip:** use the String's **CompareTo()** method to compare names.

2. Back in the *Main()* method, call the *people.sort()* method  
**people.sort(new PersonNameComparer());**
3. Display people to make sure names are sorted.

**\*\* End \*\***

