Assignment 4

Tianle Yuan

04/21/2023

04 Generics

Test your knowledge

1. **Describe the problem generics address.**

A: Generics allow you to use class/methods that should work with any data type.

2. **How would you create a list of strings, using the generic List class?**

A: List<string> stringList = new List<string>().

3. **How many generic type parameters does the Dictionary class have?**

A: 2. TKey and TValue.

4. **True/False. When a generic class has multiple type parameters, they must all match.**

A: False.

5. **What method is used to add items to a List object?**

A: Add(T item).

6. **Name two methods that cause items to be removed from a List.**

A: Remove(T item) and RemoveAt(int index).

7. **How do you indicate that a class has a generic type parameter?**

A: With <T> after the class name.

8. **True/False. Generic classes can only have one generic type parameter.**

A: False.

9. **True/False. Generic type constraints limit what can be used for the generic type.**

A: True.

10. **True/False. Constraints let you use the methods of the thing you are constraining to.**

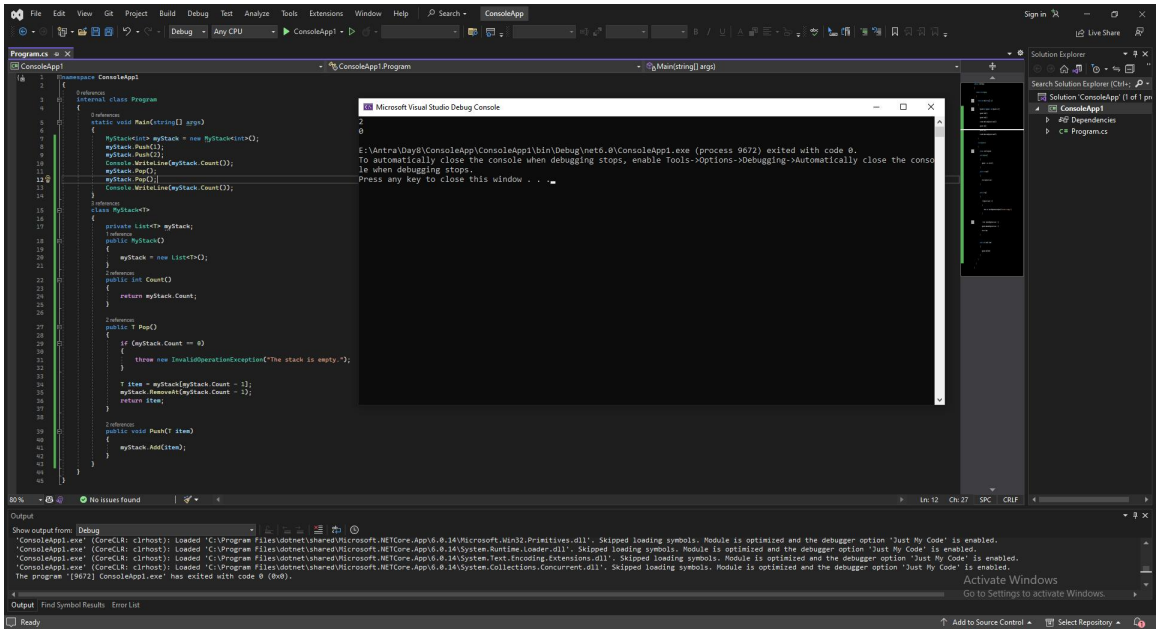
A: True

Practice working with Generics

1. Create a custom Stack class MyStack<T> that can be used with any data type which has following methods

1. **int Count()**
2. **T Pop()**
3. **Void Push()**

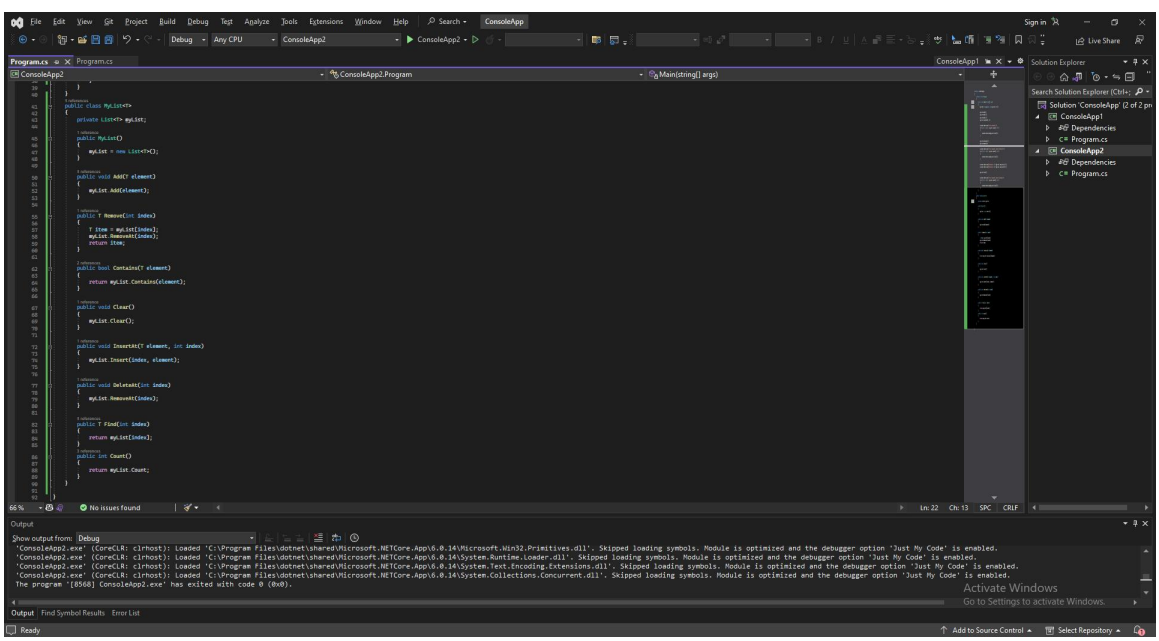
A: Here below is my code



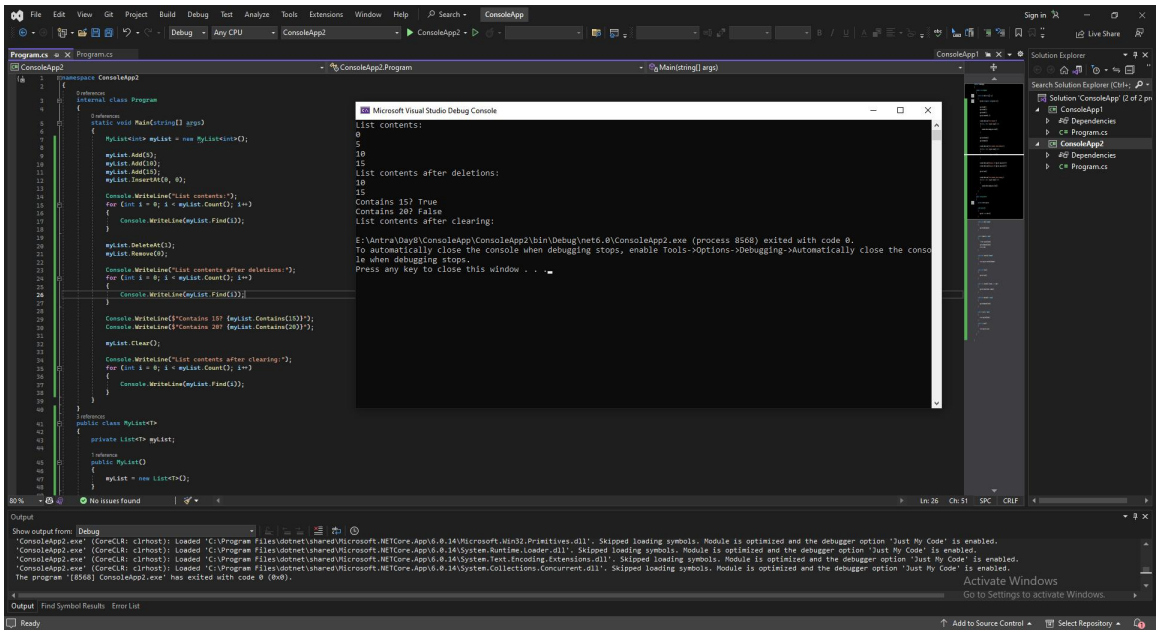
2. Create a Generic List data structure MyList<T> that can store any data type. Implement the following methods.

1. **void Add (T element)**
2. **T Remove (int index)**
3. **bool Contains (T element)**
4. **void Clear ()**
5. **void InsertAt (T element, int index)**
6. **void DeleteAt (int index)**
7. **T Find (int index)**

A: Here below is the data structure I defined:



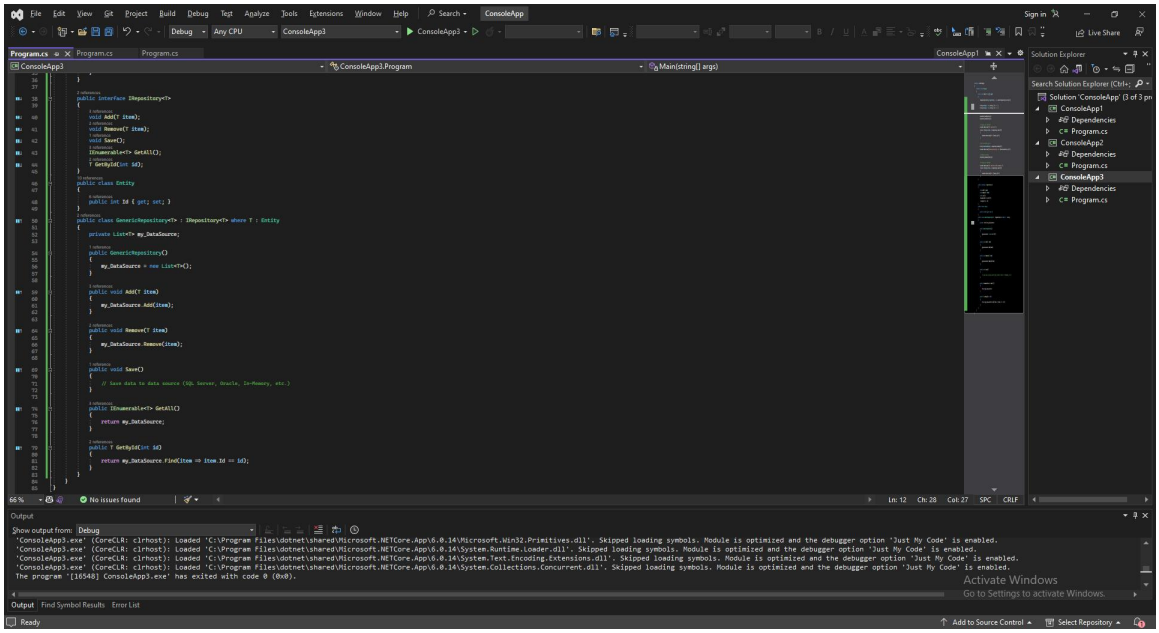
Here below is the code of usage of MyList<T>:



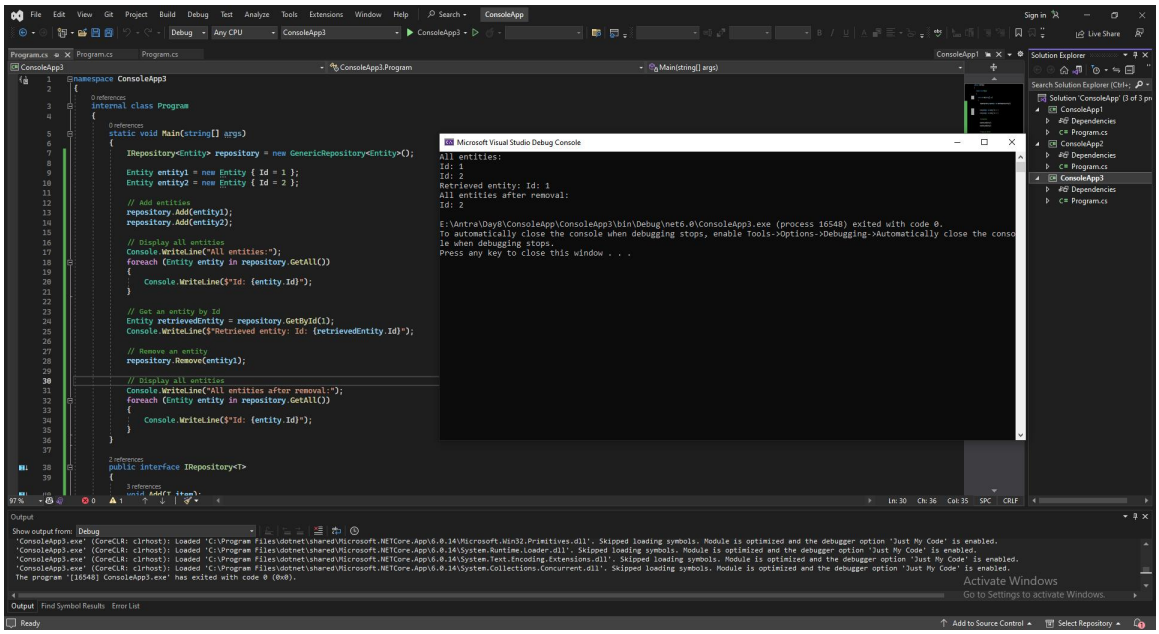
3. Implement a GenericRepository<T> class that implements IRepository<T> interface that will have common /CRUD/ operations so that it can work with any data source such as SQL Server, Oracle, In-Memory Data etc. Make sure you have a type constraint on T were it should be of reference type and can be of type Entity which has one property called Id. IRepository<T> should have following methods

1. **void Add(T item)**
2. **void Remove(T item)**
3. **Void Save()**
4. **IEnumerable<T> GetAll()**
5. **T GetById(int id)**

A: Here below is the data structure I defined:



Here below is the code of usage of GenericRepository<T>:



Explore following topics

- **Generics in .NET**
- **Generic classes and methods**
- **Collections and Data Structures**
- **Commonly Used Collection Types**