Test your Knowledge  
1. Describe the problem generics address.

Need to find a proper way to elegantly generalize and reuse codes that are specified by a special type.

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

var lists = new List<string>();

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

Two parameters. Key and Value.

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

No. Example:

var dic=new Dictionary<string, int>();

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

list.Add();

Using index is also an option.

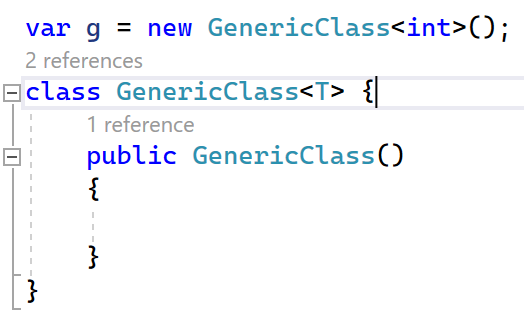
list[0] = 1;

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

list.Remove(); // remove the first occurrence of the given input param.

list.RemoveAt(); // remove the element in the given index

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



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

False. Dictionary has two generic type parameter.

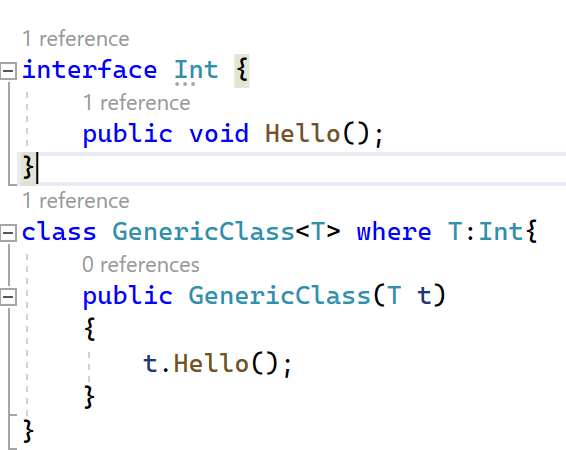
9. True/False. Generic type constraints limit what can be used for the generic type.

True.

Only the class that fulfill the requirement of where keyword can be used.

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

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()

https://github.com/wangziyannb/CSharpAssignments/blob/master/Assignment04\_Generics/MyStack.cs

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)

https://github.com/wangziyannb/CSharpAssignments/blob/master/Assignment04\_Generics/MyList.cs

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)

https://github.com/wangziyannb/CSharpAssignments/tree/master/Assignment04\_Generics/GenericRepository

Test Case for MyStack and MyList

