(T23)比較 Stack、Queue
CourseGUID: 29f1196a-1950-41a4-b9c1-dd13a9e92d92
==
(T23)比較 Stack、Queue
==
0. Summary
----
1. New Project
1.1. Create New Project : Sample
----
2. Sample : Program.cs
==

0. Summary

1.
Stack is First In Last out
Reference:
https://msdn.microsoft.com/en-us/library/3278tedw(v=vs.110).aspx
1.1.
//Stack.Push(Object obj)
Inserts an object at the top of the Stack.
1.2.
//Stack.Peek()
Returns the object at the top of the Stack without removing it.
1.3.
//Enumerable.FirstOrDefault<TSource>(IEnumerable<TSource> source)
//Enumerable.FirstOrDefault<TSource>(IEnumerable<TSource> source, Func<TSource, Boolean> predicate)
Returns the first element of the sequence that satisfies a condition or a default value if no such element is found.
1.4.
//Stack<T>.Pop()
Removes and returns the object at the top of the Stack<T>.
1.5.
//Stack<T>.Count
Gets the number of elements contained in the Stack<T>.

2.
Queue is First in First out.
Reference:
https://msdn.microsoft.com/en-us/library/7977ey2c(v=vs.110).aspx
2.1.
//Queue<T>.Enqueue(T item)
Adds an object to the end of the Queue<T>.
2.2.
//Queue<T>.Peek()
Returns the object at the beginning of the Queue<T> without removing it.
2.3.
//Enumerable.FirstOrDefault<TSource>(IEnumerable<TSource> source)
//Enumerable.FirstOrDefault<TSource>(IEnumerable<TSource> source, Func<TSource, Boolean> predicate)
Returns the first element of the sequence that satisfies a condition or a default value if no such element is found.

2.4.
//Queue<T>.Dequeue()
Removes and returns the object at the beginning of the Queue<T>.
2.5.
//Queue<T>.Count
Gets the number of elements contained in the Queue<T>.

==

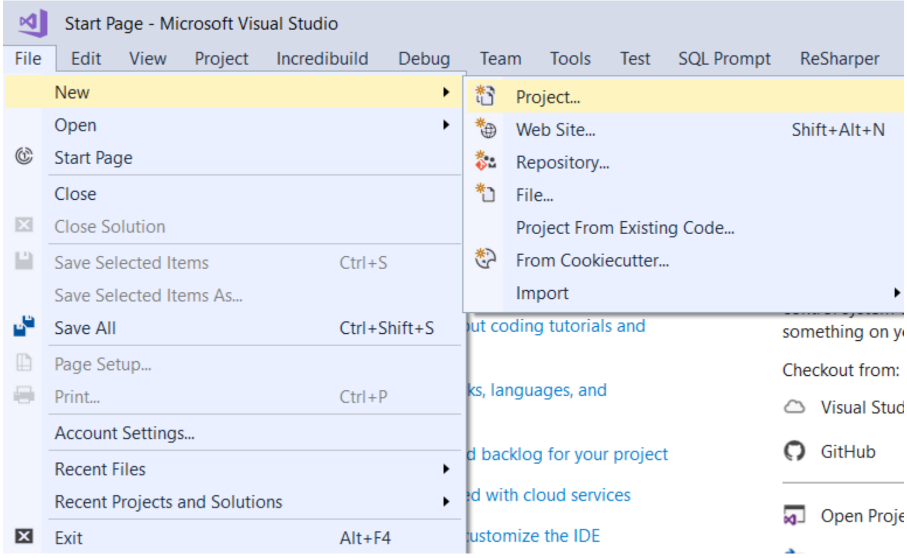
1. New Project

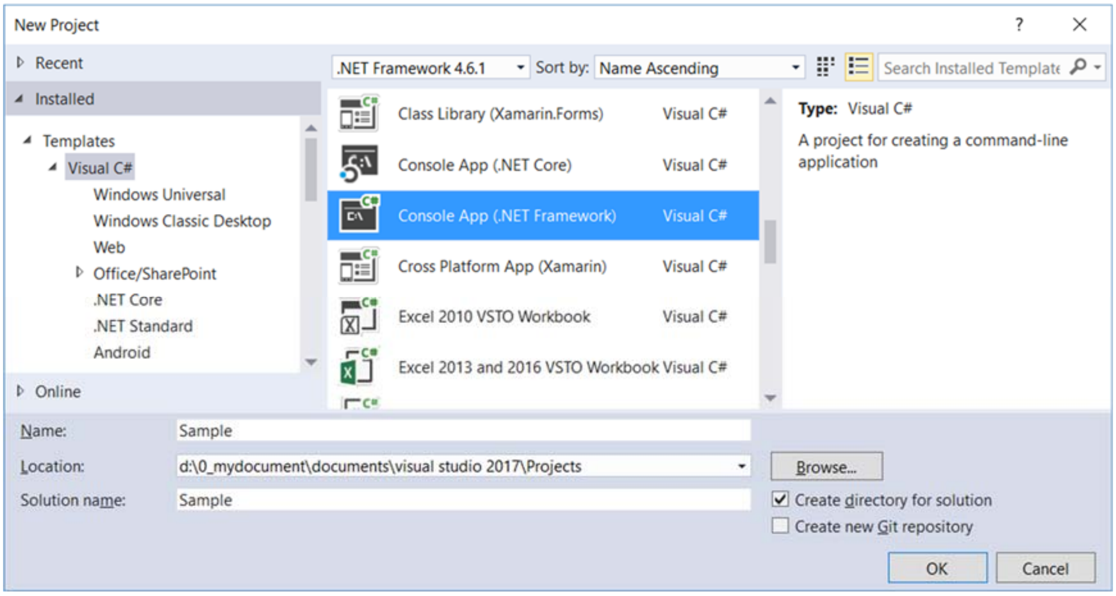
1.1. Create New Project : Sample

File --> New --> Project... -->

Visual C# --> **Console App (.Net Framework)** -->

Name: **Sample**




New Project
?
Recent
Installed
Templates
Visual C#
Windows Universal
Windows Classic Desktop
Web
Office/SharePoint
.NET Core
.NET Standard
Android
Online
Sort by:
Search Installed Template
Class Library (Xamarin.Forms) Visual C#
Console App (.NET Core) Visual C#
Console App (.NET Framework) Visual C#
Cross Platform App (Xamarin) Visual C#
Excel 2010 VSTO Workbook Visual C#
Excel 2013 and 2016 VSTO Workbook Visual C#
Browse...
OK
Cancel

.NET Framework 4.6.1 Sort by: Name Ascending

Type: Visual C#

A project for creating a command-line application

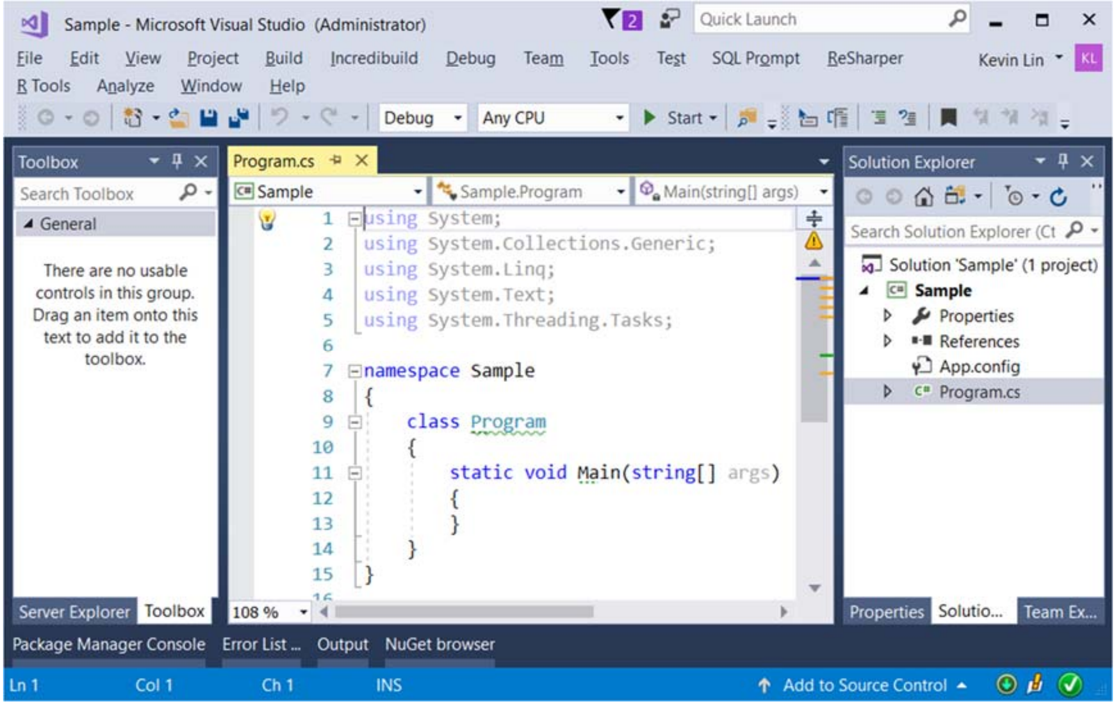
Name: Sample

Location: d:\0_mydocument\documents\visual studio 2017\Projects

Solution name: Sample

☑ Create directory for solution

□ Create new Git repository



There are no usable controls in this group. Drag an item onto this text to add it to the toolbox.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Sample
{
    class Program
    {
        static void Main(string[] args)
        {
        }
    }
}
```

==

2. Sample : Program.cs

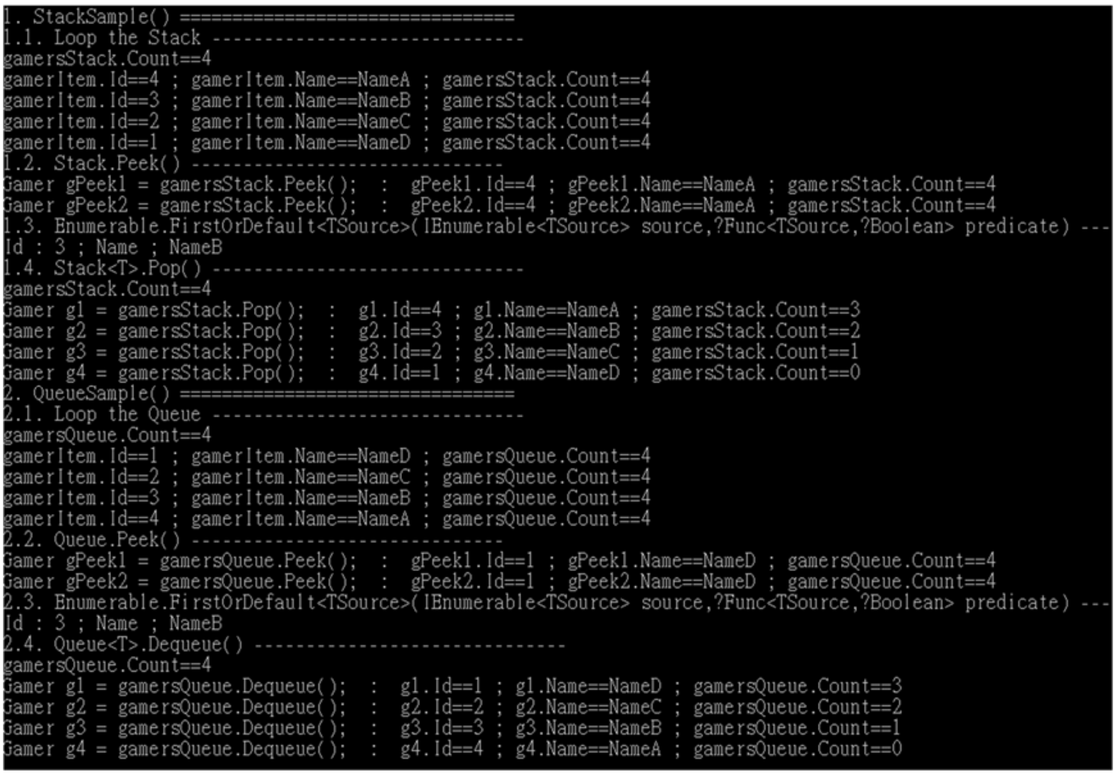
```
using System;
using System.Collections.Generic;
using System.Linq;
using OnLineGame;
namespace Sample
{
    class Program
     {
        static void Main(string[] args)
         {
            // 1. ====================================================
            Console.WriteLine("1. StackSample() ================================");
             StackSample();
            // 2. ====================================================
            Console.WriteLine("2. QueueSample() ================================");
             QueueSample();
            Console.ReadLine();
         }
        // 1. ====================================================
        static void StackSample()
         {
            // 1.0. --------------------------------------------------
            // Create a Stack which is First In Last out.
            Stack<Gamer> gamersStack = new Stack<Gamer>();
            //Stack.Push(Object obj)
            //Inserts an object at the top of the Stack.
             gamersStack.Push(new Gamer { Id = 1, Name = "NameD" });
            gamersStack.Push(new Gamer { Id = 2, Name = "NameC" });
            gamersStack.Push(new Gamer { Id = 3, Name = "NameB" });
            gamersStack.Push(new Gamer { Id = 4, Name = "NameA" });

            // 1.1. ----------------------------------------------------------
            Console.WriteLine("1.1. Loop the Stack ------------------------------");
            Console.WriteLine($"gamersStack.Count=={gamersStack.Count}");
            foreach (Gamer gamerItem in gamersStack)
            {
                Console.WriteLine($"gamerItem.Id=={gamerItem.Id} ; gamerItem.Name=={gamerItem.Name} ;
gamersStack.Count=={gamersStack.Count}");
            }
            //1.1.Loop the Stack ------------------------------
            //gamersStack.Count == 4
            //gamerItem.Id == 4; gamerItem.Name == NameA; gamersStack.Count == 4
            //gamerItem.Id == 3; gamerItem.Name == NameB; gamersStack.Count == 4
            //gamerItem.Id == 2; gamerItem.Name == NameC; gamersStack.Count == 4
            //gamerItem.Id == 1; gamerItem.Name == NameD; gamersStack.Count == 4
            // 1.2. ----------------------------------------------------------
            //Stack.Peek()
            //Returns the object at the top of the Stack without removing it.
            Console.WriteLine("1.2. Stack.Peek() ------------------------------");
            Gamer gPeek1 = gamersStack.Peek();
            Console.WriteLine($"Gamer gPeek1 = gamersStack.Peek();  :  gPeek1.Id=={gPeek1.Id} ;
gPeek1.Name=={gPeek1.Name} ; gamersStack.Count=={gamersStack.Count}");
            Gamer gPeek2 = gamersStack.Peek();
            Console.WriteLine($"Gamer gPeek2 = gamersStack.Peek();  :  gPeek2.Id=={gPeek2.Id} ;
gPeek2.Name=={gPeek2.Name} ; gamersStack.Count=={gamersStack.Count}");
            //1.2. Stack.Peek() ------------------------------
            //Gamer gPeek1 = gamersStack.Peek();  :  gPeek1.Id == 4; gPeek1.Name == NameA;
gamersStack.Count == 4
            //Gamer gPeek2 = gamersStack.Peek();  :  gPeek2.Id == 4; gPeek2.Name == NameA;
gamersStack.Count == 4
            // 1.3. ----------------------------------------------------------
            //Enumerable.FirstOrDefault<TSource>(IEnumerable<TSource> source)
            //Enumerable.FirstOrDefault<TSource>(IEnumerable<TSource> source, Func<TSource, Boolean>
predicate)
            //Returns the first element of the sequence that satisfies a condition or a default value if
no such element is found.
            Console.WriteLine("1.3. Enumerable.FirstOrDefault<TSource>(IEnumerable<TSource> source,
Func<TSource, Boolean> predicate) ------------------------------");
            Gamer firstOrDefaultStackGamer = gamersStack.FirstOrDefault(g => g.Id == 3);
            //Console.WriteLine(firstOrDefaultStackGamer != null ?
            //    firstOrDefaultStackGamer.ToString() :
            //    "gamersStack.FirstOrDefault(g => g.Id==3) == NULL");
            Console.WriteLine(firstOrDefaultStackGamer?.ToString() ?? "gamersStack.FirstOrDefault(g =>
g.Id==3) == NULL");
            //1.3. Enumerable.FirstOrDefault<TSource>(IEnumerable<TSource> source,?Func<TSource,?Boolean>
predicate) ------------------------------
            //Id: 3; Name; NameB
            // 1.4. ----------------------------------------------------------
            //Stack<T>.Pop()
            //Removes and returns the object at the top of the Stack<T>.
            //Stack<T>.Count
            //Gets the number of elements contained in the Stack<T>.
            Console.WriteLine("1.4. Stack<T>.Pop() ------------------------------");
            Console.WriteLine($"gamersStack.Count=={gamersStack.Count}");
            Gamer g1 = gamersStack.Pop();
            Console.WriteLine($"Gamer g1 = gamersStack.Pop();  :  g1.Id=={g1.Id} ; g1.Name=={g1.Name} ; gamersStack.Count=={gamersStack.Count}");
            Gamer g2 = gamersStack.Pop();
            Console.WriteLine($"Gamer g2 = gamersStack.Pop();  :  g2.Id=={g2.Id} ; g2.Name=={g2.Name} ; gamersStack.Count=={gamersStack.Count}");
            Gamer g3 = gamersStack.Pop();
            Console.WriteLine($"Gamer g3 = gamersStack.Pop();  :  g3.Id=={g3.Id} ; g3.Name=={g3.Name} ; gamersStack.Count=={gamersStack.Count}");
            Gamer g4 = gamersStack.Pop();
            Console.WriteLine($"Gamer g4 = gamersStack.Pop();  :  g4.Id=={g4.Id} ; g4.Name=={g4.Name} ; gamersStack.Count=={gamersStack.Count}");
            //1.4. Stack<T>.Pop() ------------------------------
            //gamersStack.Count == 4
            //Gamer g1 = gamersStack.Pop();  :  g1.Id == 4; g1.Name == NameA; gamersStack.Count == 3
            //Gamer g2 = gamersStack.Pop();  :  g2.Id == 3; g2.Name == NameB; gamersStack.Count == 2
            //Gamer g3 = gamersStack.Pop();  :  g3.Id == 2; g3.Name == NameC; gamersStack.Count == 1
            //Gamer g4 = gamersStack.Pop();  :  g4.Id == 1; g4.Name == NameD; gamersStack.Count == 0
        }


       // 2. ====================================================
        static void QueueSample()
        {
            // 1.0. --------------------------------------------------
            // Create a Queue which is First In First out.
            Queue<Gamer> gamersQueue = new Queue<Gamer>();
            //Queue<T>.Enqueue(T item)
            //Adds an object to the end of the Queue<T>.
            gamersQueue.Enqueue(new Gamer { Id = 1, Name = "NameD" });
            gamersQueue.Enqueue(new Gamer { Id = 2, Name = "NameC" });
            gamersQueue.Enqueue(new Gamer { Id = 3, Name = "NameB" });
            gamersQueue.Enqueue(new Gamer { Id = 4, Name = "NameA" });
            // 1.1. ------------------------------------------------------------
            Console.WriteLine("1.1. Loop the Queue ------------------------------");
            Console.WriteLine($"gamersQueue.Count=={gamersQueue.Count}");
            foreach (Gamer gamerItem in gamersQueue)
            {
                Console.WriteLine($"gamerItem.Id=={gamerItem.Id} ; gamerItem.Name=={gamerItem.Name} ; gamersQueue.Count=={gamersQueue.Count}");
            }
            //1.1.Loop the Queue ------------------------------
            //gamersQueue.Count == 4
            //gamerItem.Id == 1; gamerItem.Name == NameD; gamersQueue.Count == 4
            //gamerItem.Id == 2; gamerItem.Name == NameC; gamersQueue.Count == 4
            //gamerItem.Id == 3; gamerItem.Name == NameB; gamersQueue.Count == 4
            //gamerItem.Id == 4; gamerItem.Name == NameA; gamersQueue.Count == 4
            // 1.2. ------------------------------------------------------------
            //Queue<T>.Peek()
            //Returns the object at the beginning of the Queue<T> without removing it.
            Console.WriteLine("1.2. Queue.Peek() ------------------------------");
            Gamer gPeek1 = gamersQueue.Peek();
            Console.WriteLine($"Gamer gPeek1 = gamersQueue.Peek();  :  gPeek1.Id=={gPeek1.Id} ;
gPeek1.Name=={gPeek1.Name} ; gamersQueue.Count=={gamersQueue.Count}");
            Gamer gPeek2 = gamersQueue.Peek();
            Console.WriteLine($"Gamer gPeek2 = gamersQueue.Peek();  :  gPeek2.Id=={gPeek2.Id} ;
gPeek2.Name=={gPeek2.Name} ; gamersQueue.Count=={gamersQueue.Count}");
            //1.2. Queue.Peek() ------------------------------
            //Gamer gPeek1 = gamersQueue.Peek();  :  gPeek1.Id == 1; gPeek1.Name == NameD;
gamersQueue.Count == 4
            //Gamer gPeek2 = gamersQueue.Peek();  :  gPeek2.Id == 1; gPeek2.Name == NameD;
gamersQueue.Count == 4
            // 1.3. ------------------------------------------------------------
            //Enumerable.FirstOrDefault<TSource>(IEnumerable<TSource> source)
            //Enumerable.FirstOrDefault<TSource>(IEnumerable<TSource> source, Func<TSource, Boolean>
predicate)
            //Returns the first element of the sequence that satisfies a condition or a default value if
no such element is found.
            Console.WriteLine("1.3. Enumerable.FirstOrDefault<TSource>(IEnumerable<TSource> source,
Func<TSource, Boolean> predicate) ------------------------------");
            Gamer firstOrDefaultQueueGamer = gamersQueue.FirstOrDefault(g => g.Id == 3);
            //Console.WriteLine(firstOrDefaultQueueGamer != null ?
            //    firstOrDefaultQueueGamer.ToString() :
            //    "gamersQueue.FirstOrDefault(g => g.Id==3) == NULL");
            Console.WriteLine(firstOrDefaultQueueGamer?.ToString() ?? "gamersQueue.FirstOrDefault(g =>
g.Id==3) == NULL");
            //1.3. Enumerable.FirstOrDefault<TSource>(IEnumerable<TSource> source,?Func<TSource,?Boolean>
predicate) ------------------------------
            //Id: 3; Name; NameB
            // 1.4. ------------------------------------------------------------
            //Queue<T>.Dequeue()
            //Removes and returns the object at the beginning of the Queue<T>.
            //Queue<T>.Count
            //Gets the number of elements contained in the Queue<T>.
            Console.WriteLine("1.4. Queue<T>.Dequeue() ------------------------------");
            Console.WriteLine($"gamersQueue.Count=={gamersQueue.Count}");
            Gamer g1 = gamersQueue.Dequeue();
            Console.WriteLine($"Gamer g1 = gamersQueue.Dequeue();  :  g1.Id=={g1.Id} ; g1.Name=={g1.Name} ;
gamersQueue.Count=={gamersQueue.Count}");
            Gamer g2 = gamersQueue.Dequeue();
            Console.WriteLine($"Gamer g2 = gamersQueue.Dequeue();  :  g2.Id=={g2.Id} ; g2.Name=={g2.Name} ;
gamersQueue.Count=={gamersQueue.Count}");
            Gamer g3 = gamersQueue.Dequeue();
            Console.WriteLine($"Gamer g3 = gamersQueue.Dequeue();  :  g3.Id=={g3.Id} ; g3.Name=={g3.Name} ;
gamersQueue.Count=={gamersQueue.Count}");
            Gamer g4 = gamersQueue.Dequeue();
            Console.WriteLine($"Gamer g4 = gamersQueue.Dequeue();  :  g4.Id=={g4.Id} ; g4.Name=={g4.Name} ;
gamersQueue.Count=={gamersQueue.Count}");
            //1.4. Queue<T>.Dequeue() ------------------------------
            //gamersQueue.Count == 4
            //Gamer g1 = gamersQueue.Dequeue();  :  g1.Id == 1; g1.Name == NameD; gamersQueue.Count == 3
            //Gamer g2 = gamersQueue.Dequeue();  :  g2.Id == 2; g2.Name == NameC; gamersQueue.Count == 2
            //Gamer g3 = gamersQueue.Dequeue();  :  g3.Id == 3; g3.Name == NameB; gamersQueue.Count == 1
            //Gamer g4 = gamersQueue.Dequeue();  :  g4.Id == 4; g4.Name == NameA; gamersQueue.Count == 0s
        }
    }
}
namespace OnLineGame
```

```
{
    public class Gamer
    {
        public int Id { get; set; }
        public string Name { get; set; }
        public override string ToString()
        {
            return $"Id : {Id} ; Name ; {Name}";
        }
    }
}

/*
1.
Stack is First In Last out
Reference:
https://msdn.microsoft.com/en-us/library/3278tedw(v=vs.110).aspx
1.1.
//Stack.Push(Object obj)
Inserts an object at the top of the Stack.
1.2.
//Stack.Peek()
Returns the object at the top of the Stack without removing it.
1.3.
//Enumerable.FirstOrDefault<TSource>(IEnumerable<TSource> source)
//Enumerable.FirstOrDefault<TSource>(IEnumerable<TSource> source, Func<TSource, Boolean> predicate)
Returns the first element of the sequence that satisfies a condition or a default value if no such
element is found.
1.4.
//Stack<T>.Pop()
Removes and returns the object at the top of the Stack<T>.
1.5.
//Stack<T>.Count
Gets the number of elements contained in the Stack<T>.
------------------------------------------------------
2.
Queue is First in First out.
Reference:
https://msdn.microsoft.com/en-us/library/7977ey2c(v=vs.110).aspx
2.1.
//Queue<T>.Enqueue(T item)
Adds an object to the end of the Queue<T>.
2.2.
//Queue<T>.Peek()
Returns the object at the beginning of the Queue<T> without removing it.
2.3.
//Enumerable.FirstOrDefault<TSource>(IEnumerable<TSource> source)
//Enumerable.FirstOrDefault<TSource>(IEnumerable<TSource> source, Func<TSource, Boolean> predicate)
Returns the first element of the sequence that satisfies a condition or a default value if no such
element is found.
2.4.
//Queue<T>.Dequeue()
Removes and returns the object at the beginning of the Queue<T>.
2.5.
//Queue<T>.Count
Gets the number of elements contained in the Queue<T>.
*/
```



```
1. StackSample() ================================
1.1. Loop the Stack ------------------------------
gamersStack.Count==4
gamerItem.Id==4 ; gamerItem.Name==NameA ; gamersStack.Count==4
gamerItem.Id==3 ; gamerItem.Name==NameB ; gamersStack.Count==4
gamerItem.Id==2 ; gamerItem.Name==NameC ; gamersStack.Count==4
gamerItem.Id==1 ; gamerItem.Name==NameD ; gamersStack.Count==4
1.2. Stack.Peek() ------------------------------
Gamer gPeek1 = gamersStack.Peek();  :  gPeek1.Id==4 ; gPeek1.Name==NameA ; gamersStack.Count==4
Gamer gPeek2 = gamersStack.Peek();  :  gPeek2.Id==4 ; gPeek2.Name==NameA ; gamersStack.Count==4
1.3. Enumerable.FirstOrDefault<TSource>(IEnumerable<TSource> source,?Func<TSource,?Boolean> predicate) ---
Id : 3 ; Name ; NameB
1.4. Stack<T>.Pop() ------------------------------
gamersStack.Count==4
Gamer g1 = gamersStack.Pop();  :  g1.Id==4 ; g1.Name==NameA ; gamersStack.Count==3
Gamer g2 = gamersStack.Pop();  :  g2.Id==3 ; g2.Name==NameB ; gamersStack.Count==2
Gamer g3 = gamersStack.Pop();  :  g3.Id==2 ; g3.Name==NameC ; gamersStack.Count==1
Gamer g4 = gamersStack.Pop();  :  g4.Id==1 ; g4.Name==NameD ; gamersStack.Count==0
2. QueueSample() ================================
2.1. Loop the Queue ------------------------------
gamersQueue.Count==4
gamerItem.Id==1 ; gamerItem.Name==NameD ; gamersQueue.Count==4
gamerItem.Id==2 ; gamerItem.Name==NameC ; gamersQueue.Count==4
gamerItem.Id==3 ; gamerItem.Name==NameB ; gamersQueue.Count==4
gamerItem.Id==4 ; gamerItem.Name==NameA ; gamersQueue.Count==4
2.2. Queue.Peek() ------------------------------
Gamer gPeek1 = gamersQueue.Peek();  :  gPeek1.Id==1 ; gPeek1.Name==NameD ; gamersQueue.Count==4
Gamer gPeek2 = gamersQueue.Peek();  :  gPeek2.Id==1 ; gPeek2.Name==NameD ; gamersQueue.Count==4
2.3. Enumerable.FirstOrDefault<TSource>(IEnumerable<TSource> source,?Func<TSource,?Boolean> predicate) ---
Id : 3 ; Name ; NameB
2.4. Queue<T>.Dequeue() ------------------------------
gamersQueue.Count==4
Gamer g1 = gamersQueue.Dequeue();  :  g1.Id==1 ; g1.Name==NameD ; gamersQueue.Count==3
Gamer g2 = gamersQueue.Dequeue();  :  g2.Id==2 ; g2.Name==NameC ; gamersQueue.Count==2
Gamer g3 = gamersQueue.Dequeue();  :  g3.Id==3 ; g3.Name==NameB ; gamersQueue.Count==1
Gamer g4 = gamersQueue.Dequeue();  :  g4.Id==4 ; g4.Name==NameA ; gamersQueue.Count==0
```