C#

Functional

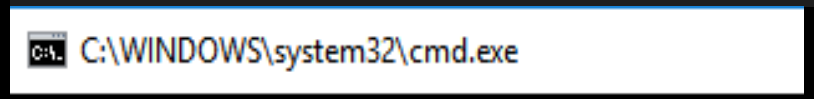
Delegate

```
delegate void OuputString(string str);

static void Main(string[] args)
{
    var outputFunc = new OuputString(StrToConsole);

    var str = "E boy";
    outputFunc(str);
}

private static void StrToConsole(string str)
{
    Console.WriteLine(str);
}
```




```
E boy
Press any key to continue . . .
```

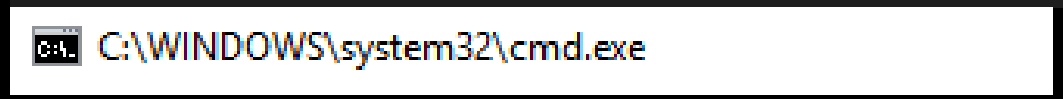
Delegate

```
delegate void OuputString(string str);

static void Main(string[] args)
{
    var outputFunc = new OuputString(LowercaseToConsole);

    var str = "E boy";
    outputFunc(str);
}

private static void LowercaseToConsole(string str)
{
    Console.WriteLine(str.ToLower());
}
```




```
e boy
Press any key to continue . . .
```

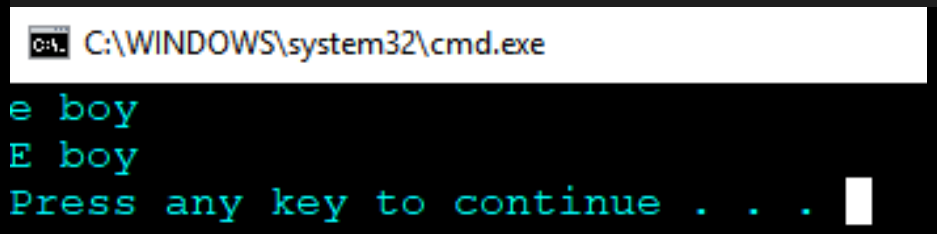
Delegate

```
delegate void OuputString(string str);

static void Main(string[] args)
{
    OuputString outputFunc = null;

    outputFunc += LowercaseToConsole;
    outputFunc += StrToConsole;

    var str = "E boy";
    outputFunc(str);
}
```



Delegate

```
delegate void OuputString(string str);

static void Main(string[] args)
{
    OuputString outputFunc = null;

    outputFunc += LowercaseToConsole;
    outputFunc += StrToConsole;
    outputFunc += LowercaseToConsole;
    outputFunc += StrToConsole;
    outputFunc -= LowercaseToConsole;

    var str = "E boy";
    outputFunc(str);
}
```

Delegate

```
delegate void OuputString(string str);

static void Main(string[] args)
{
    OuputString outputFunc = null;

    outputFunc += LowercaseToConsole;
    outputFunc += StrToConsole;
    outputFunc += LowercaseToConsole;
    outputFunc += StrToConsole;
    outputFunc -= LowercaseToConsole;

    var str = "E boy";
    outputFunc(str);
}
```

```
C:\WINDOWS\system32\cmd.exe
e boy
E boy
E boy
Press any key to continue . . .
```

Delegate

```
static void Main(string[] args)
{
    Action<string> outputFunc = LowercaseToConsole;

    outputFunc += LowercaseToConsole;
    outputFunc += StrToConsole;

    var str = "E boy";
    outputFunc(str);
}
```

Delegate

```
static void Main(string[] args)
{
    Func<string, int> outputFunc = LowercaseToConsole;

    var str = "E boy";
    var length = outputFunc(str);
    Console.WriteLine($"{length} caharacters was changed");
}

private static int LowercaseToConsole(string str)
{
    Console.WriteLine(str.ToLower());
    return str.Count(char.IsUpper);
}
```


```
e boy
1 caharacters was changed
Press any key to continue . . .
```

Delegate

```
static void Main(string[] args)
{
    Func<string, int> outputFunc = x =>
    {
        Console.WriteLine(x.ToLower());
        return x.Count(char.IsUpper);
    };

    var str = "E boy";
    var length = outputFunc(str);
    Console.WriteLine($"{length} caharacters was changed");
}
```

Delegate

```
var audioSource = Console.ReadLine();
string[] audios;
switch (audioSource)
{
    case "vk":
        audios = DownloadFromVk();
        Console.WriteLine(string.Join("\r\n", audios));
        while (audios.Length == 10)
        {
            audios = DownloadFromVk();
            Console.WriteLine(string.Join("\r\n", audios));
        }
        break;
    case "yandex":
        audios = DownloadFromYandex();
        Console.WriteLine(string.Join("\r\n", audios));
        while (audios.Length == 10)
        {
            audios = DownloadFromYandex();
            Console.WriteLine(string.Join("\r\n", audios));
        }
        break;
    default: throw new Exception("I dont know what u want.");
}
```

Delegate

```
var audioSource = Console.ReadLine();
string[] audios;
do
{
    switch (audioSource)
    {
        case "vk":
            audios = DownloadFromVk();
            Console.WriteLine(string.Join("\r\n", audios));
            break;
        case "yandex":
            audios = DownloadFromYandex();
            Console.WriteLine(string.Join("\r\n", audios));
            break;
        default: throw new Exception("I dont know what u want.");
    }
} while (audios.Length == 10);
```

Delegate

```
var audioSource = Console.ReadLine();

Func<string[]> getAudioFunc;

switch (audioSource)
{
    case "vk": getAudioFunc = DownloadFromVk; break;
    case "yandex":getAudioFunc = DownloadFromYandex;break;
    default: throw new Exception("I dont know what u want.");
}

string[] audios;
do
{
    audios = getAudioFunc();
    Console.WriteLine(string.Join("\r\n", audios));
} while (audios.Length == 10);
```

LINQ

```
namespace System.Collections
{
    public interface IEnumerable
    {
        IEnumerator GetEnumerator();
    }
}
```

```
public interface IEnumerator
{
    bool MoveNext();

    object Current { get; }

    void Reset();
}
```

LINQ

```
static void Main(string[] args)
{
    var str = "This is Collection of 35 characters.";
    Console.WriteLine(str.Count(c => char.IsUpper(c) && char.IsLetter(c)));
}
```

```
static void Main(string[] args)
{
    var str = "This is Collection of 35 characters.";
    Console.WriteLine(str.Count(c => char.IsDigit(c) && char.IsLower(c)));
}
```

LINQ

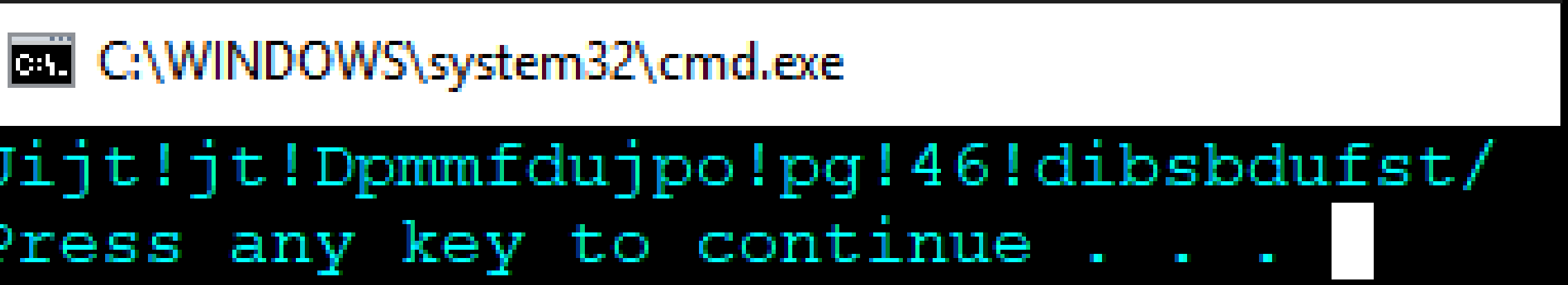
```
var str = "This is Collection of 35 characters.";
var digitsAndUpper = str.Where(c => char.IsDigit(c) || char.IsUpper(c)).ToArray();
Console.WriteLine(new string(digitsAndUpper));
```

```
var str = "This is Collection of 35 characters.";
var digitsAndUpper = str.Select(c => c + 1).Select(Convert.ToChar).ToArray();
Console.WriteLine(new string(digitsAndUpper));
```

LINQ

```
var str = "This is Collection of 35 characters.";
var digitsAndUpper = str.Where(c => char.IsDigit(c) || char.IsUpper(c)).ToArray();
Console.WriteLine(new string(digitsAndUpper));
```

```
var str = "This is Collection of 35 characters.";
var digitsAndUpper = str.Select(c => c + 1).Select(Convert.ToChar).ToArray();
Console.WriteLine(new string(digitsAndUpper));
```



LINQ

```
var str = "This is Collection of 35 characters.";
var charsCount = str
    .GroupBy(c => c)
    .Select(g =>
        (Character: g.Key,
            Count: g.Count())
    )
    .OrderByDescending(x => x.Count)
    .ThenBy(x => x.Character)
    .ToArray();

Console.WriteLine(string.Join("\r\n", charsCount));
```

LINQ

```
var str = "This is Collection of 35 characters.";
var charsCount = str
    .GroupBy(c => c)
    .Select(g =>
        (Character: g.Key,
            Count: g.Count())
    )
    .OrderByDescending(x => x.Count)
    .ThenBy(x => x.Character)
    .ToArray();

Console.WriteLine(string.Join("\r\n", charsCount));
```

```
C:\WIN
( , 5)
(c, 3)
(i, 3)
(o, 3)
(s, 3)
(a, 2)
(e, 2)
(h, 2)
(l, 2)
(r, 2)
(t, 2)
(., 1)
(3, 1)
(5, 1)
(C, 1)
(T, 1)
(f, 1)
(n, 1)
```