C# - 12

Collections

Dynamic Array

When I create a integer array So but default what data it will store?

```
class NosArray
{
    int[] nos = new int[5];
    9 references
```

------it will store 0, because it is a integer type and we have seen integer will be 0 type, it will always be initialized to 0

----if it is a reference type
if i write like this

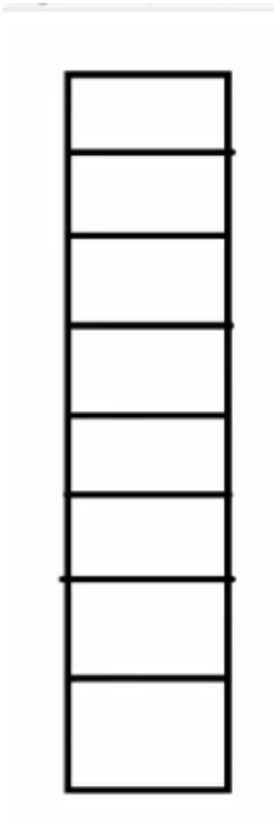
```
int[] nos;
```

then it will store null in it

but we are initializing it, as we are initializing it and its of integer type, so by default it will store integer value i.e 0

Now How do you get the index number ?

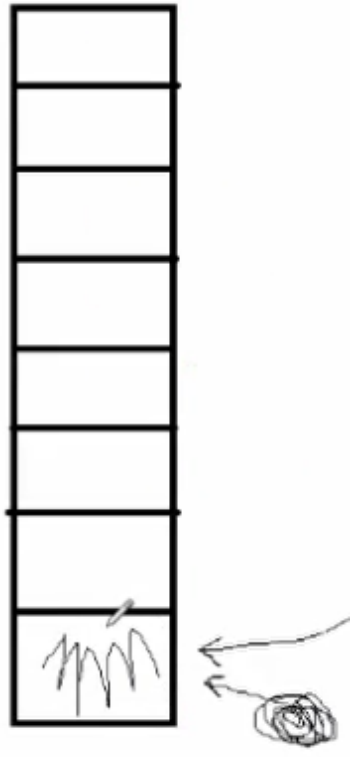
Imagine you are in mall and you are maintaining the stall where you take peoples bag and give them token



you will always know weather it is blank or it is filled
So in our case we always fill this and when someone comes and say empty this we'll empty this
Now how do you know weather it is empty or filled, From where did you start
So you will hold a particular pointer (Pointer is just a location), and search weather it is empty or not

So i need a pointer which say weather it is filled or not ,
we'll take a pointer cnt and initialize it to 0

if you are initializing it to 0 means you already have something in it



So you should initialize it to -1

```
int[] nos = new int[5];
int cnt = -1;
```

now as it is initialize to -1
we cannot directly say

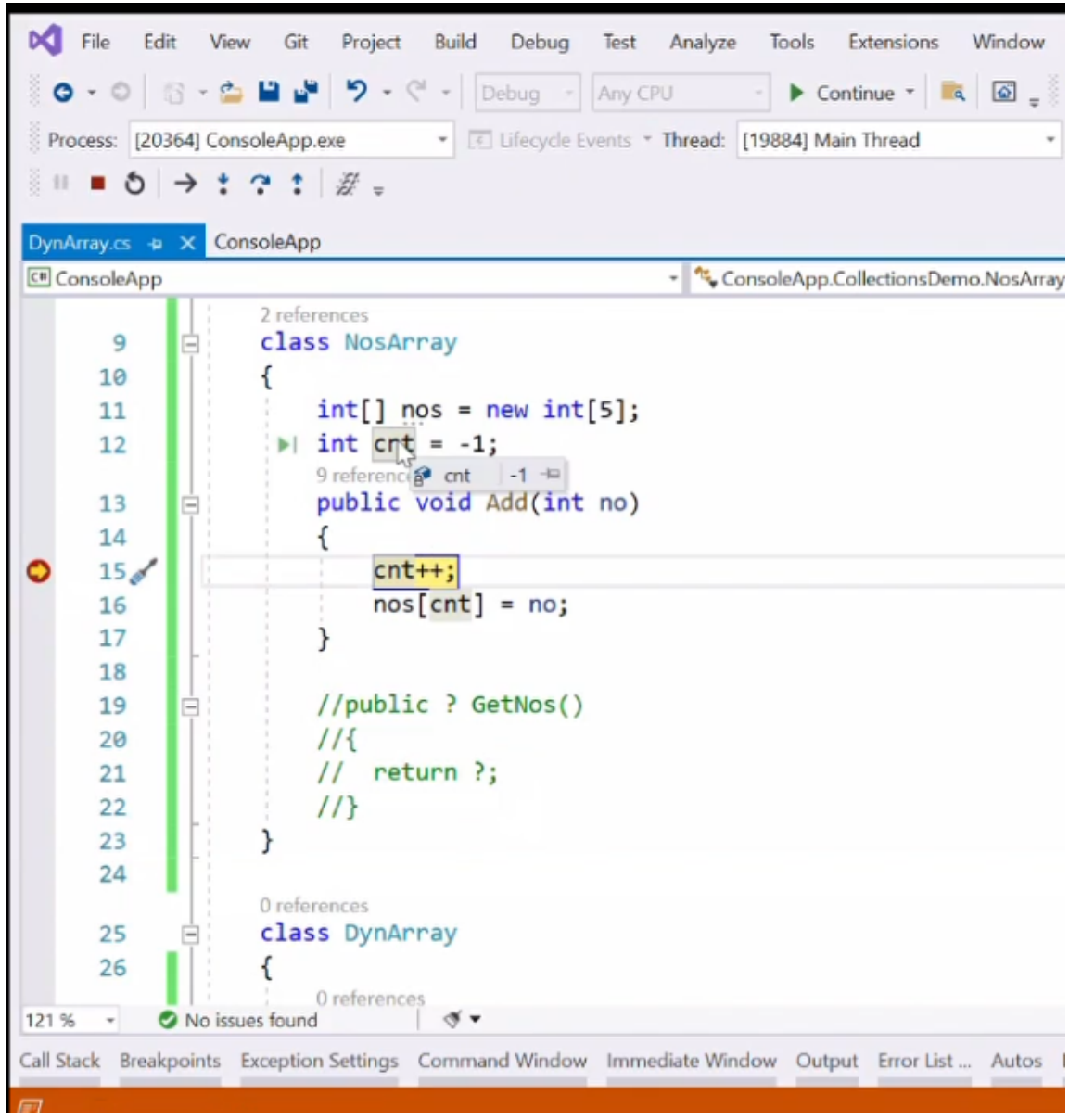
```
int[] nos = new int[5];
int cnt = -1;
9 references
public void Add(int no)
{

    nos[cnt] = no;
}
```

before putting it to my pointer i need to increment the pointer by +1

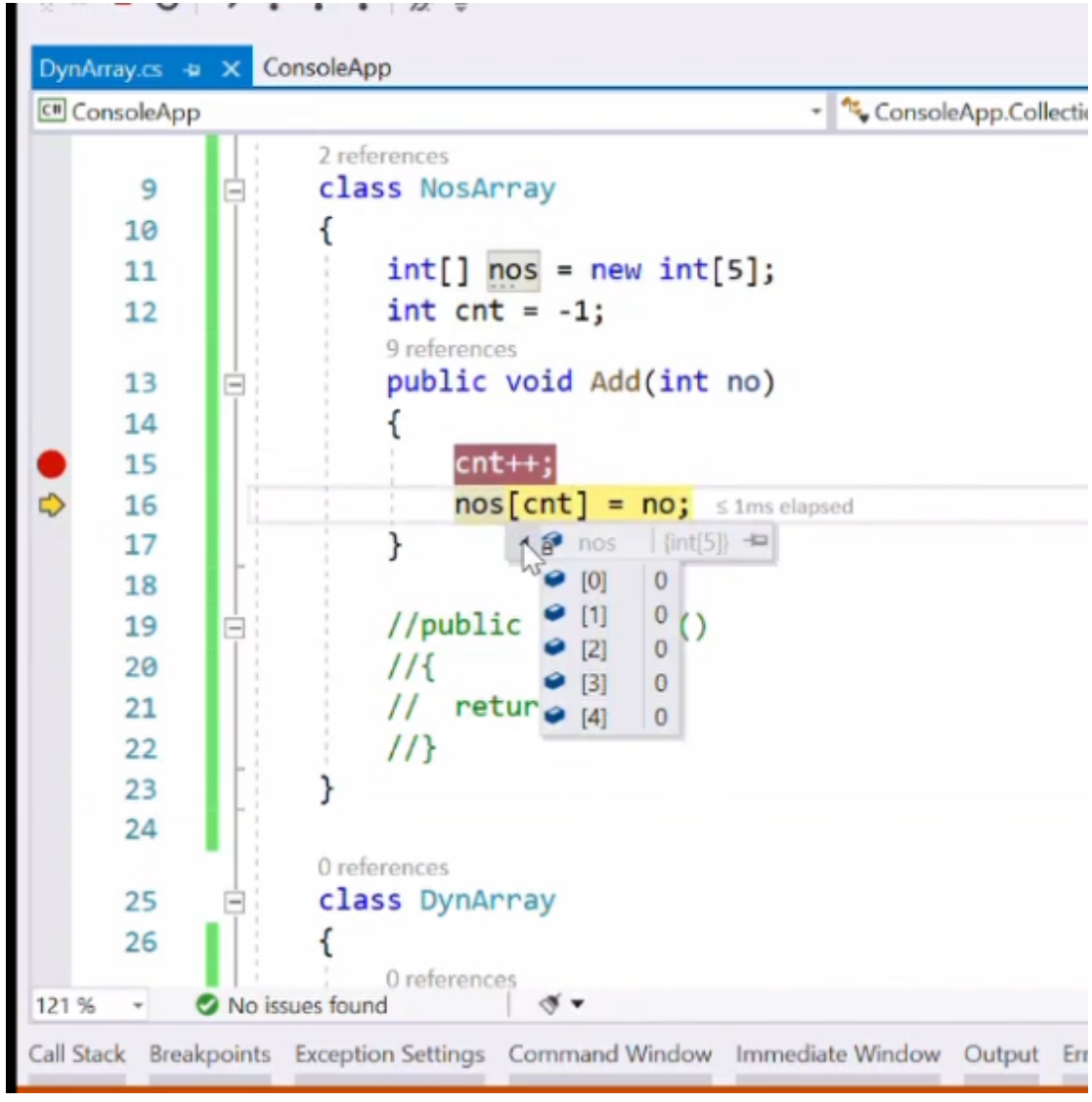
```
2 references
class NosArray
{
    int[] nos = new int[5];
    int cnt = -1;
    9 references
    public void Add(int no)
    {
        cnt++;
        nos[cnt] = no;
    }
```

Now when i do this there are chances that when i read, this will surely put all the numbers

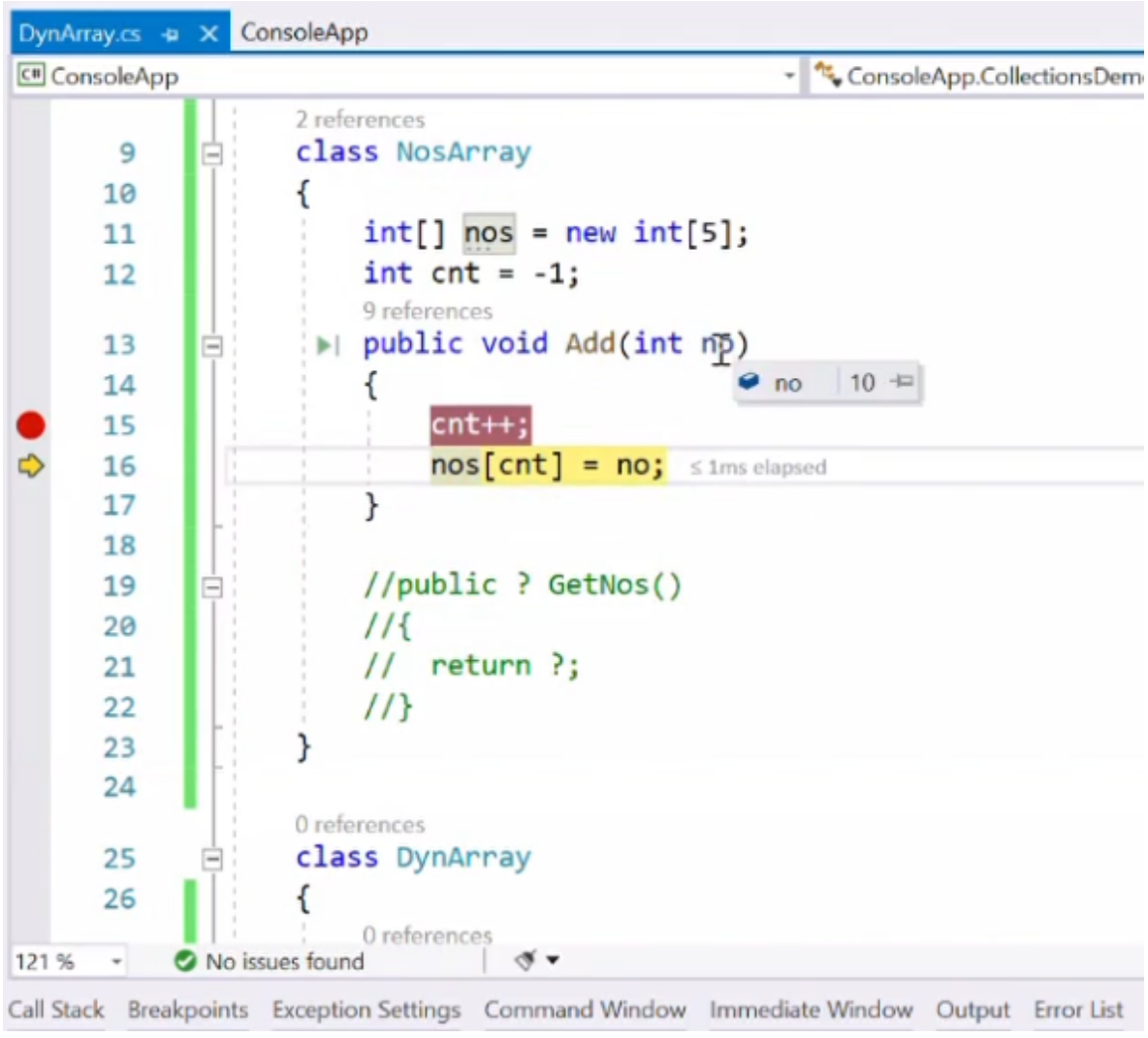
now when i debug look here
cnt is -1

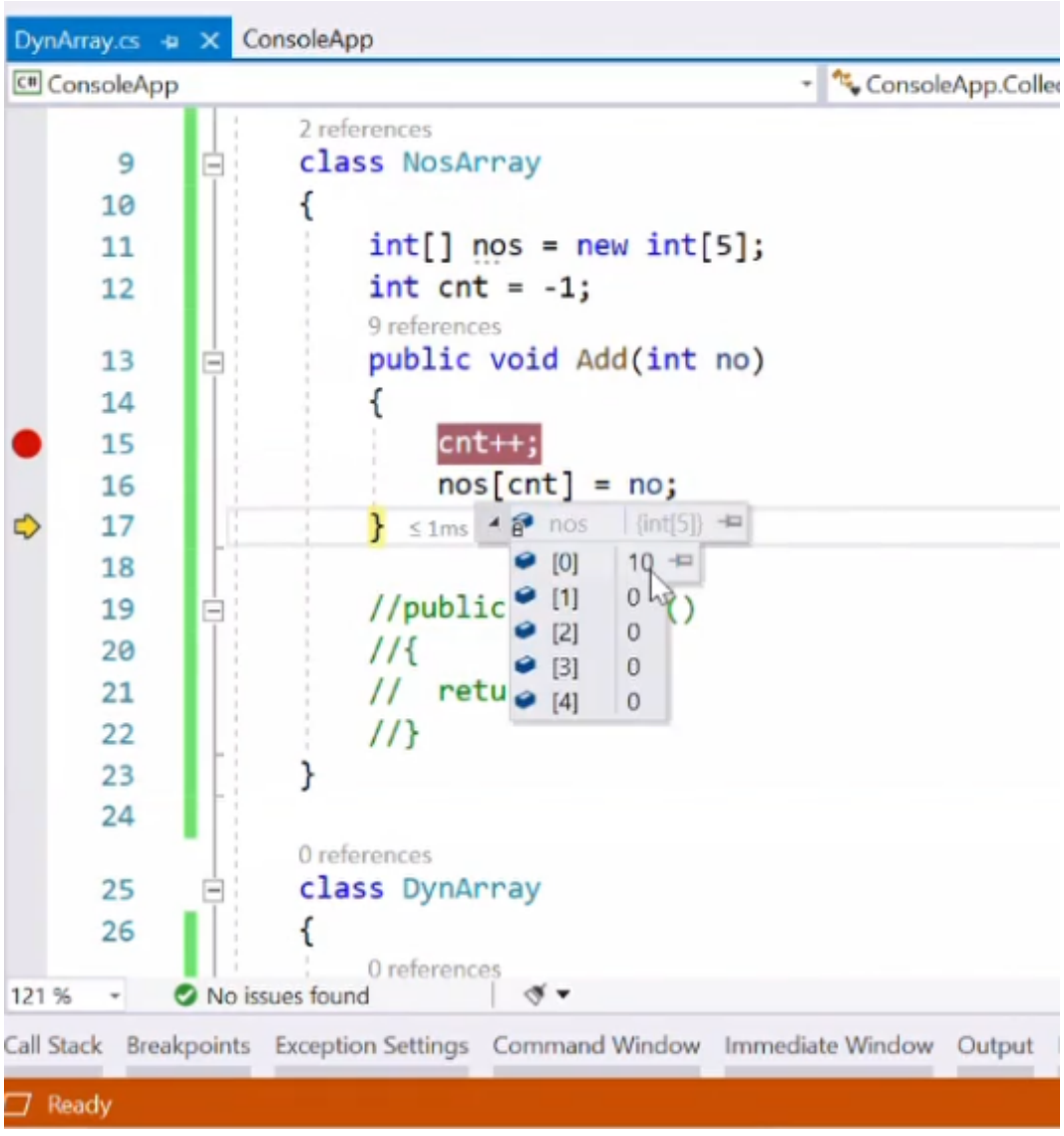

File Edit View Git Project Build Debug Test Analyze Tools Extensions Window
Debug
Any CPU
Continue
Process: [20364] ConsoleApp.exe
Lifecycle Events
Thread: [19884] Main Thread
DynArray.cs
ConsoleApp
ConsoleApp
ConsoleApp.CollectionsDemo.NosArray
2 references
class NosArray
{
int[] nos = new int[5];
int cnt = -1;
9 reference
cnt -1
public void Add(int no)
{
cnt++;
nos[cnt] = no;
}
//public ? GetNos()
//{
// return ?;
//}
}
0 references
class DynArray
{
0 references
121 %
No issues found
Call Stack
Breakpoints
Exception Settings
Command Window
Immediate Window
Output
Error List ...
Autos

Now when i say cnt++ it becomes 0
now nos is having 5 value

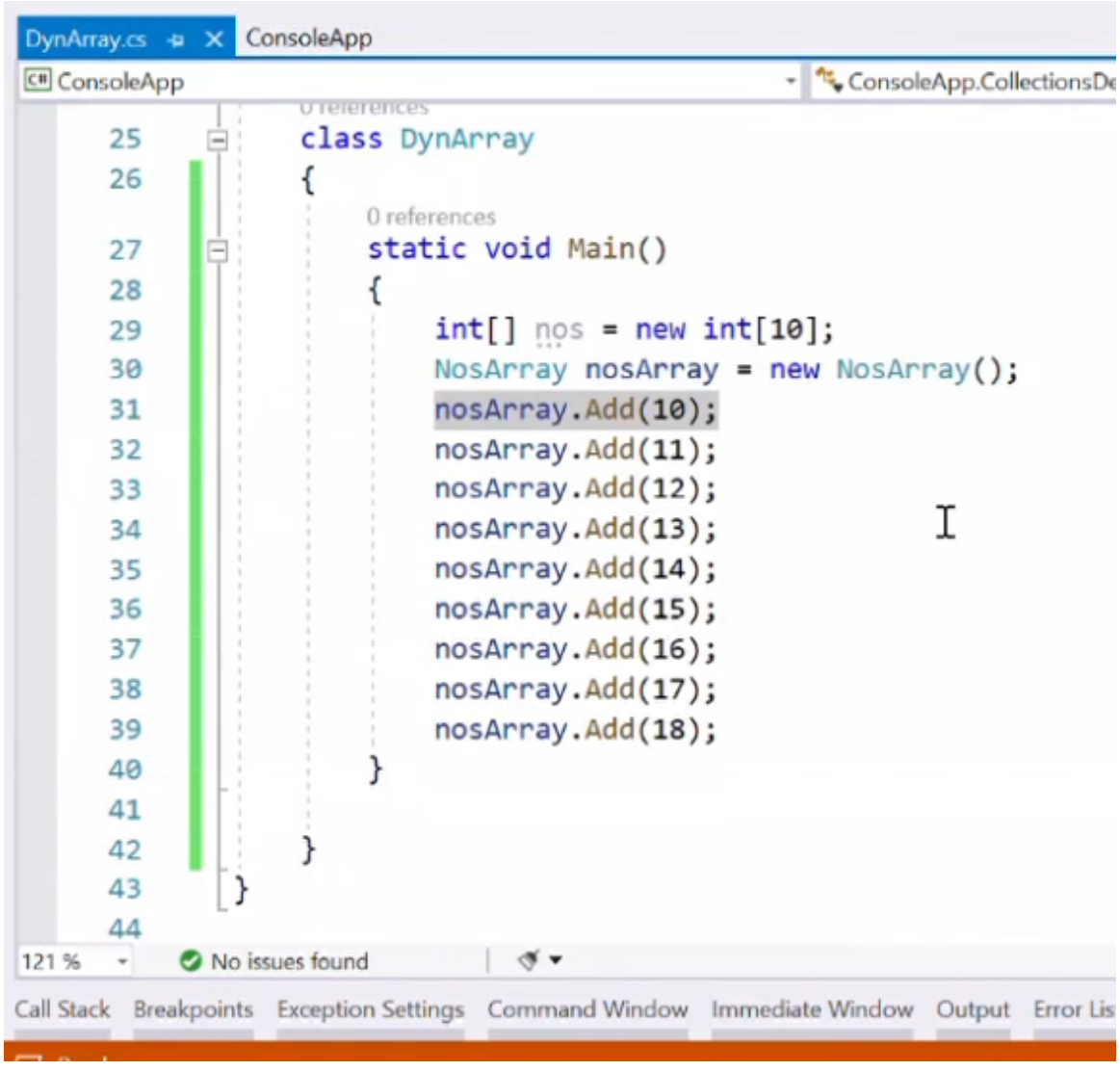


so in nos of 0, i'll store the number i.e 10

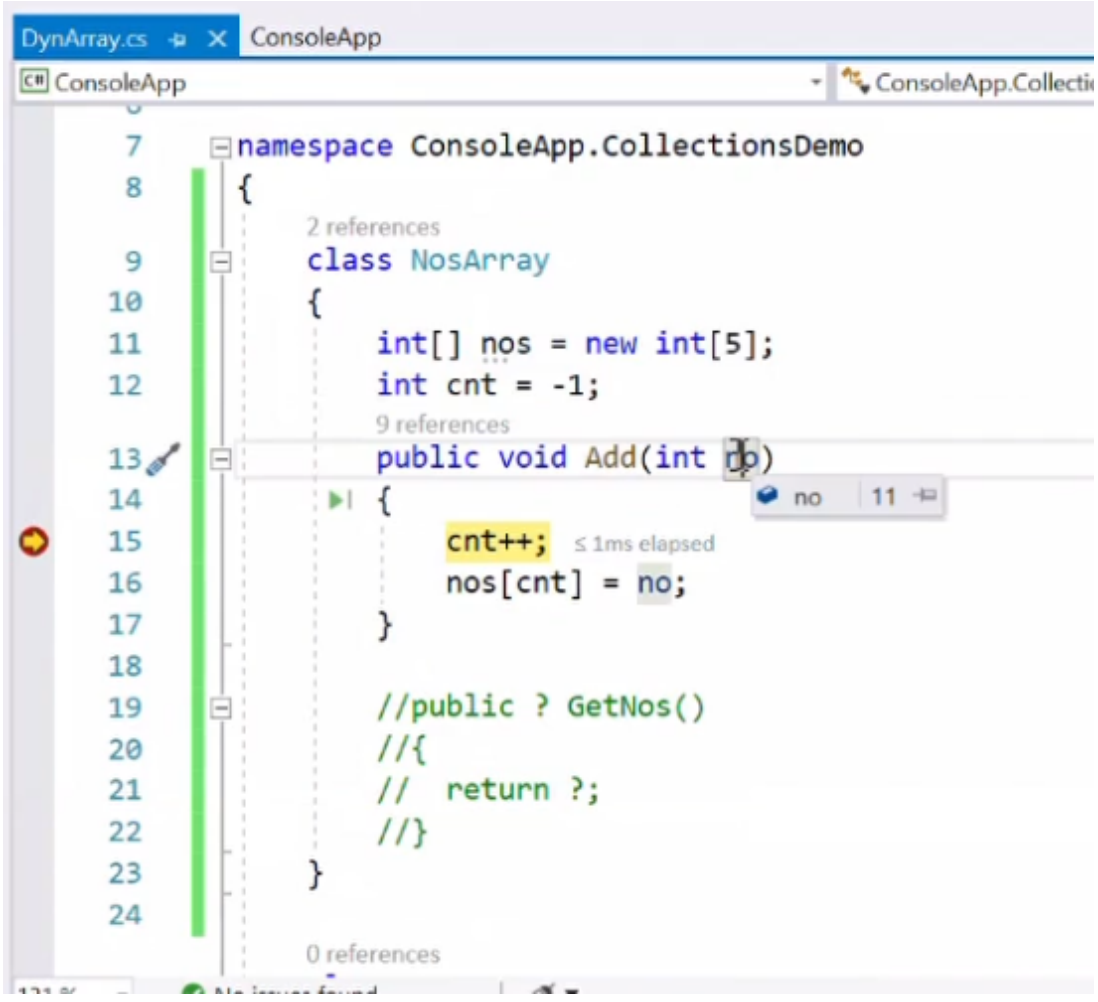

DynArray.cs
ConsoleApp
ConsoleApp
ConsoleApp.CollectionsDem
2 references
class NosArray
{
int[] nos = new int[5];
int cnt = -1;
9 references
public void Add(int no)
{
no 10
cnt++;
nos[cnt] = no;
≤ 1ms elapsed
}
//public ? GetNos()
//{
// return ?;
//}
}
0 references
class DynArray
{
0 references
121 %
No issues found
Call Stack
Breakpoints
Exception Settings
Command Window
Immediate Window
Output
Error List

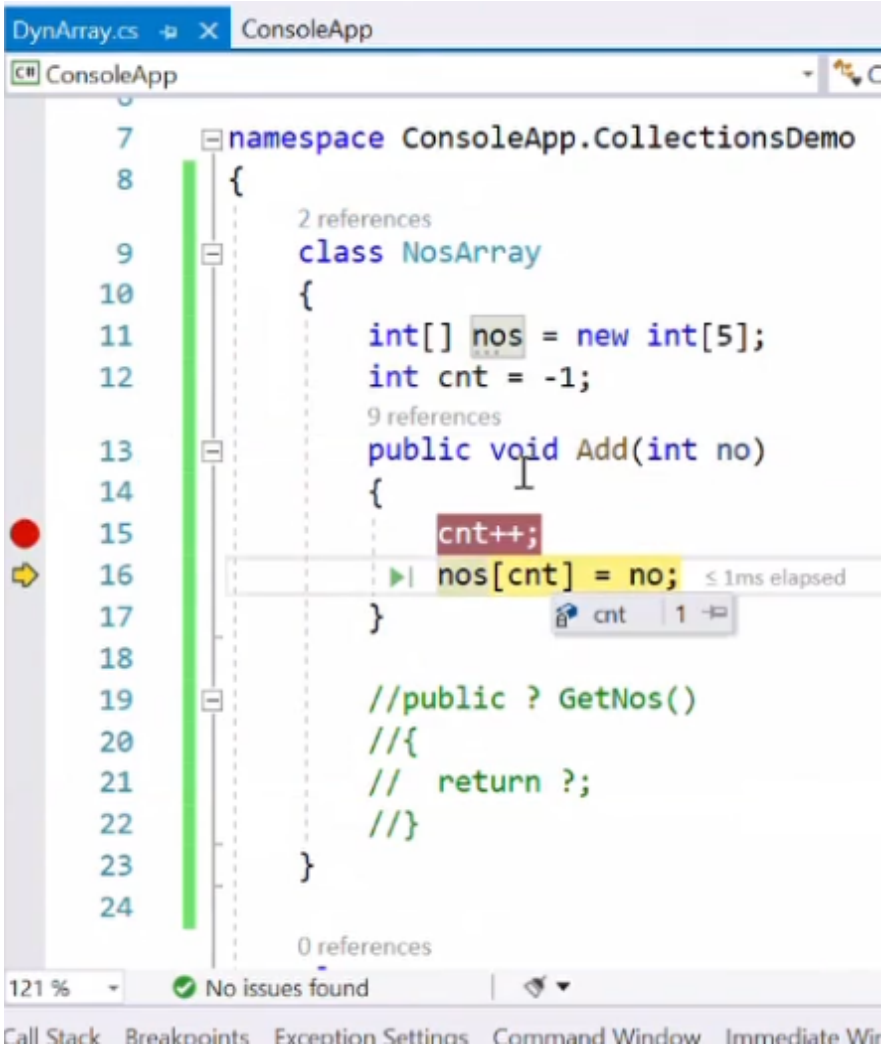


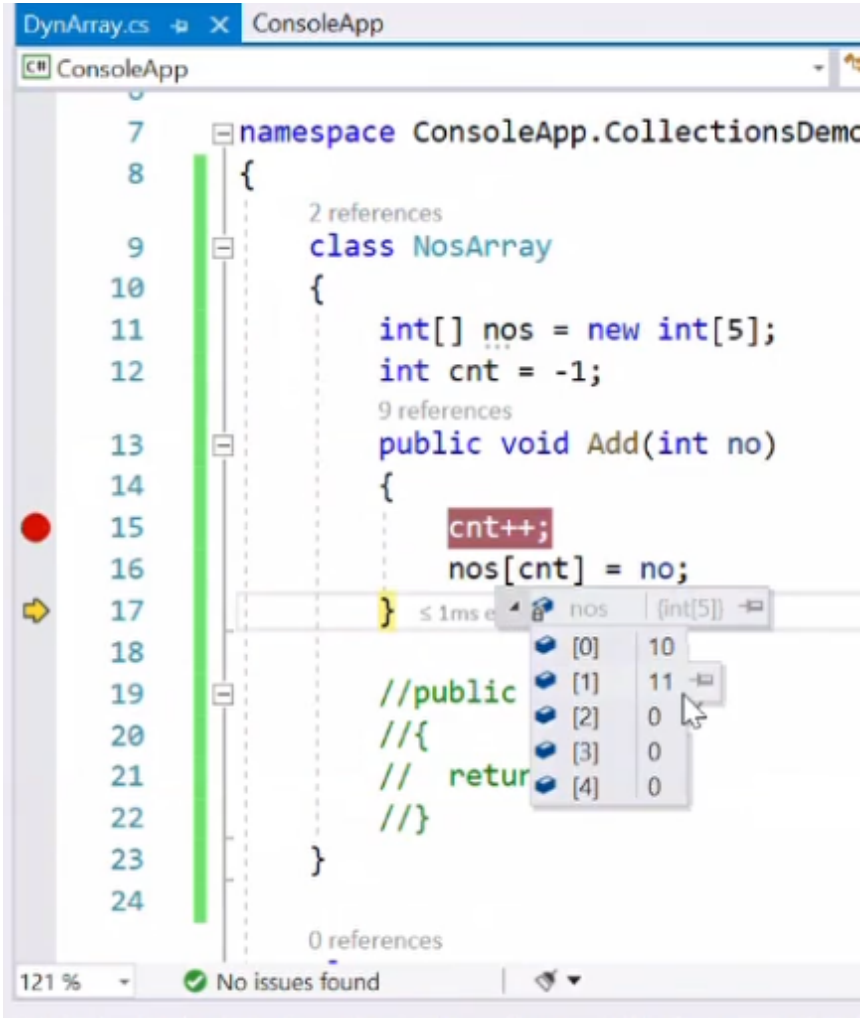
Similarly I'm calling add method to add 11, 12, 13, 14 .....



now again I'm calling nosArray.Add(11)


DynArray.cs
ConsoleApp
ConsoleApp
ConsoleApp.Collecti
namespace ConsoleApp.CollectionsDemo
{
2 references
class NosArray
{
int[] nos = new int[5];
int cnt = -1;
9 references
public void Add(int no)
{
no 11
cnt++; ≤ 1ms elapsed
nos[cnt] = no;
}
//public ? GetNos()
//{
// return ?;
//}
}
0 references

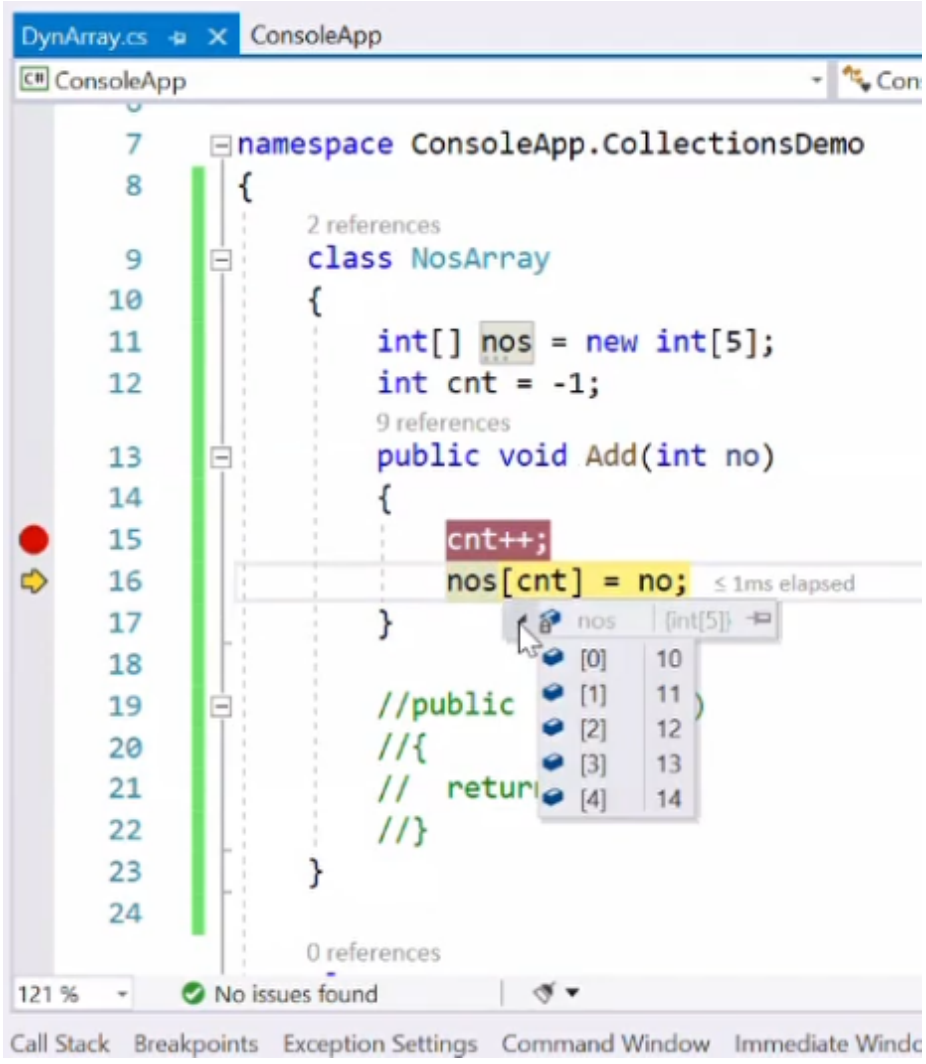

DynArray.cs
ConsoleApp
ConsoleApp
namespace ConsoleApp.CollectionsDemo
{
2 references
class NosArray
{
int[] nos = new int[5];
int cnt = -1;
9 references
public void Add(int no)
{
cnt++;
nos[cnt] = no;
≤ 1ms elapsed
cnt
1
}
//public ? GetNos()
//{
// return ?;
//}
}
0 references
121 %
No issues found
Call Stack
Breakpoints
Exception Settings
Command Window
Immediate Wi



look 11 stored here

similarly 12, 13, 14 will be stored here

now when we are trying to insert 15 look at this array is already filled



So when array is filled, where will i store the data

So when i run this I'll get **Array Index out of bond Exception**

So how will I control that

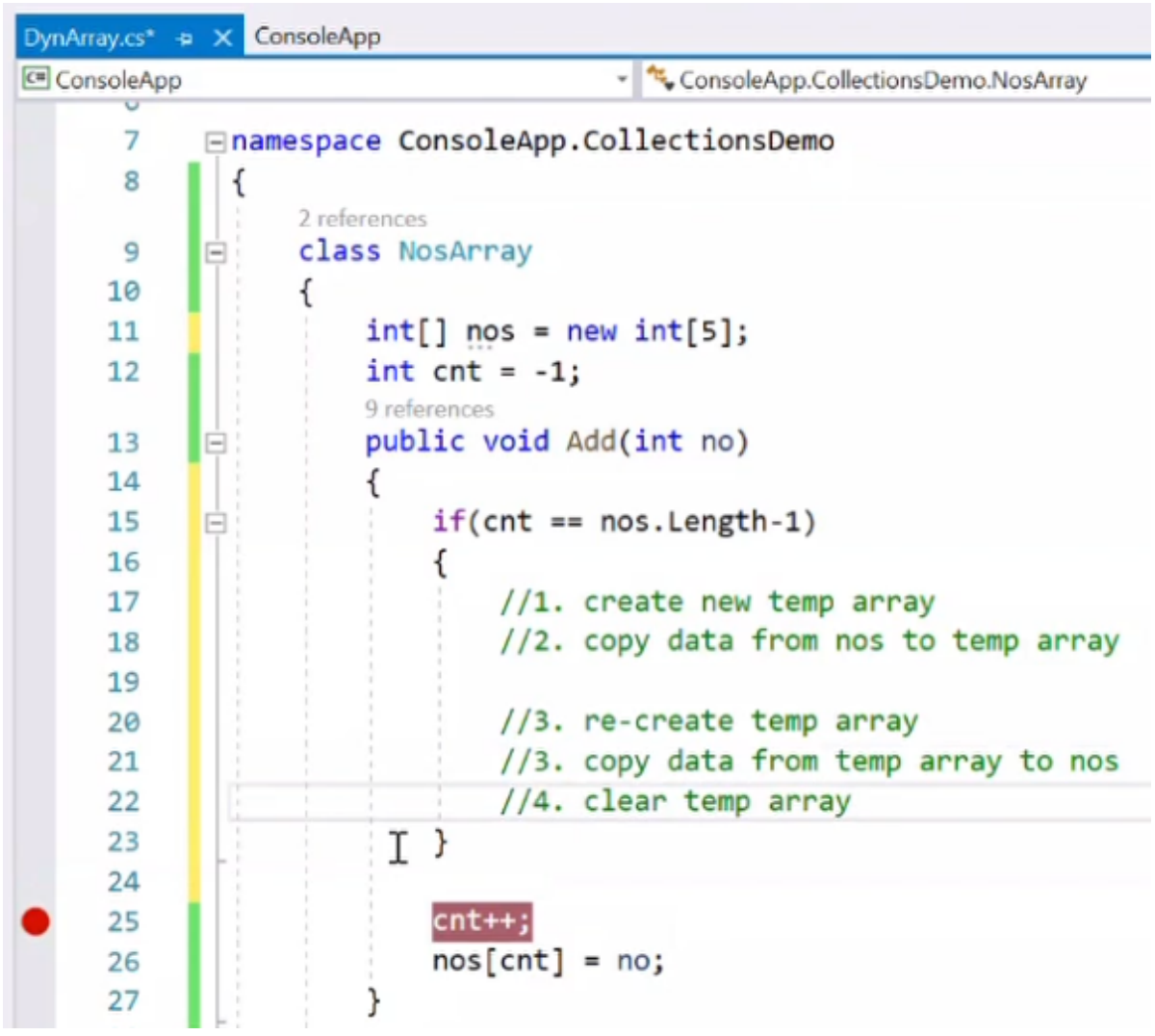
How will you extend the length

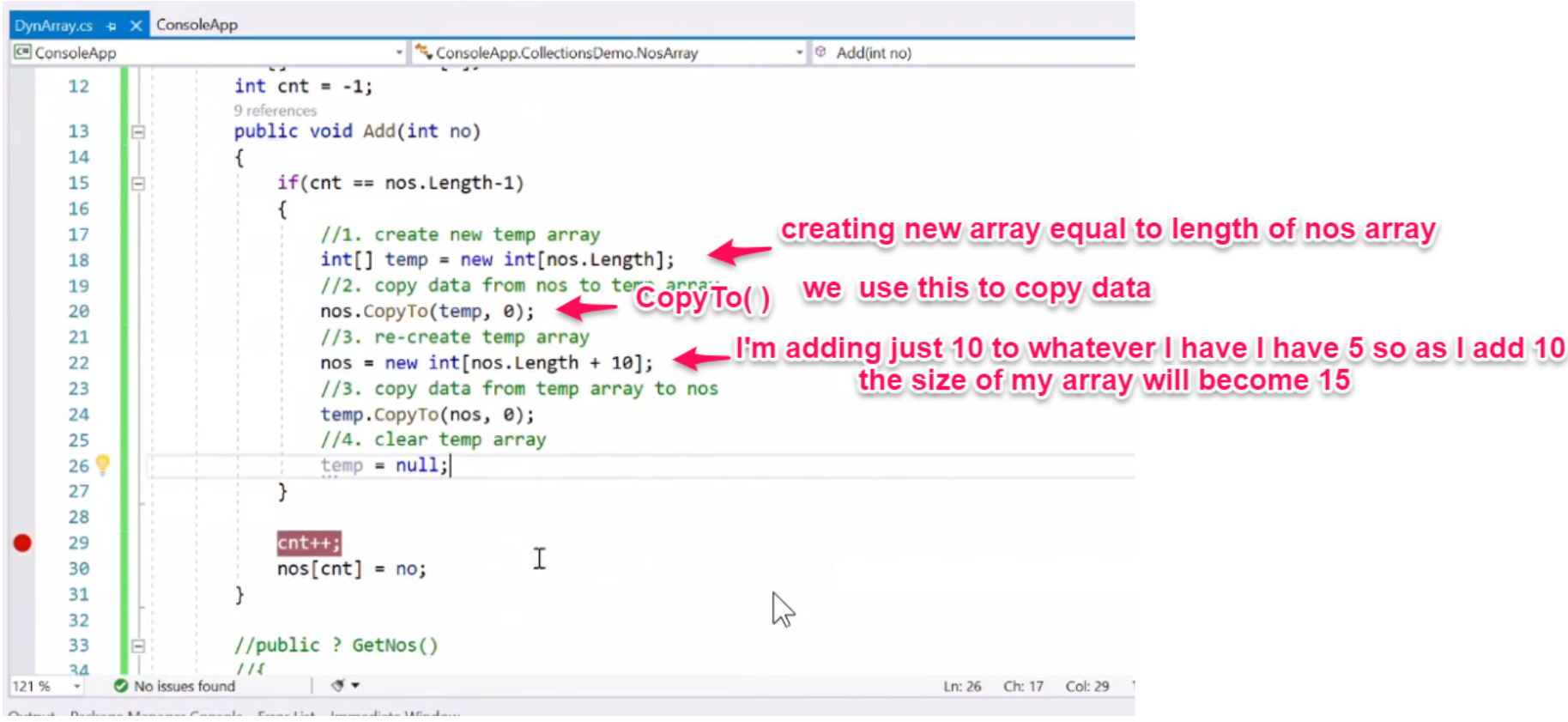
When will you extend the length or Recreate new array

---------- when cnt == nos.length - 1

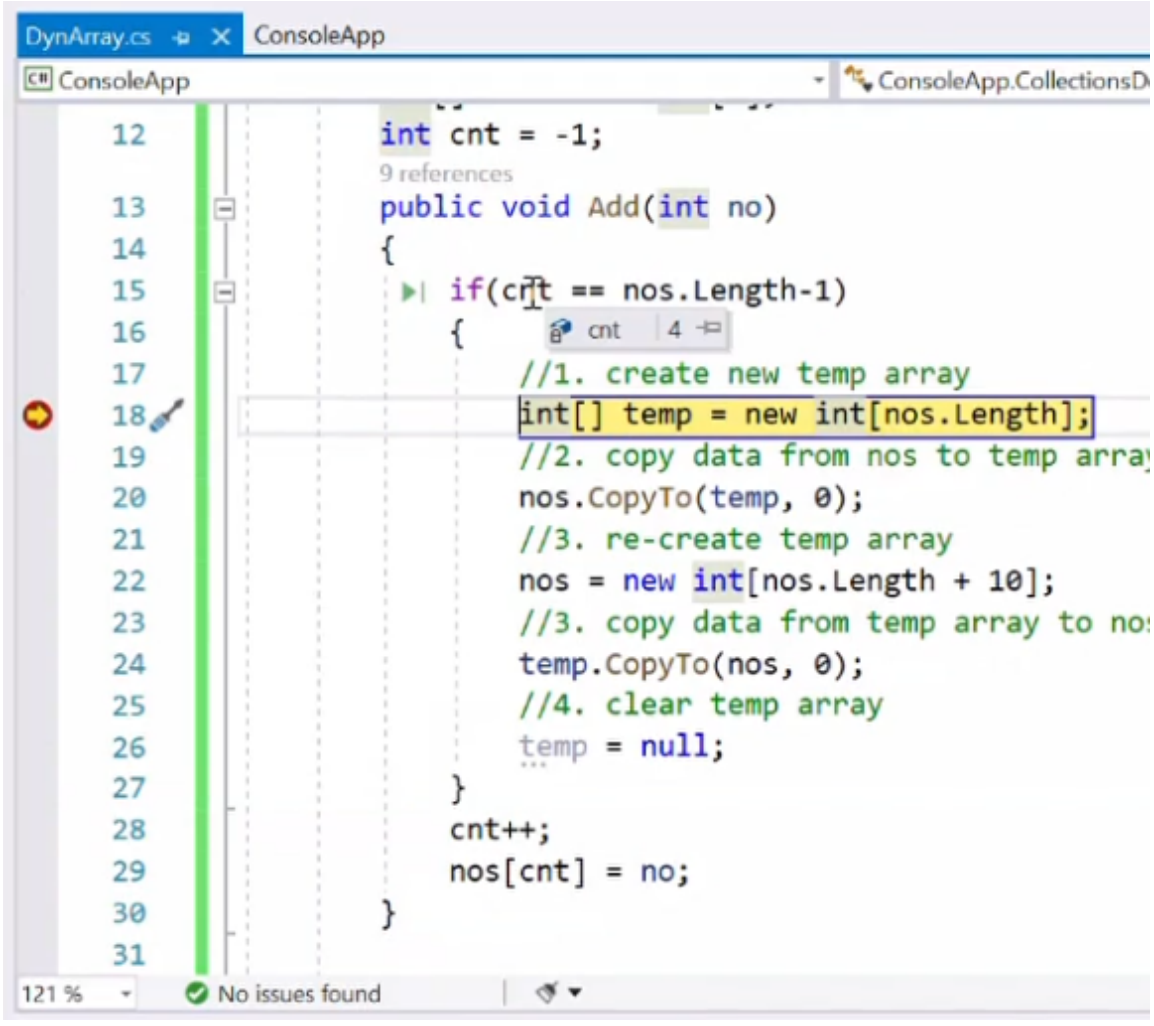
S if this condition is true then

1. I create a new temp array
2. Copy data from nos to temp array
3. Re-create the temp array
4. Copy data from temp array to nos
5. Clear temp array

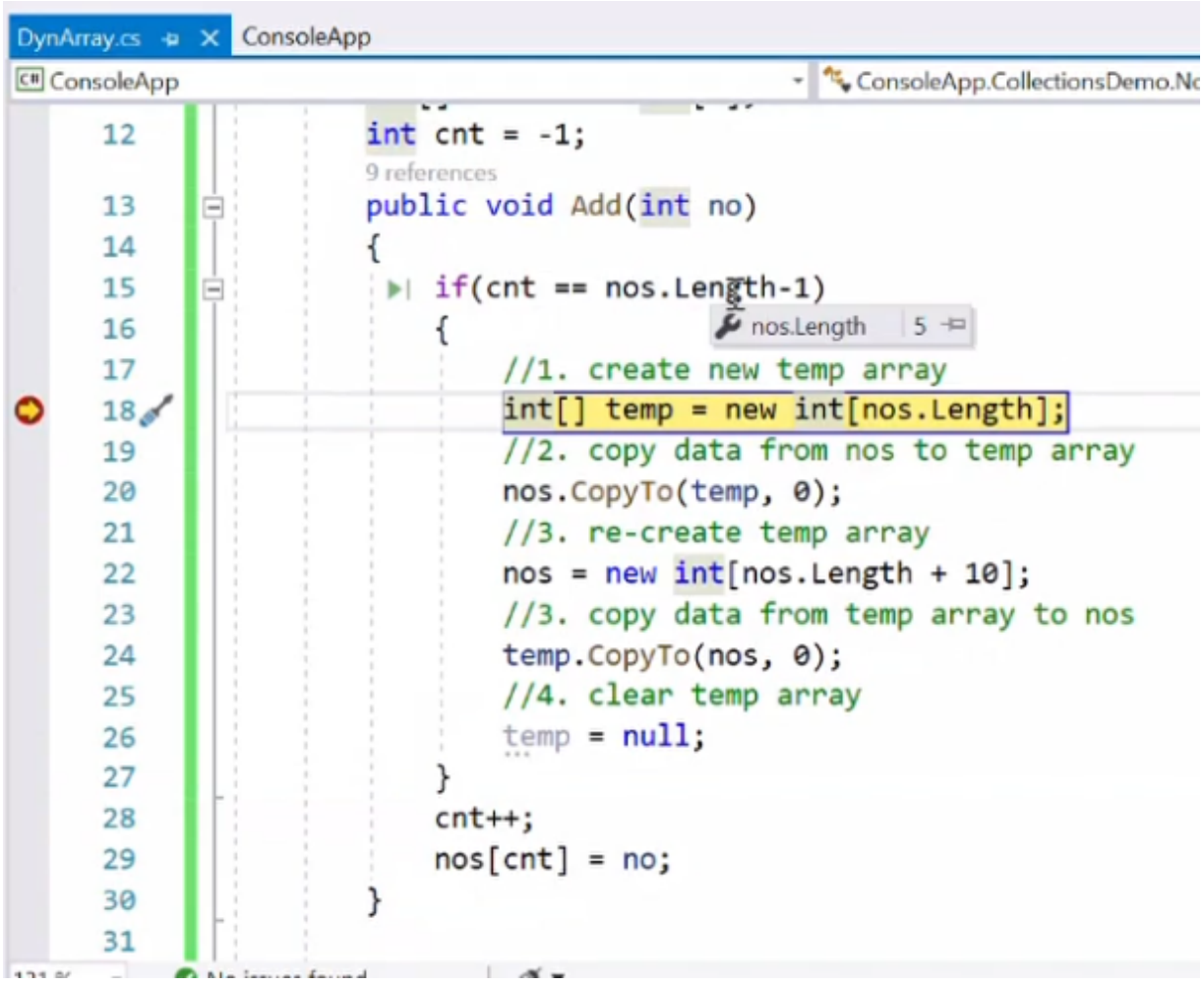

DynArray.cs*
ConsoleApp
ConsoleApp
ConsoleApp.CollectionsDemo.NosArray
namespace ConsoleApp.CollectionsDemo
{
2 references
class NosArray
{
int[] nos = new int[5];
int cnt = -1;
9 references
public void Add(int no)
{
if(cnt == nos.Length-1)
{
//1. create new temp array
//2. copy data from nos to temp array
//3. re-create temp array
//3. copy data from temp array to nos
//4. clear temp array
}
cnt++;
nos[cnt] = no;
}



my cnt value is 4

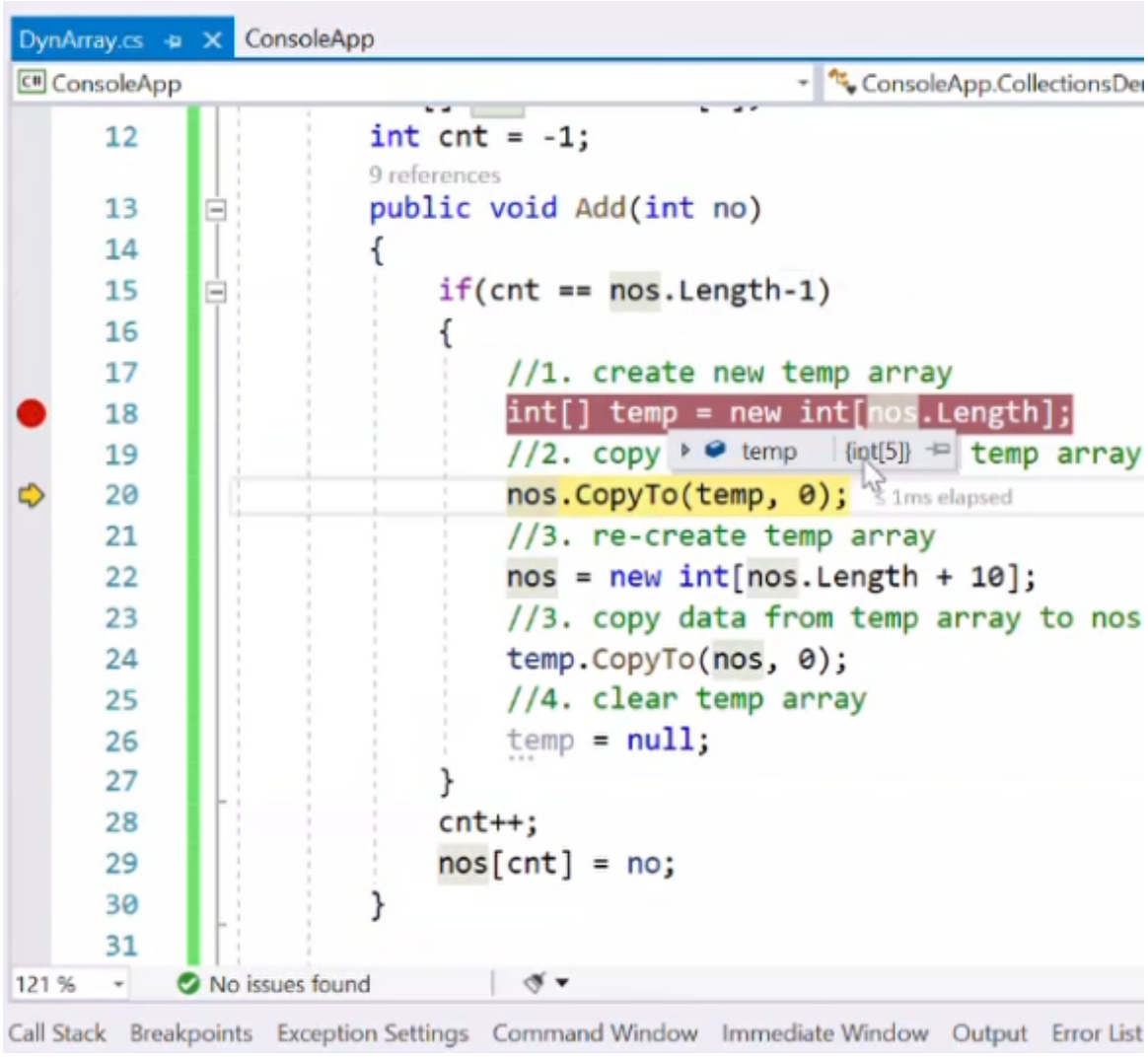


nos.length is 5 that's a reason we are doing -1 to it , so that it match 4 = = 4

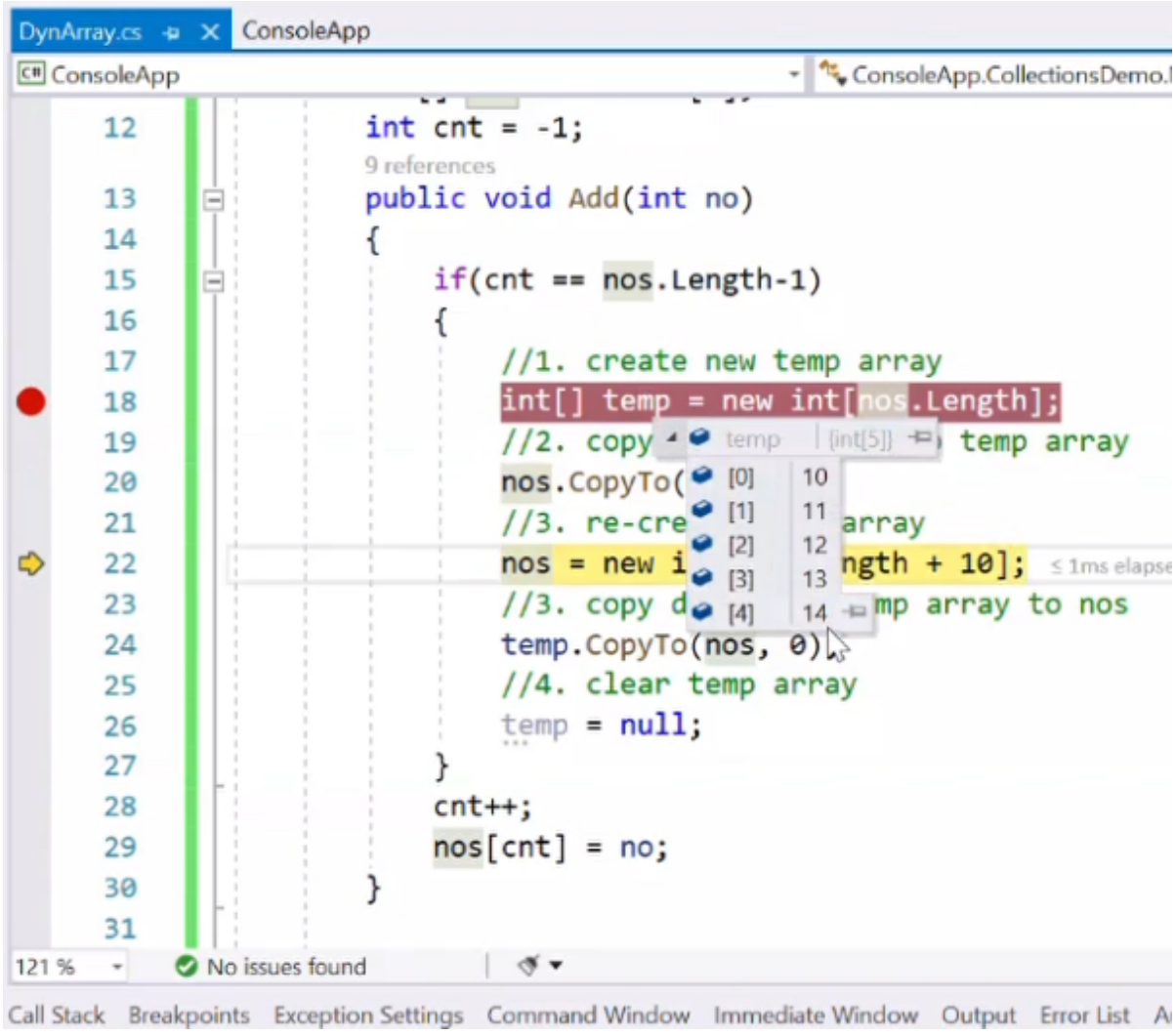


so condition matches

and temp array is created of size 5

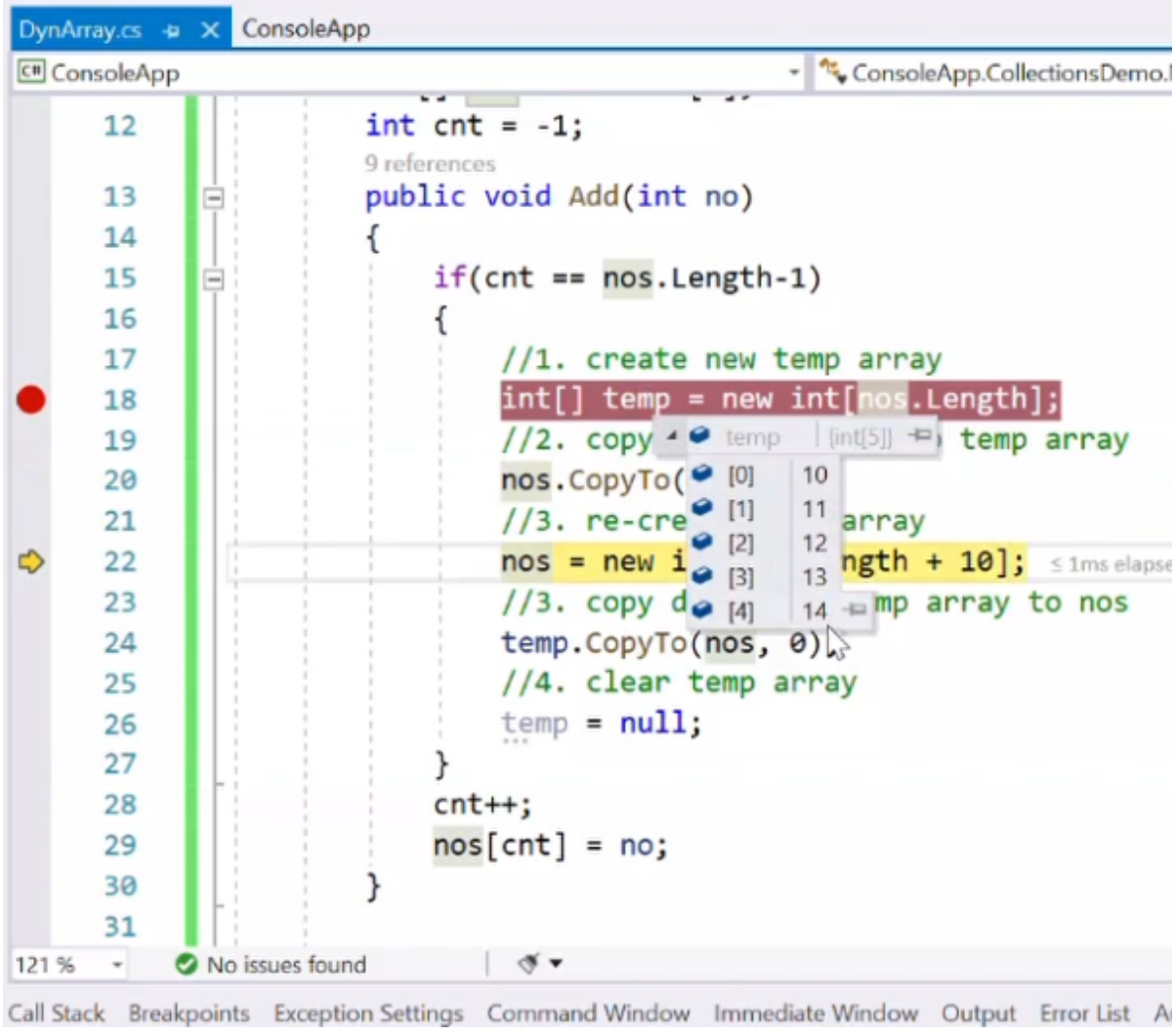


then i'm copying all the data to temp

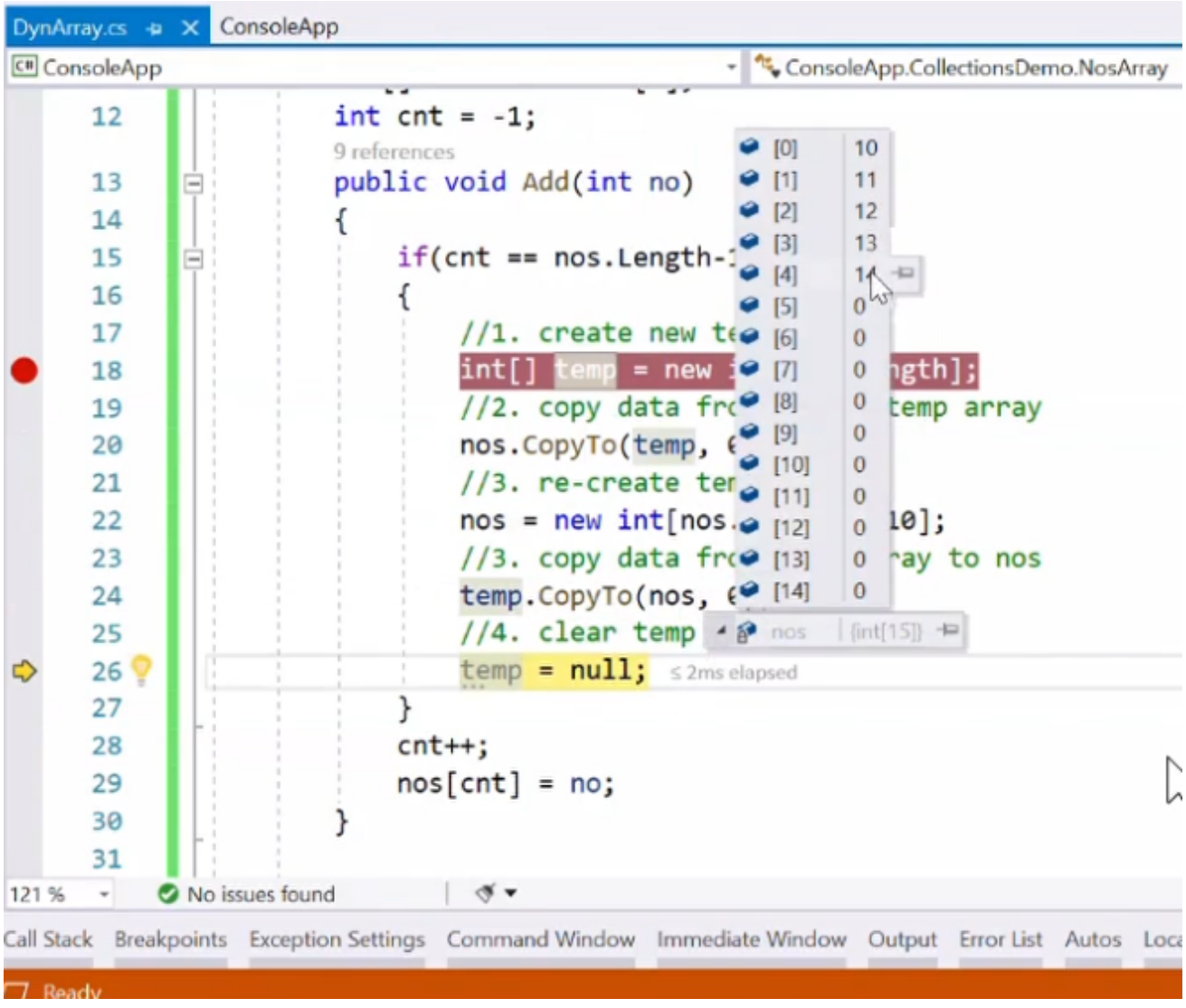


after this i'll resize nos array to +10, so it becomes of size 15 and everything is been initialize to 0

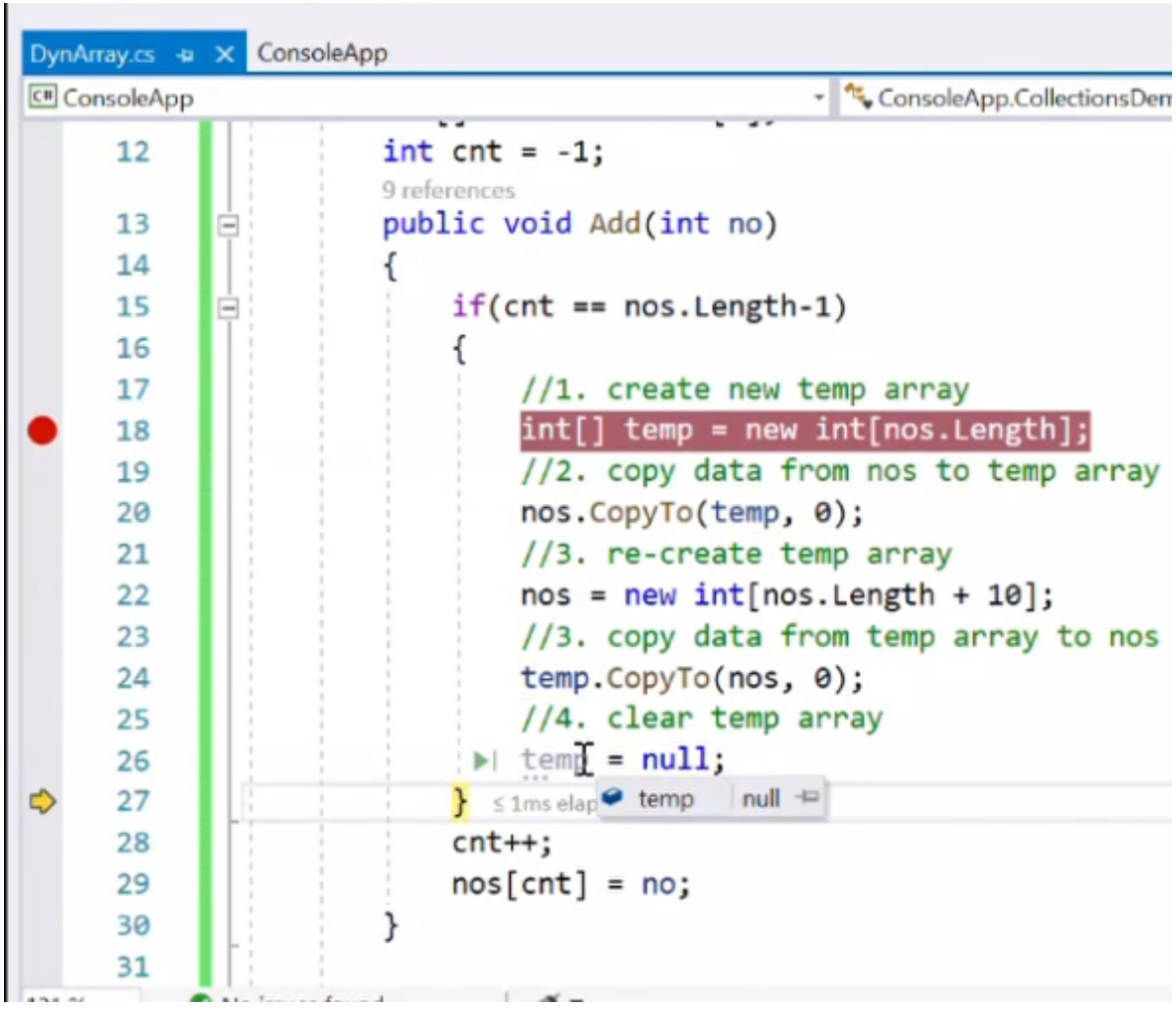
Remember: When ever you are creating a new integer array, it will be initialize to 0, not null



after that we copy temp data to nos

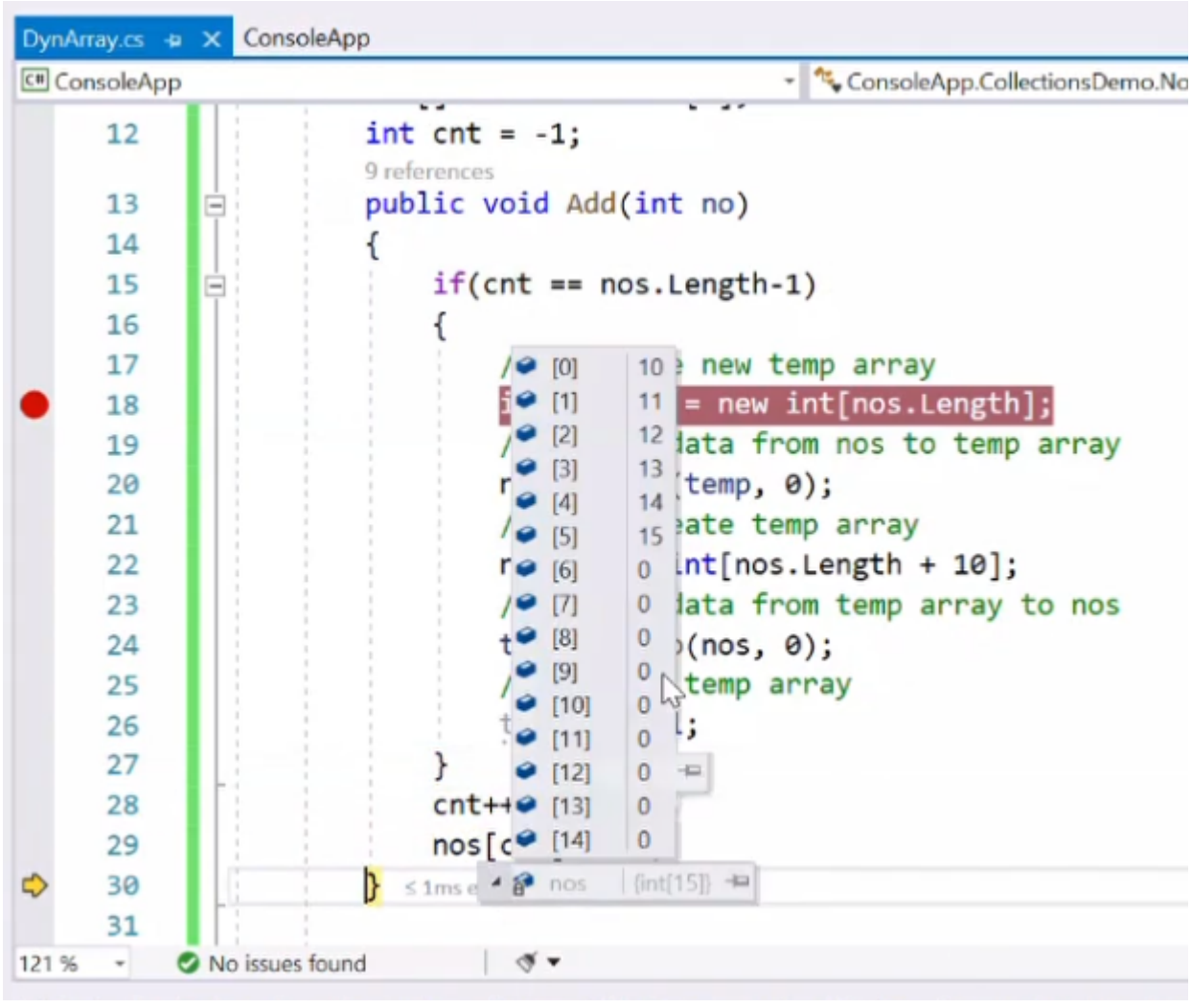


after that I initialize temp to null

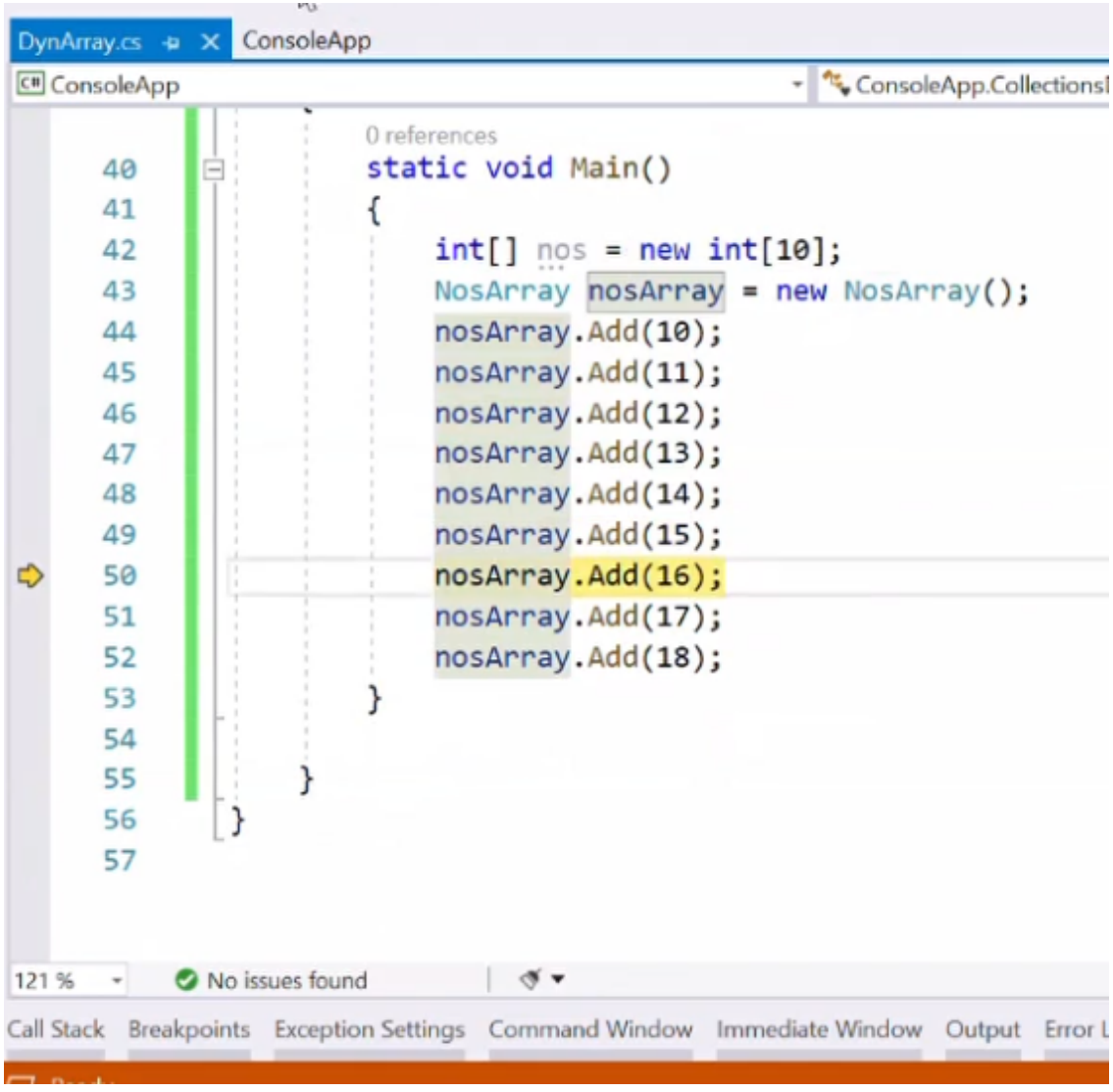


now it will store me the next number

so 15 will be stored

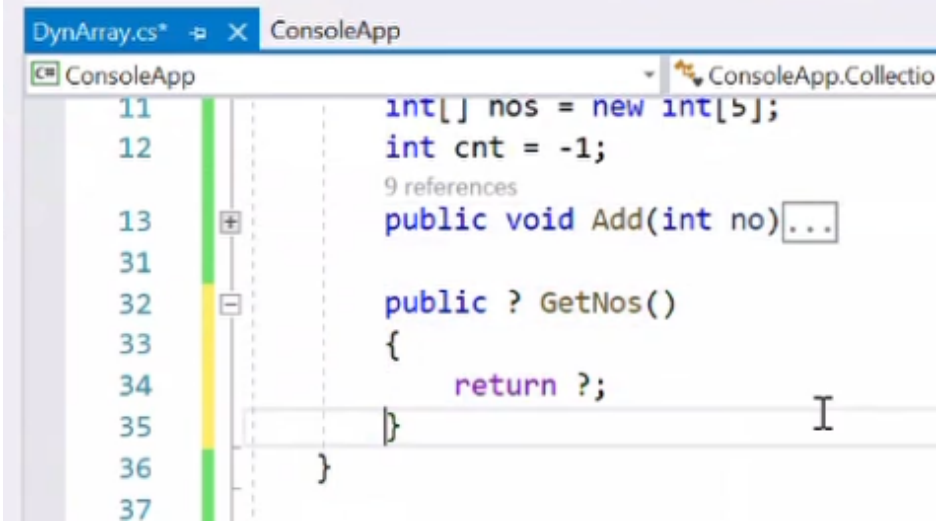


then it go for 16

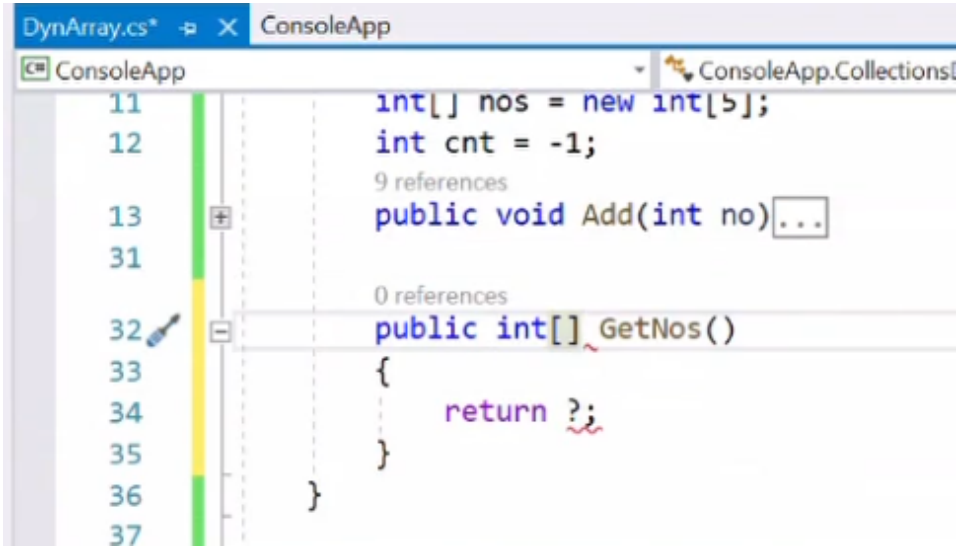


similarly 17, 18 all this numbers will be stored

Now How will you write get method?

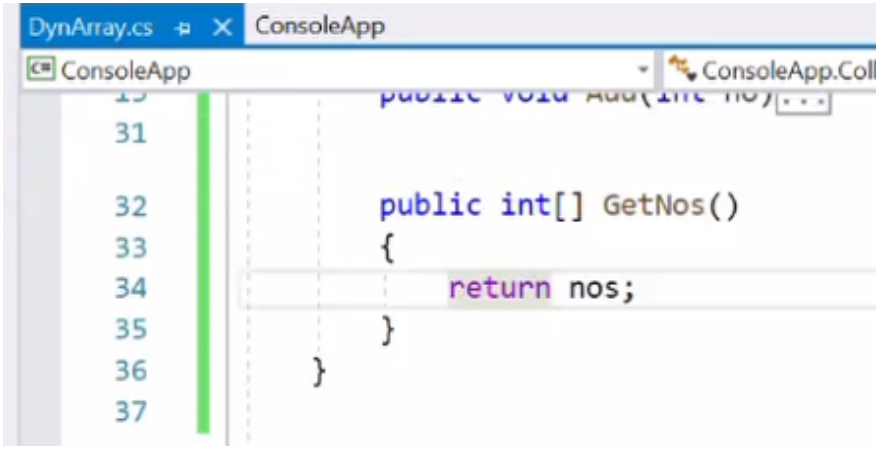


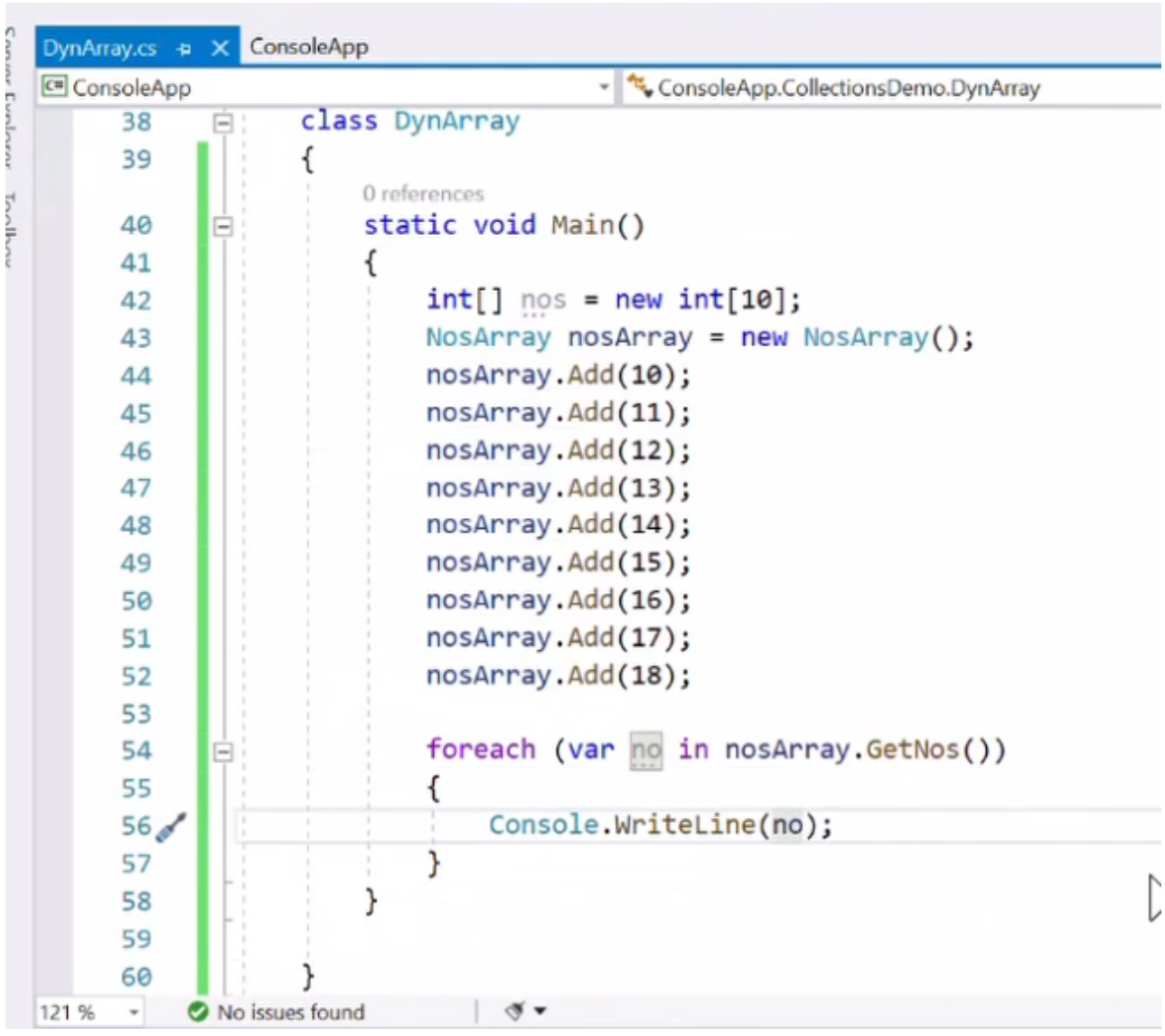
so for 1st ? obviously as it is a array it will return the numeric array only so int

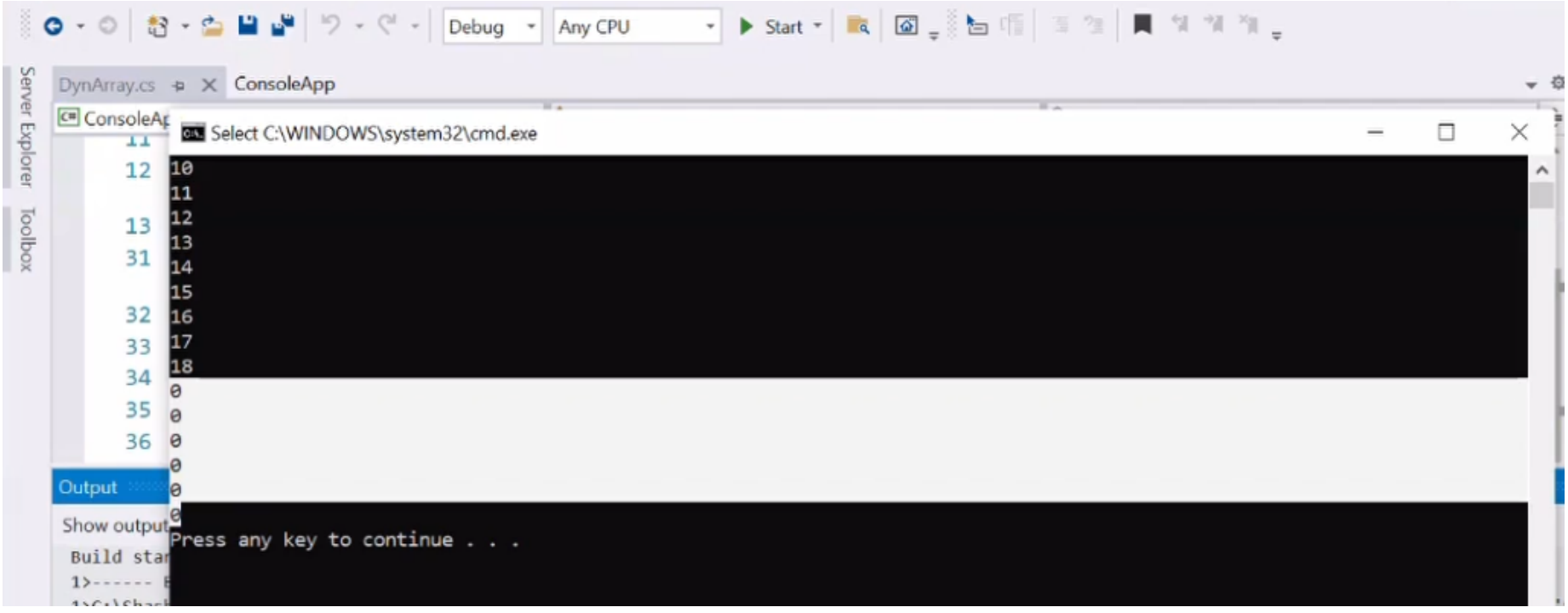


Now we need last modified Index

what if we return nos




DynArray.cs
ConsoleApp
ConsoleApp
ConsoleApp.CollectionsDemo.DynArray
class DynArray
{
0 references
static void Main()
{
int[] nos = new int[10];
NosArray nosArray = new NosArray();
nosArray.Add(10);
nosArray.Add(11);
nosArray.Add(12);
nosArray.Add(13);
nosArray.Add(14);
nosArray.Add(15);
nosArray.Add(16);
nosArray.Add(17);
nosArray.Add(18);
foreach (var no in nosArray.GetNos())
{
Console.WriteLine(no);
}
}
}
121 %
No issues found

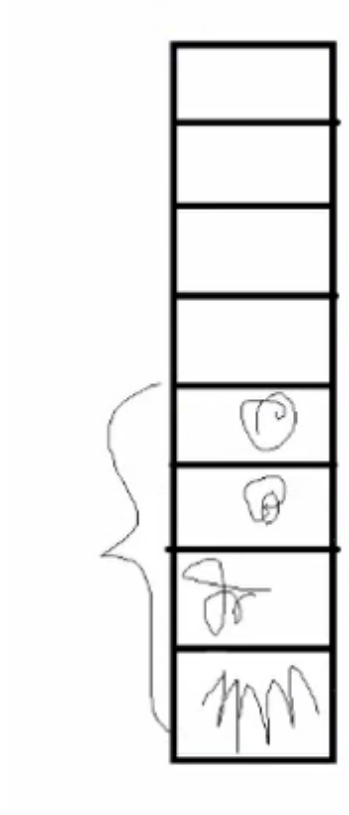


problem is we get the 0 also here

but when i say get

I want to get all the numbers meaning from 10 to 18, i don't want that 0's

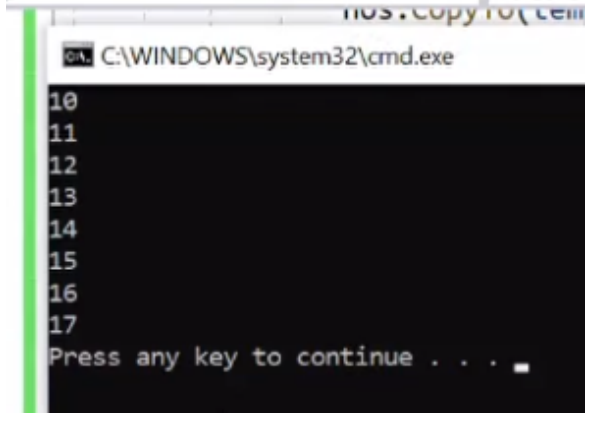
how to avoid this 0's



we have to return this 4 item from here
But how do you know that you have 4 item here, who will know that i have 4 items
----------cnt

you need to use some technique to read till cnt

```
1 reference
public int[] GetNos()
{
    int[] temp = new int[cnt];
    for (int i = 0; i < cnt; i++)
    {
        temp[i] = nos[i];
    }
    return temp;
}
```



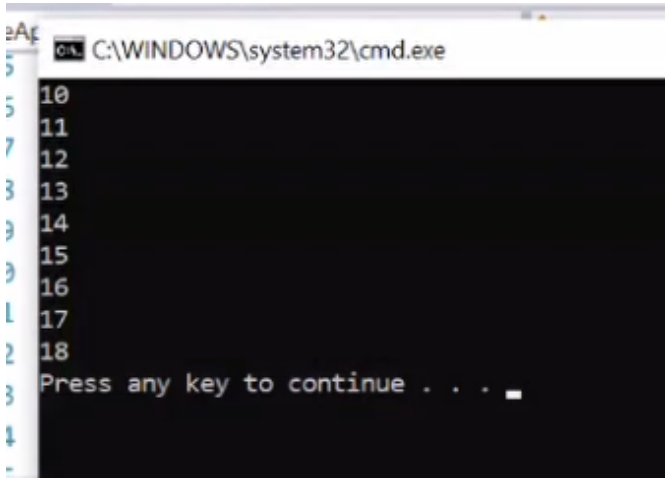
i'm getting till 17 but i should get till 18

<= aega

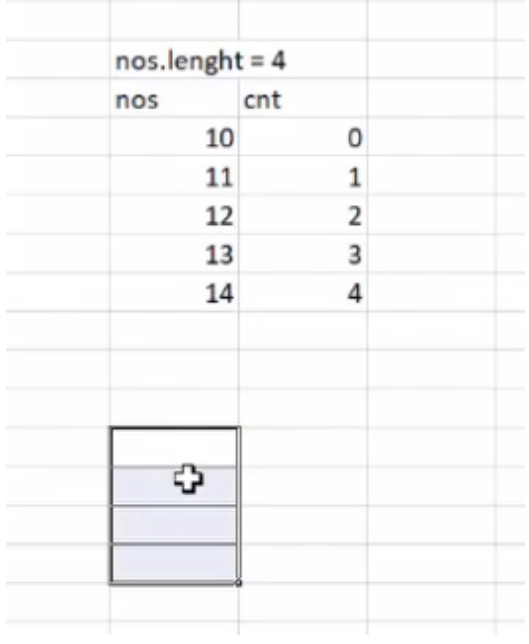
```
public int[] GetNos()
{
    int[] temp = new int[cnt];
    for (int i = 0; i <= cnt; i++)
    {
        temp[i] = nos[i];
    }
    return temp;
}
```

for this at line 34 we have to take cnt + 1

```
1 reference
32      public int[] GetNos()
33      {
34          int[] temp = new int[cnt + 1];
35          for (int i = 0; i <= cnt; i++)
36          {
37              temp[i] = nos[i];
38          }
39          return temp;
40      }
```



by doing dry run we found the error



so this is how people started thinking like array has its own disadvantage what shall we do
we'll write some code which will automatically increment the array allow the user to store particular data
so they created a package called collection and in the collection they started adding all this thing

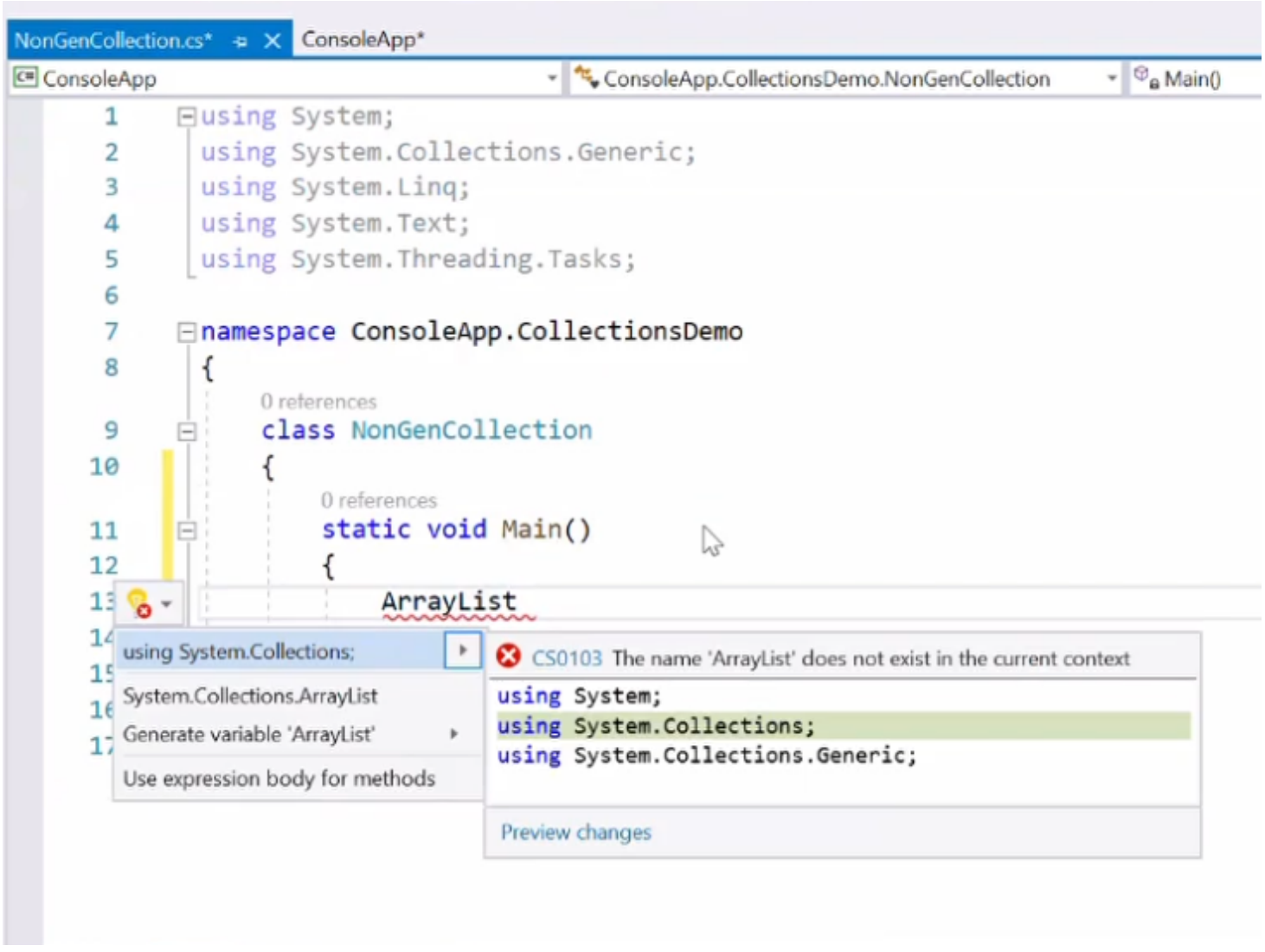
In collection we have 2 different type

1. **Generic**
2. **Non-Generic**

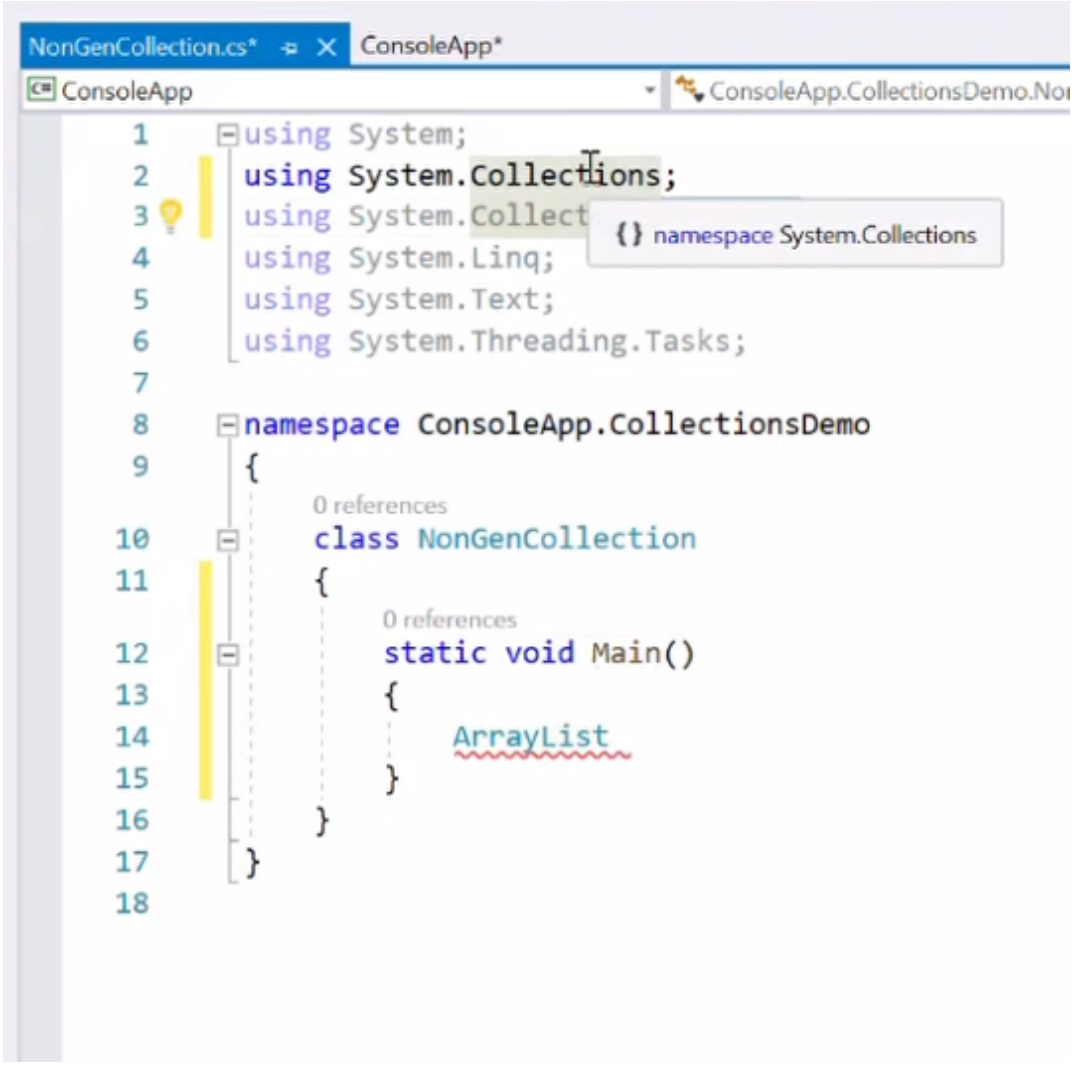
Earlier we used to use Non-Generic

What is Non-Generic collection ?

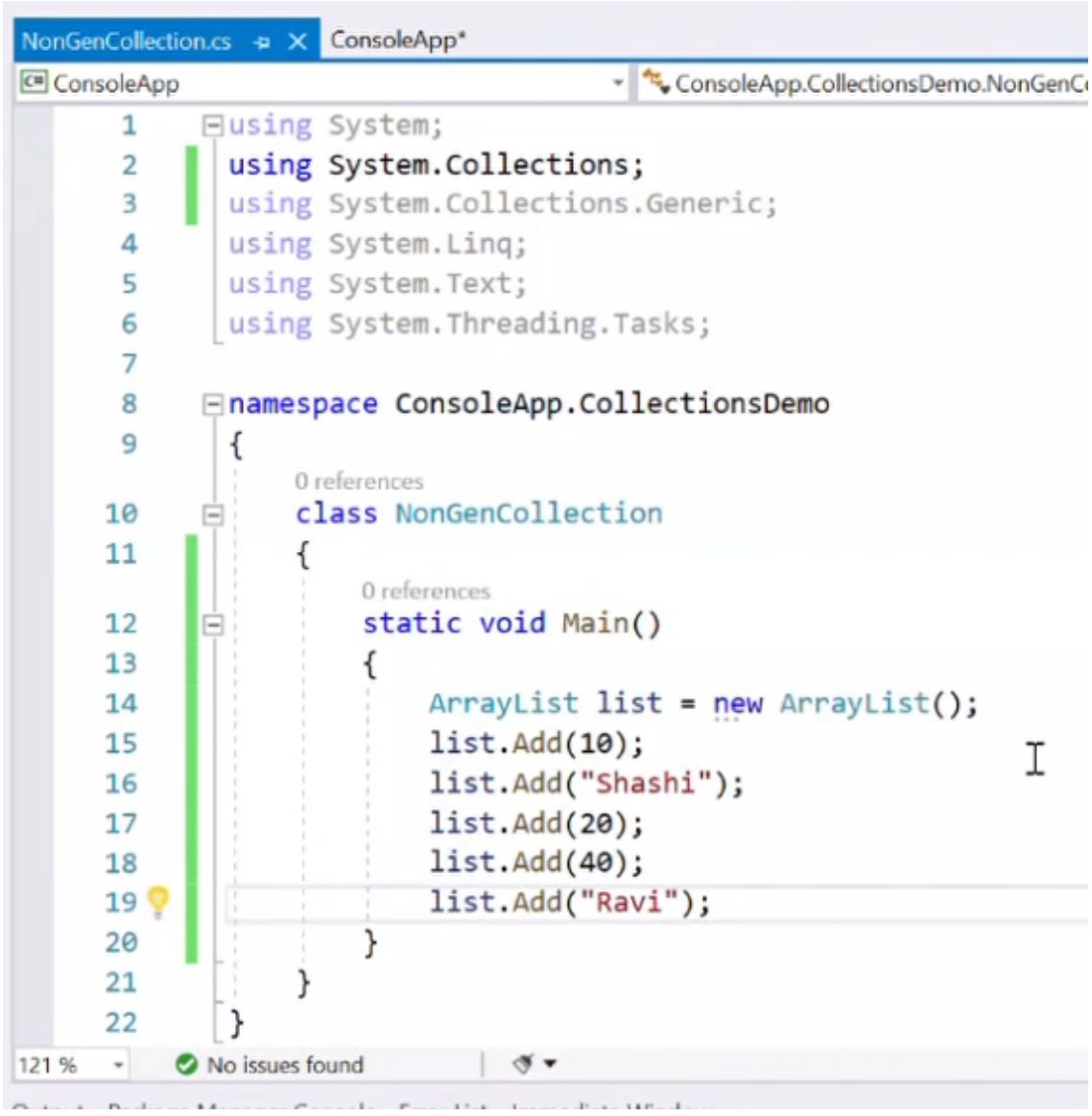
----Same array
But here it store me Object



Look at this ArrayList comes from using System.Collections
not from Collection.Generic



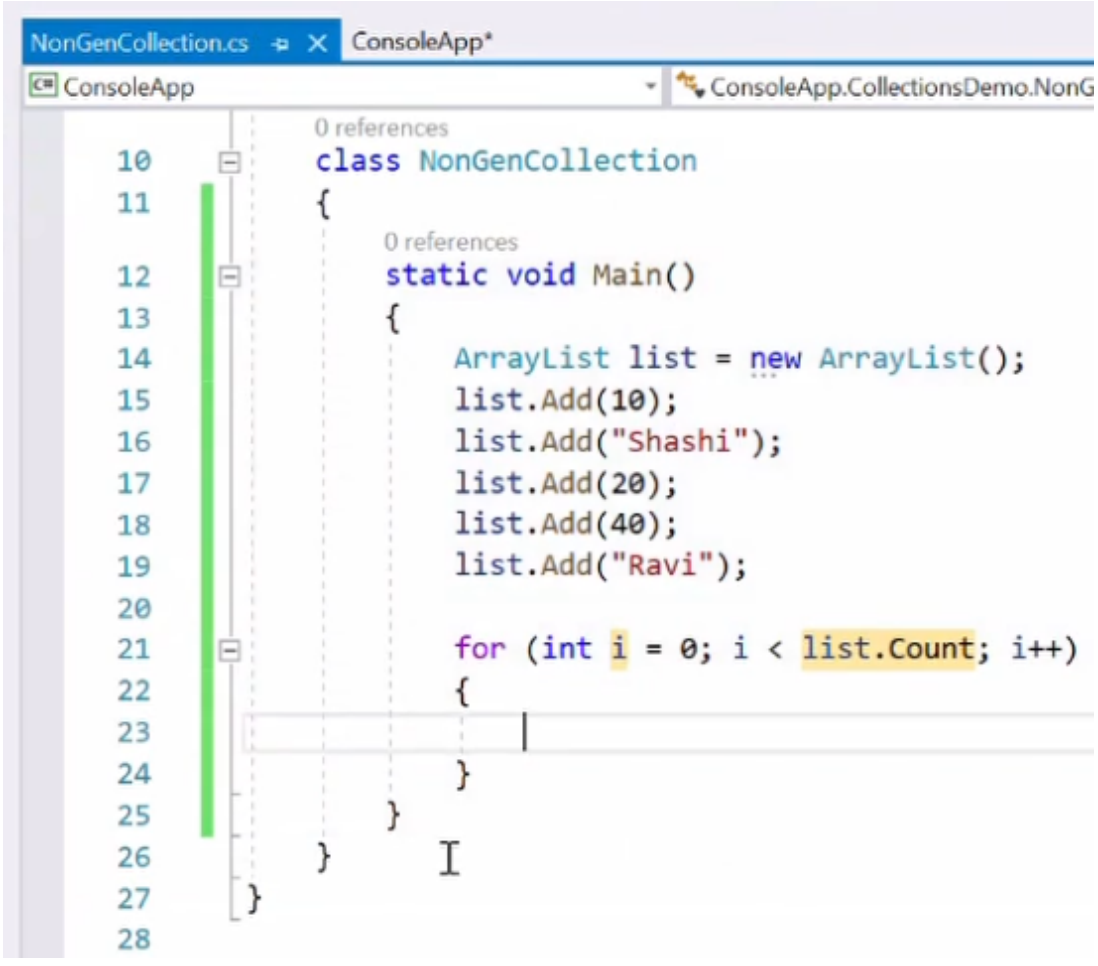
Look at this it allows me to store anything


NonGenCollection.cs
ConsoleApp*
ConsoleApp
ConsoleApp.CollectionsDemo.NonGenC
using System;
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace ConsoleApp.CollectionsDemo
{
0 references
class NonGenCollection
{
0 references
static void Main()
{
ArrayList list = new ArrayList();
list.Add(10);
list.Add("Shashi");
list.Add(20);
list.Add(40);
list.Add("Ravi");
}
}
}
121 %
No issues found

Now what is the disadvantage of using this kind of list

Now as I'm allowing the user to add 10 as well as "Shashi"
So how will you search , How will you know what datatype user have entered we don't know

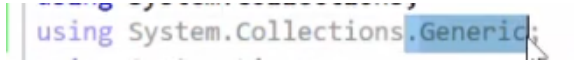
then i have to run a for loop till list.count
then take individual element
then check this particular element is of the integer type or it is string type
then Unbox it, then fetch the data



So that's becomes a biggest disadvantage

That's a reason they introduce something called Generics

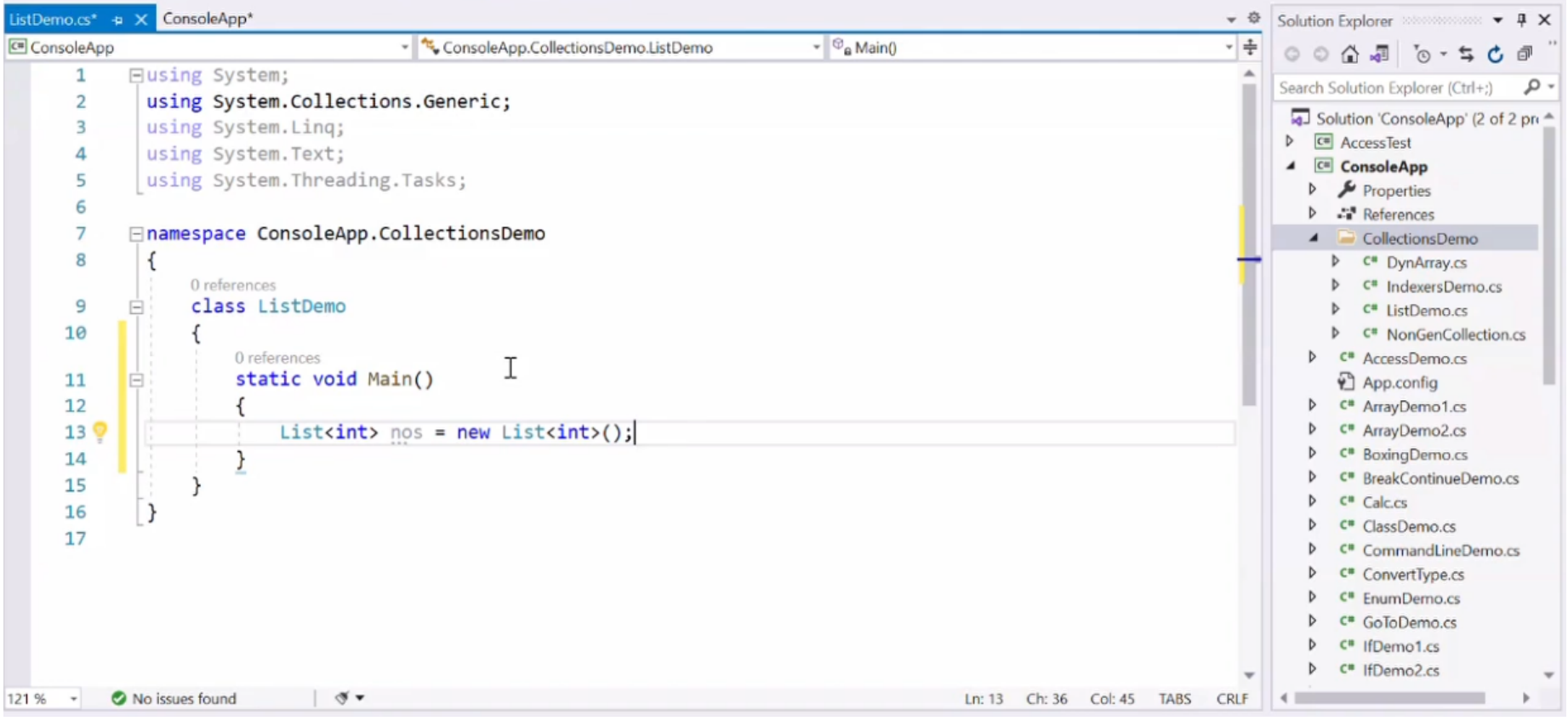
which is a part of collection



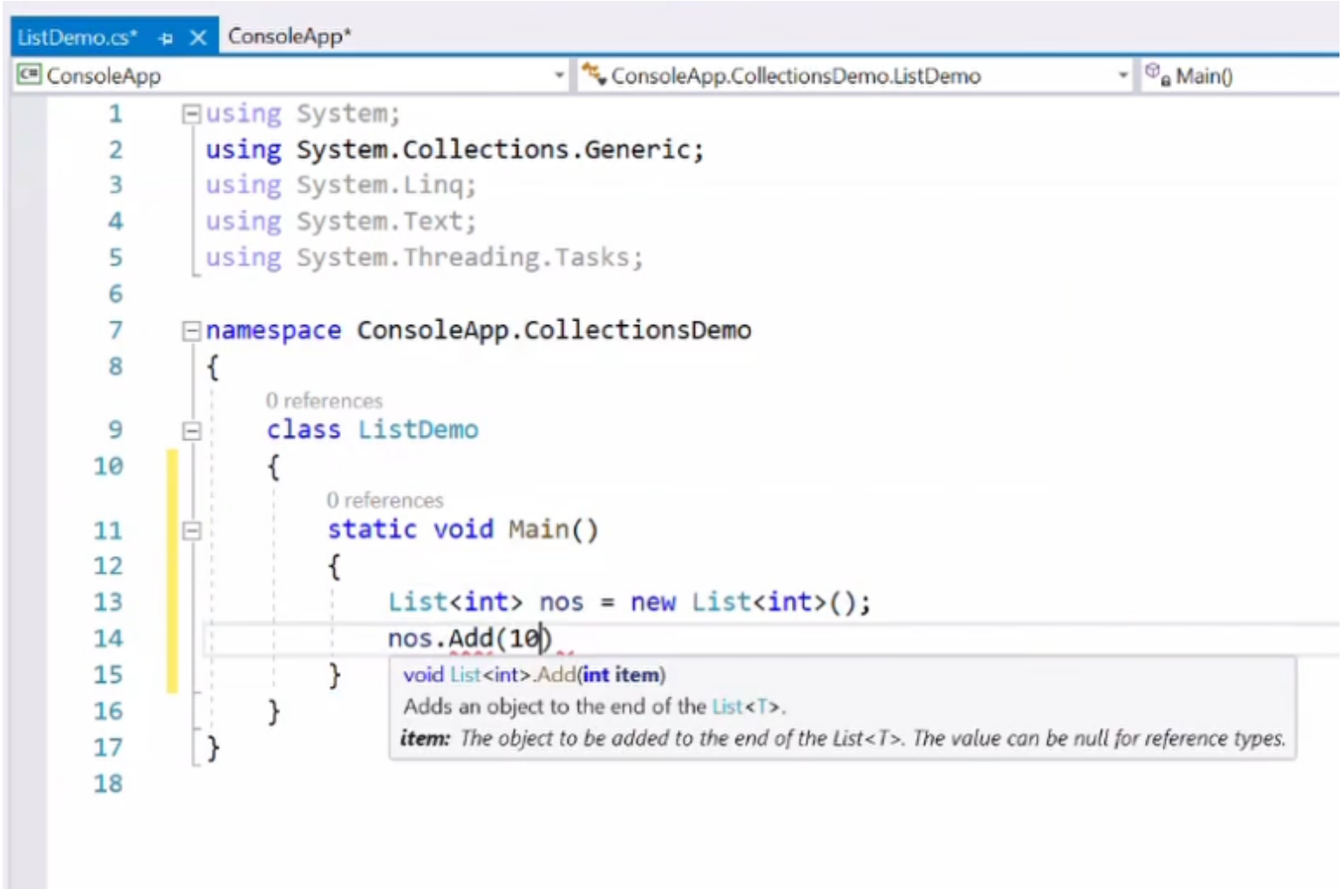
Collection.Generic

Now What is there in my Generic?

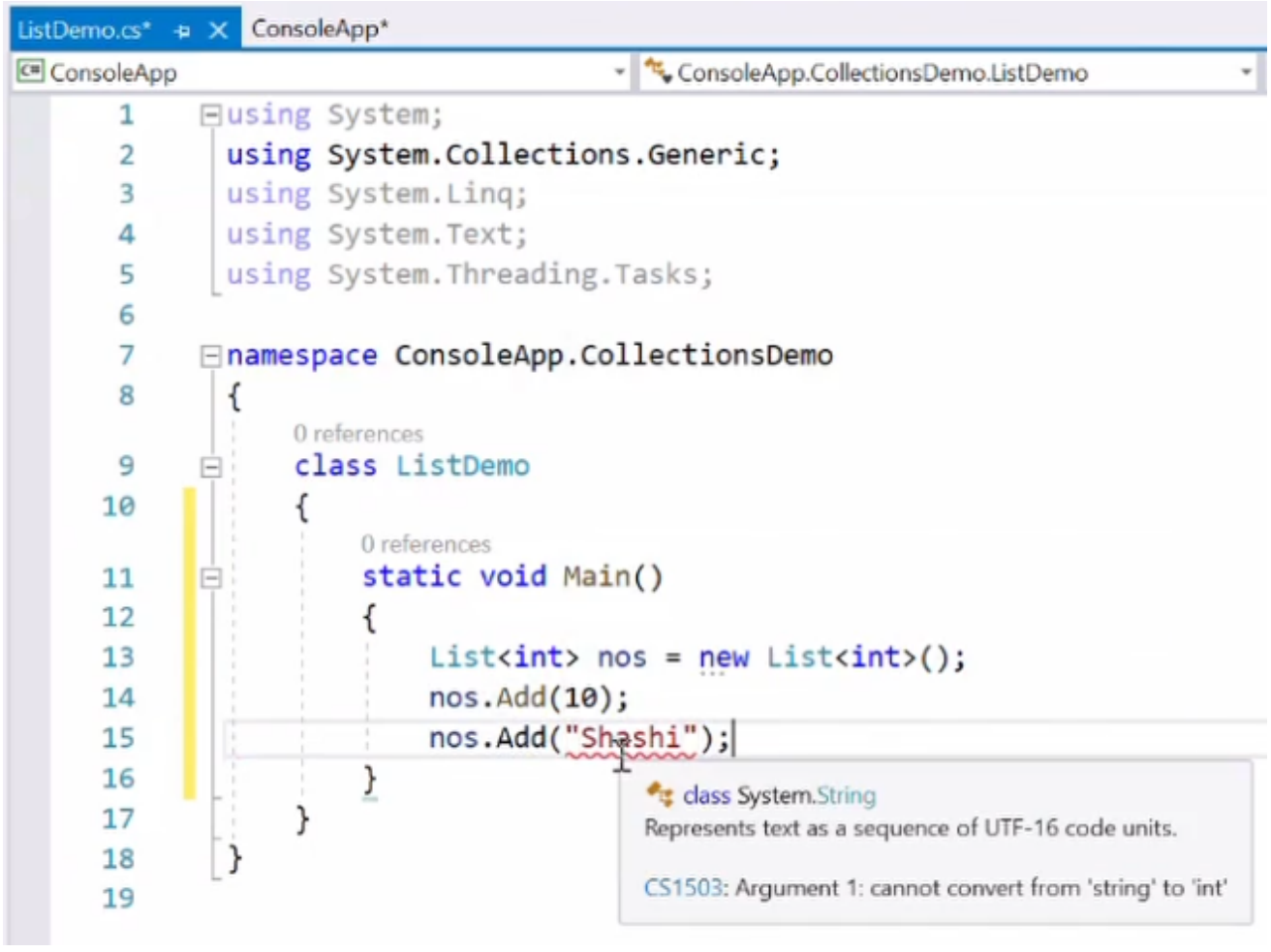
1. **List :** yesterday in trainer trainee app we used that , in the same way



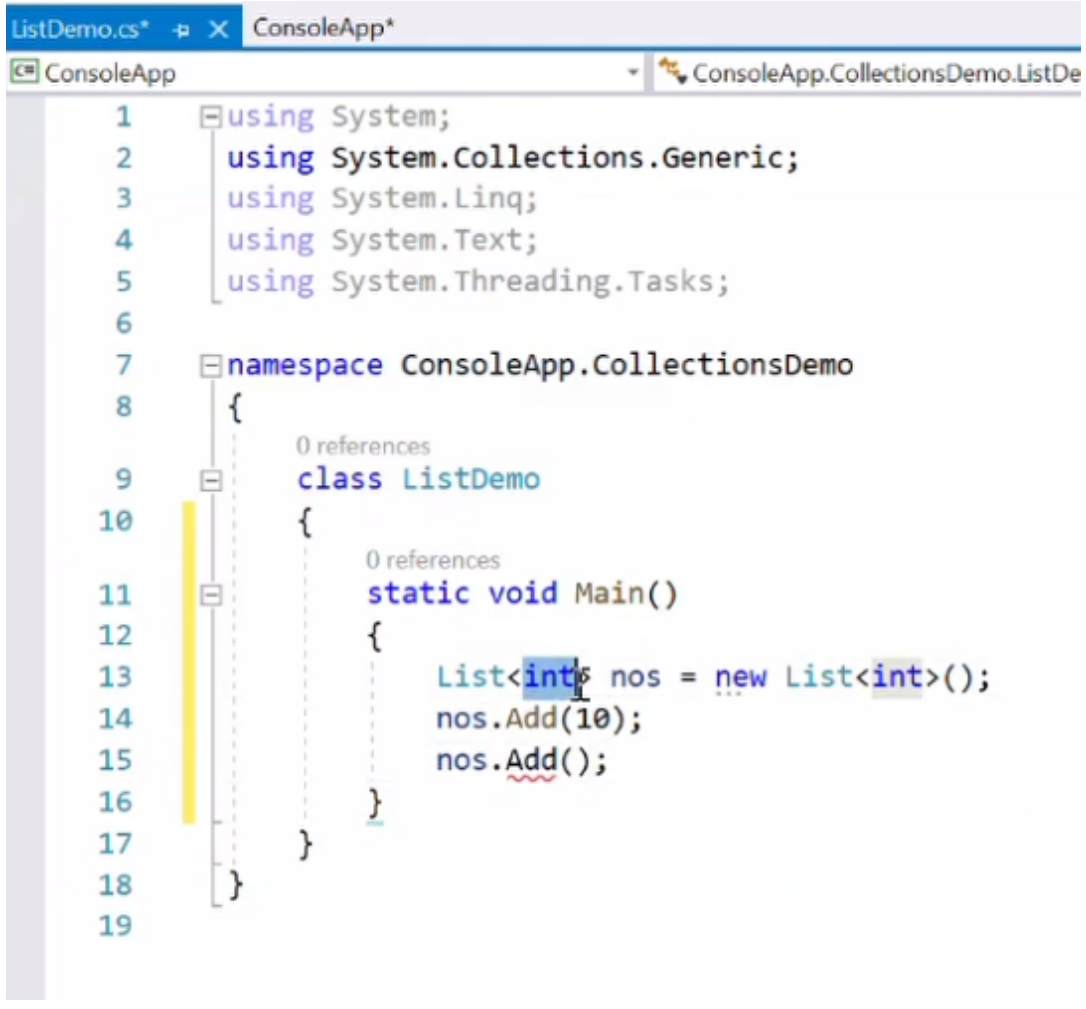
nos has a add method, look at this it takes only number type



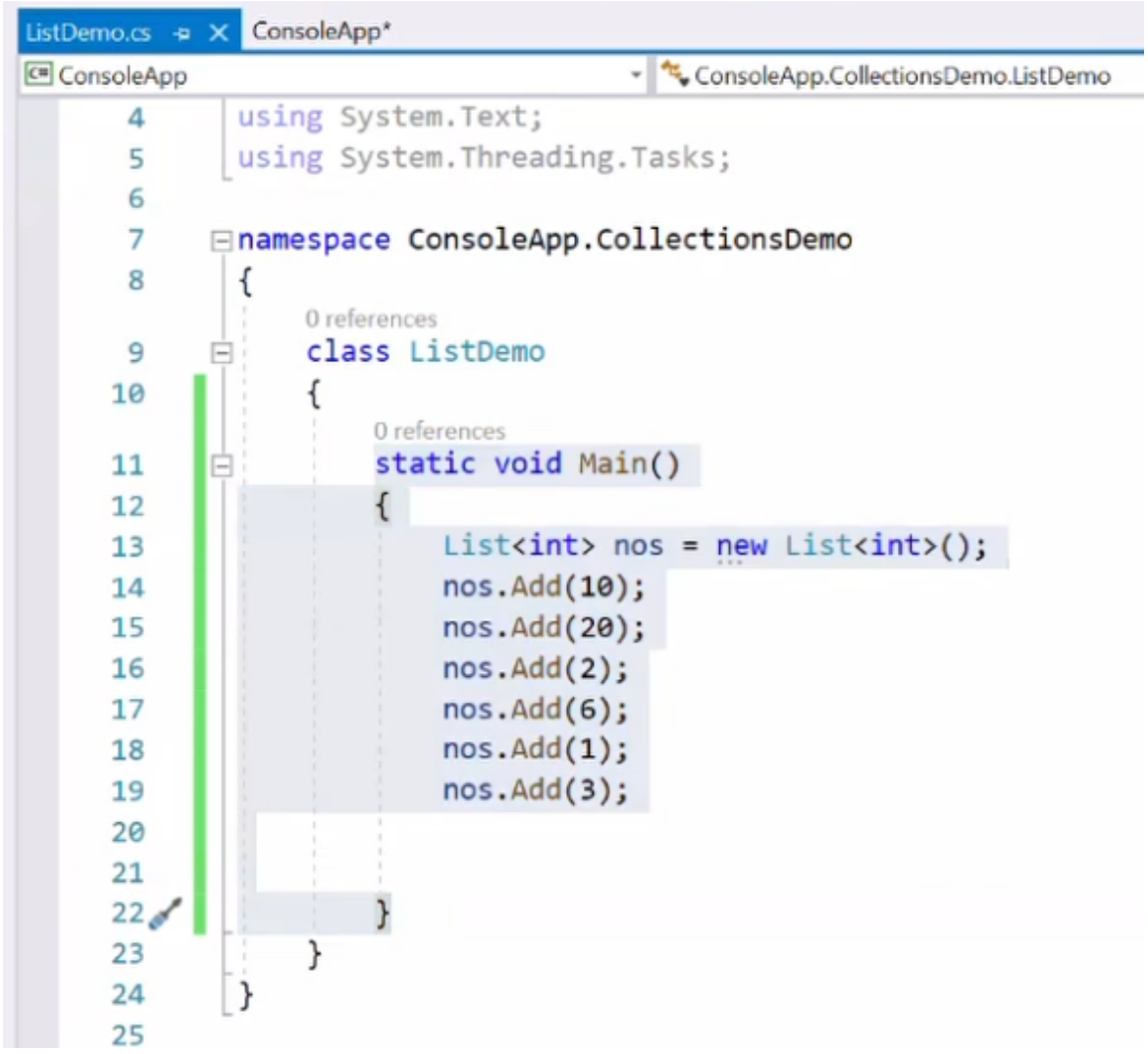
When i try to add some other data it gives me error



So this becomes a advantage of generics
it takes only the type which you pass here

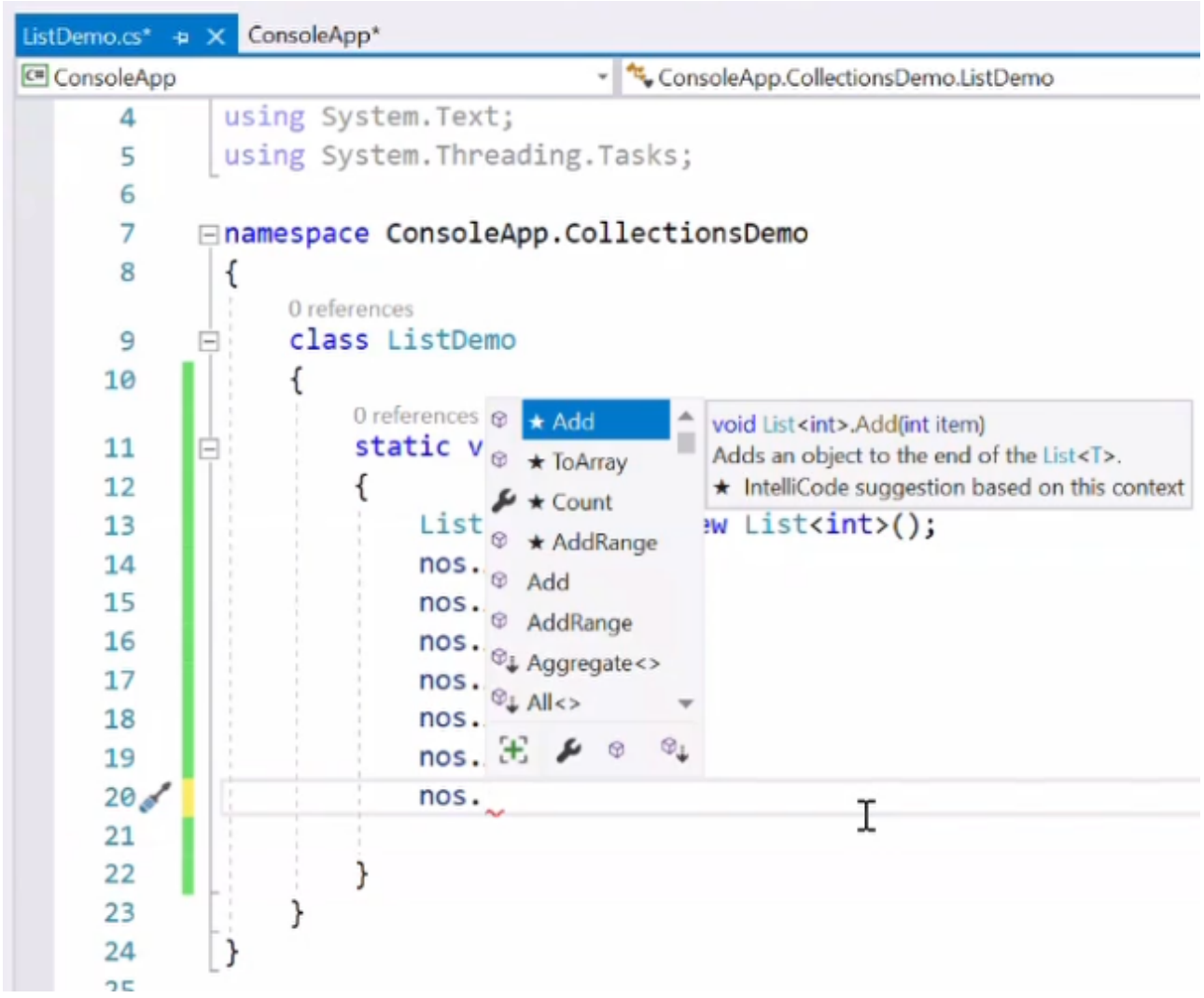


based on this it take only that data

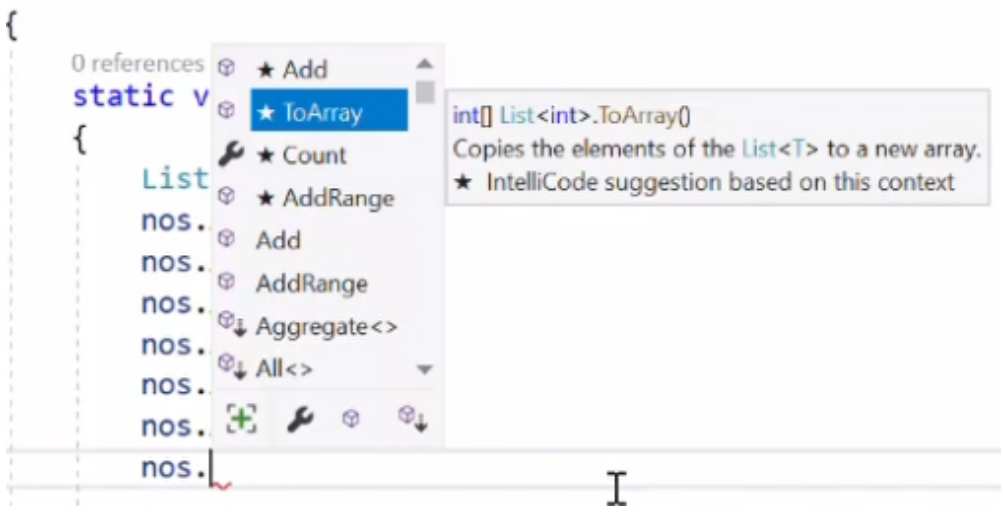


Explore the methods what list have

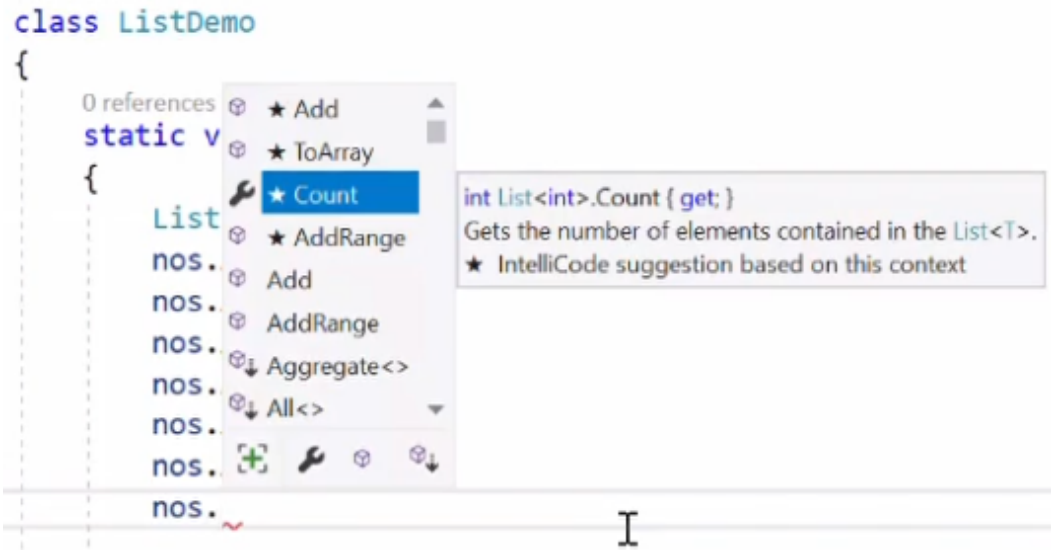
Just write nos. you will get all the methods what list have


ListDemo.cs*
ConsoleApp*
ConsoleApp
ConsoleApp.CollectionsDemo.ListDemo
using System.Text;
using System.Threading.Tasks;
namespace ConsoleApp.CollectionsDemo
{
0 references
class ListDemo
{
0 references
static v
{
List
ew List<int>();
nos.
nos.
nos.
nos.
nos.
nos.
nos.
}
}
}
★ Add
★ ToArray
★ Count
★ AddRange
Add
AddRange
Aggregate<>
All<>
void List<int>.Add(int item)
Adds an object to the end of the List<T>.
★ IntelliCode suggestion based on this context

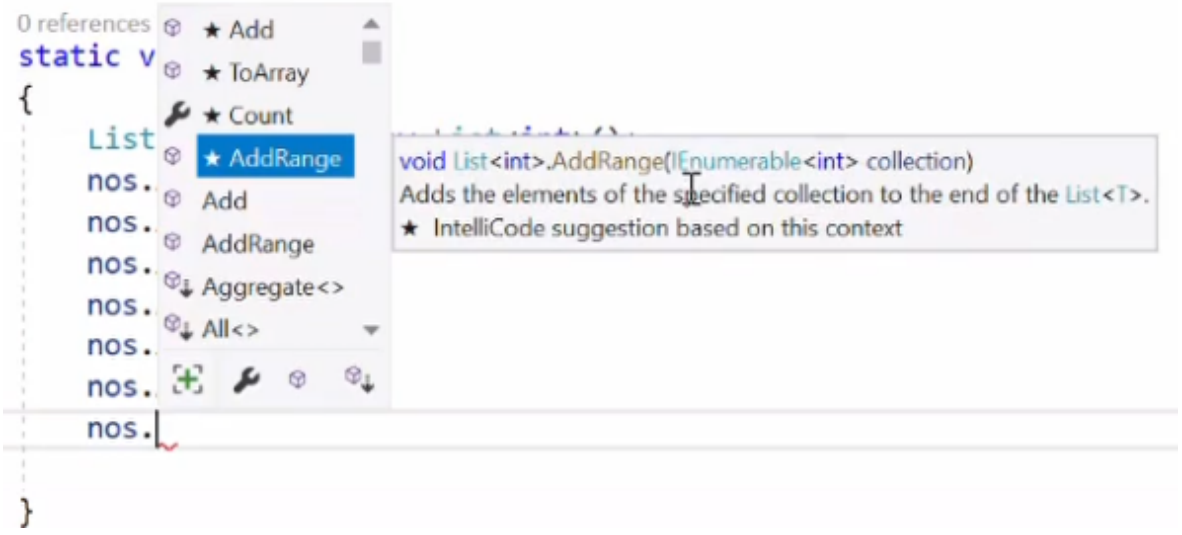
ToArray() --- it will gone to copy the elements of the list to a new array



Count() --- it will return me the length of the list, number of element in the list

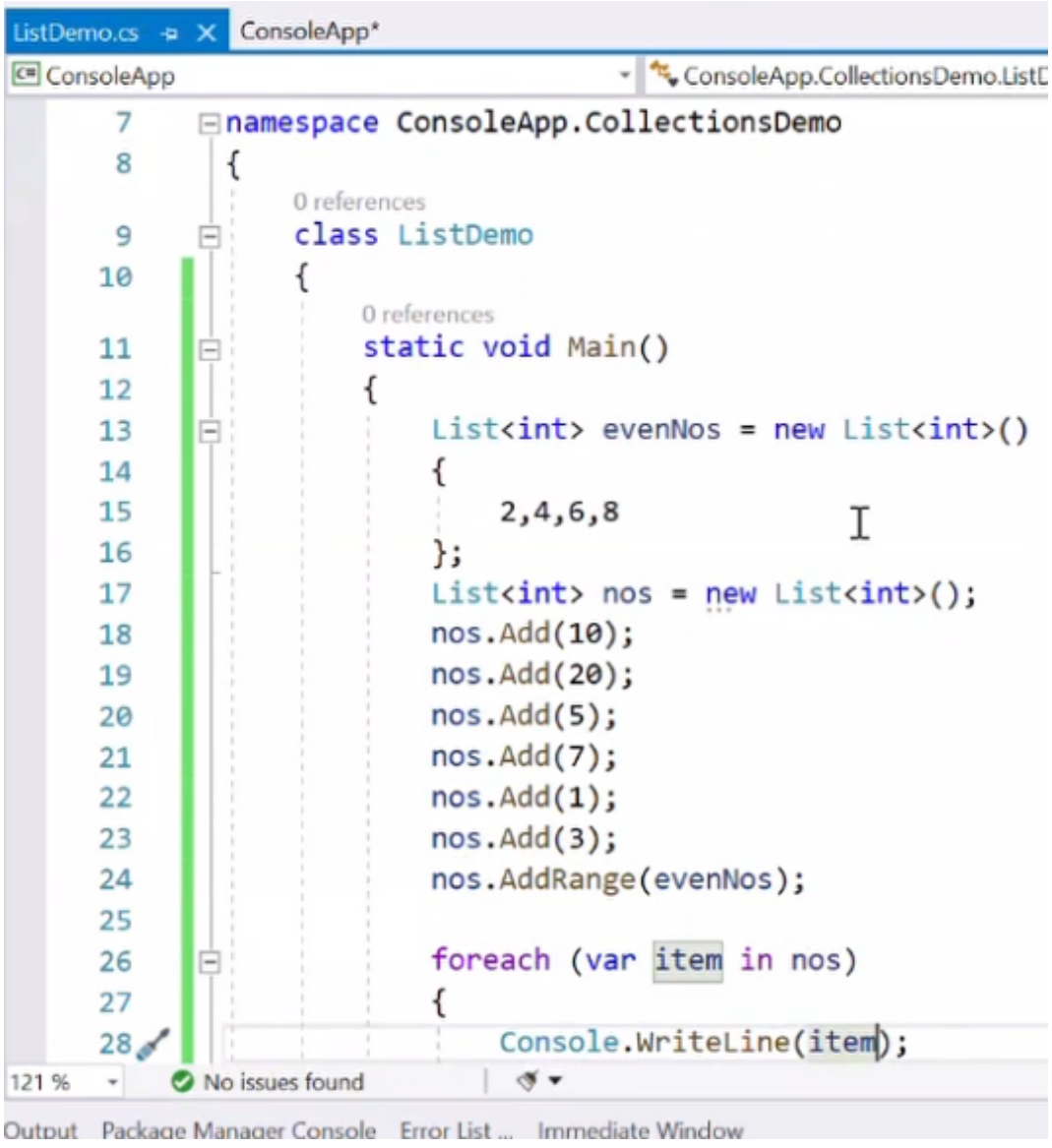


AddRange() --- it will take a existing array and it will add this existing array, look at the parameter what we are passing here, you need to pass collection as a parameter it adds the elements of the specified collection to the end of the list

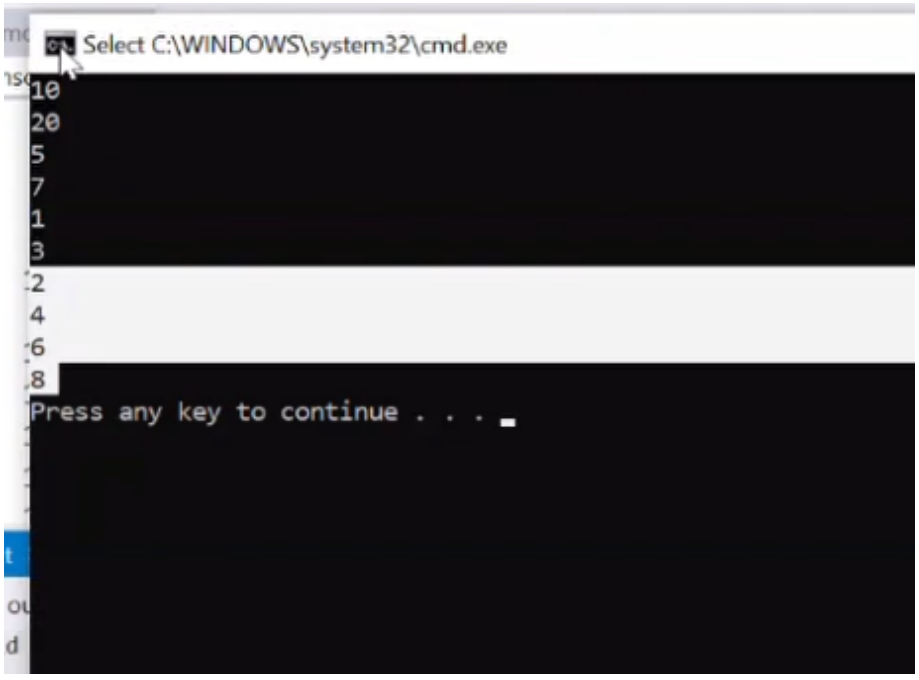


and it returns me nothing its a void type

I have a list of integers, I call it as evenNos, i have initialize it as 2 4 6 8, ill say addrange and pass evenNos


ListDemo.cs
ConsoleApp*
ConsoleApp
ConsoleApp.CollectionsDemo.ListD
namespace ConsoleApp.CollectionsDemo
{
0 references
class ListDemo
{
0 references
static void Main()
{
List<int> evenNos = new List<int>()
{
2,4,6,8
};
List<int> nos = new List<int>();
nos.Add(10);
nos.Add(20);
nos.Add(5);
nos.Add(7);
nos.Add(1);
nos.Add(3);
nos.AddRange(evenNos);
foreach (var item in nos)
{
Console.WriteLine(item);
121 %
No issues found
Output
Package Manager Console
Error List ...
Immediate Window

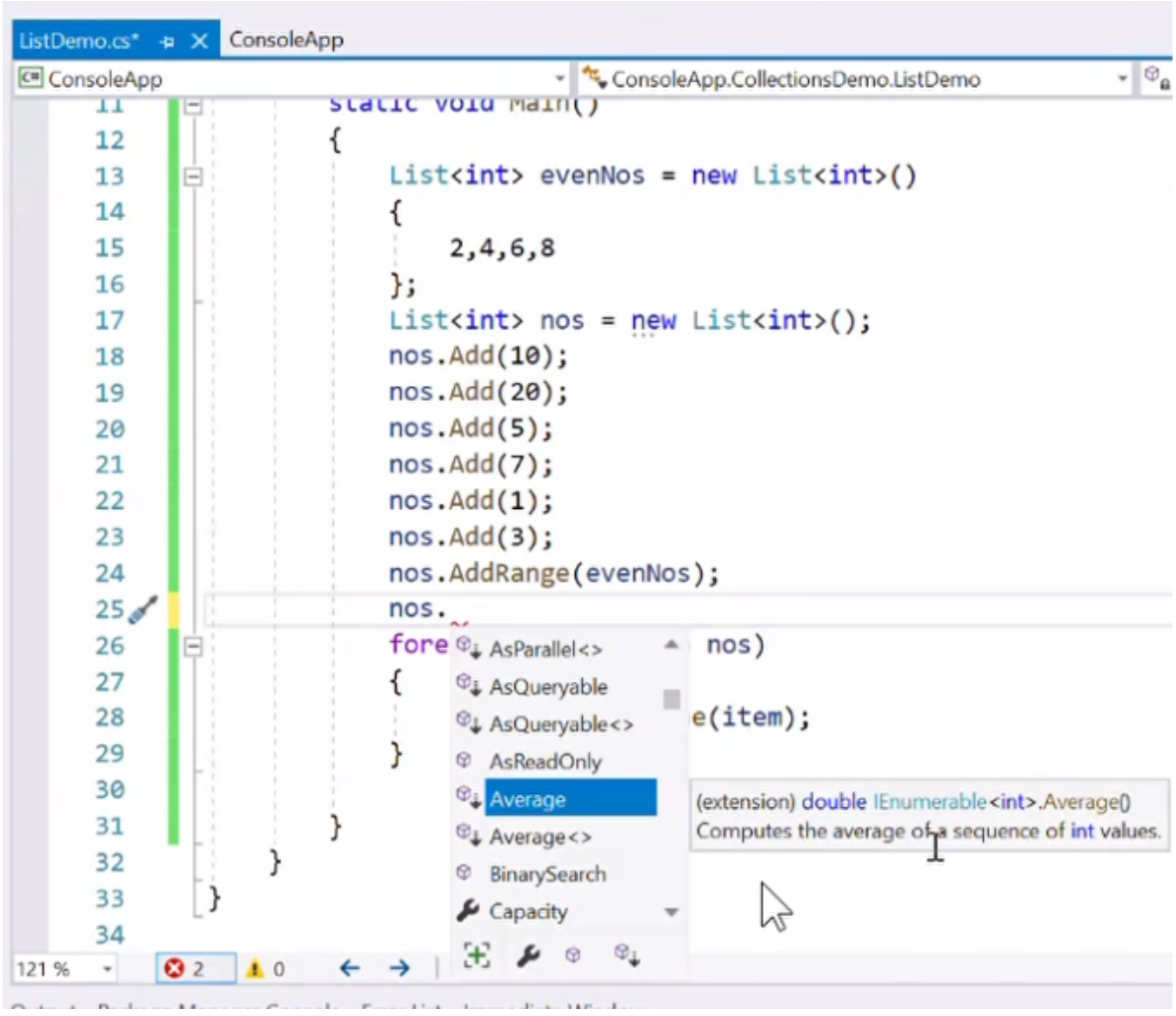
and print it using foreach



Clear() --- it removes all the elements from the list

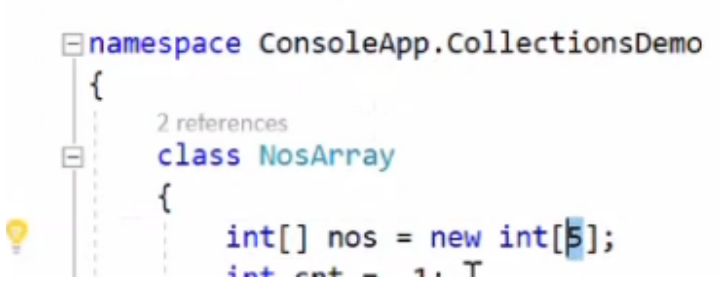
---

similarly look for other things like **average**, **Binarysearch** and all binary search you cannot run directly, you need I Comparer for it

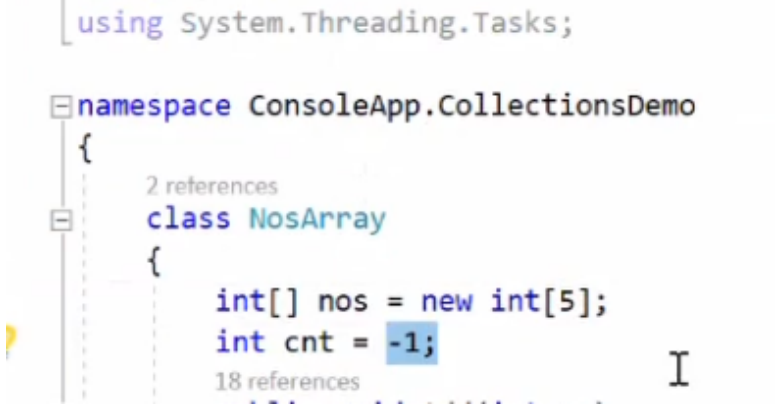

ListDemo.cs*
ConsoleApp
ConsoleApp
ConsoleApp.CollectionsDemo.ListDemo
static void Main()
{
List<int> evenNos = new List<int>()
{
2,4,6,8
};
List<int> nos = new List<int>();
nos.Add(10);
nos.Add(20);
nos.Add(5);
nos.Add(7);
nos.Add(1);
nos.Add(3);
nos.AddRange(evenNos);
nos.
fore
nos)
{
e(item);
}
}
}
}
AsParallel<>
AsQueryable
AsQueryable<>
AsReadOnly
Average
Average<>
BinarySearch
Capacity
(extension) double IEnumerable<int>.Average()
Computes the average of a sequence of int values.
121 %
2
0

What is **Count** and **Capacity**

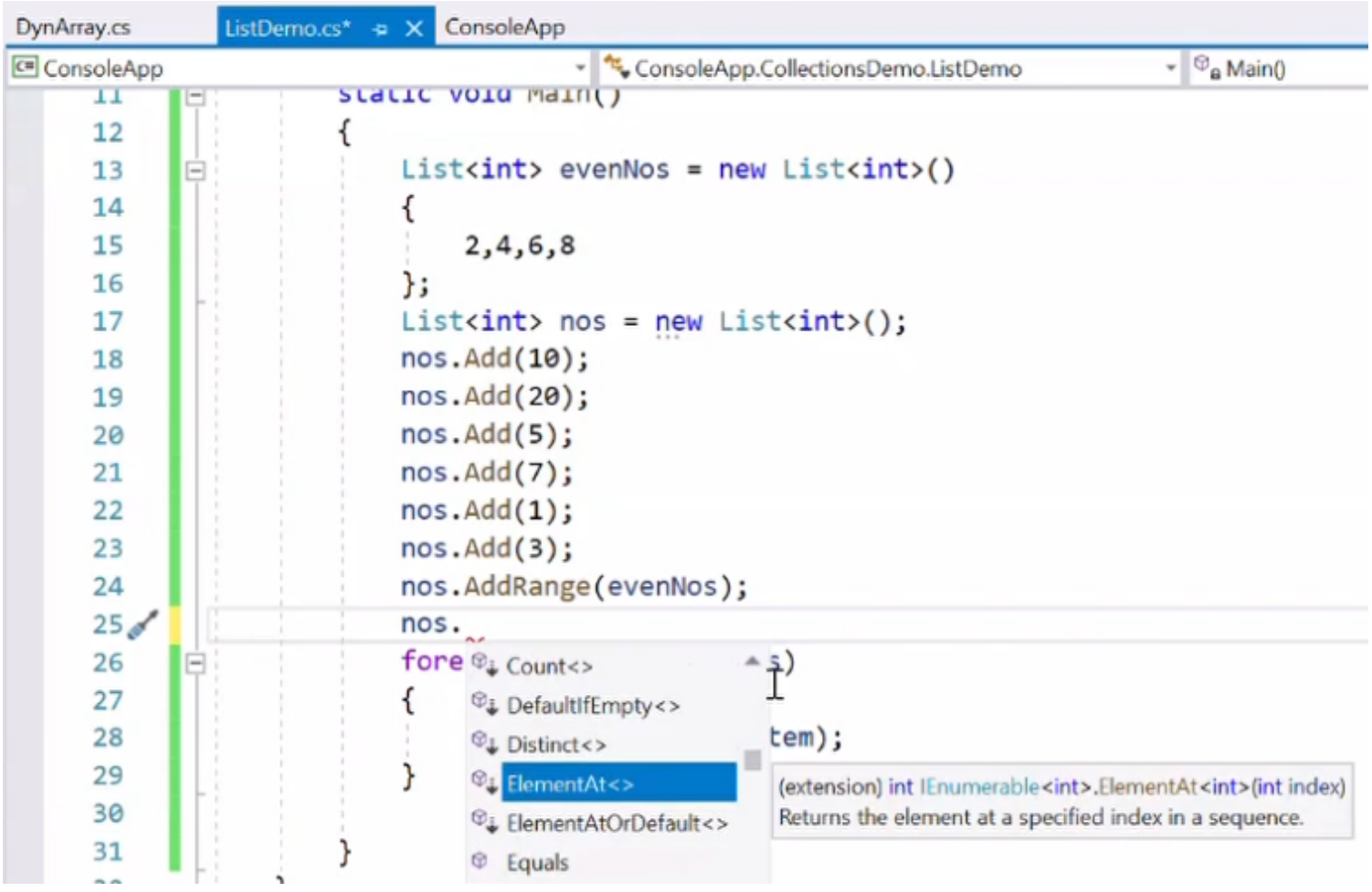
this 5 is my capacity here



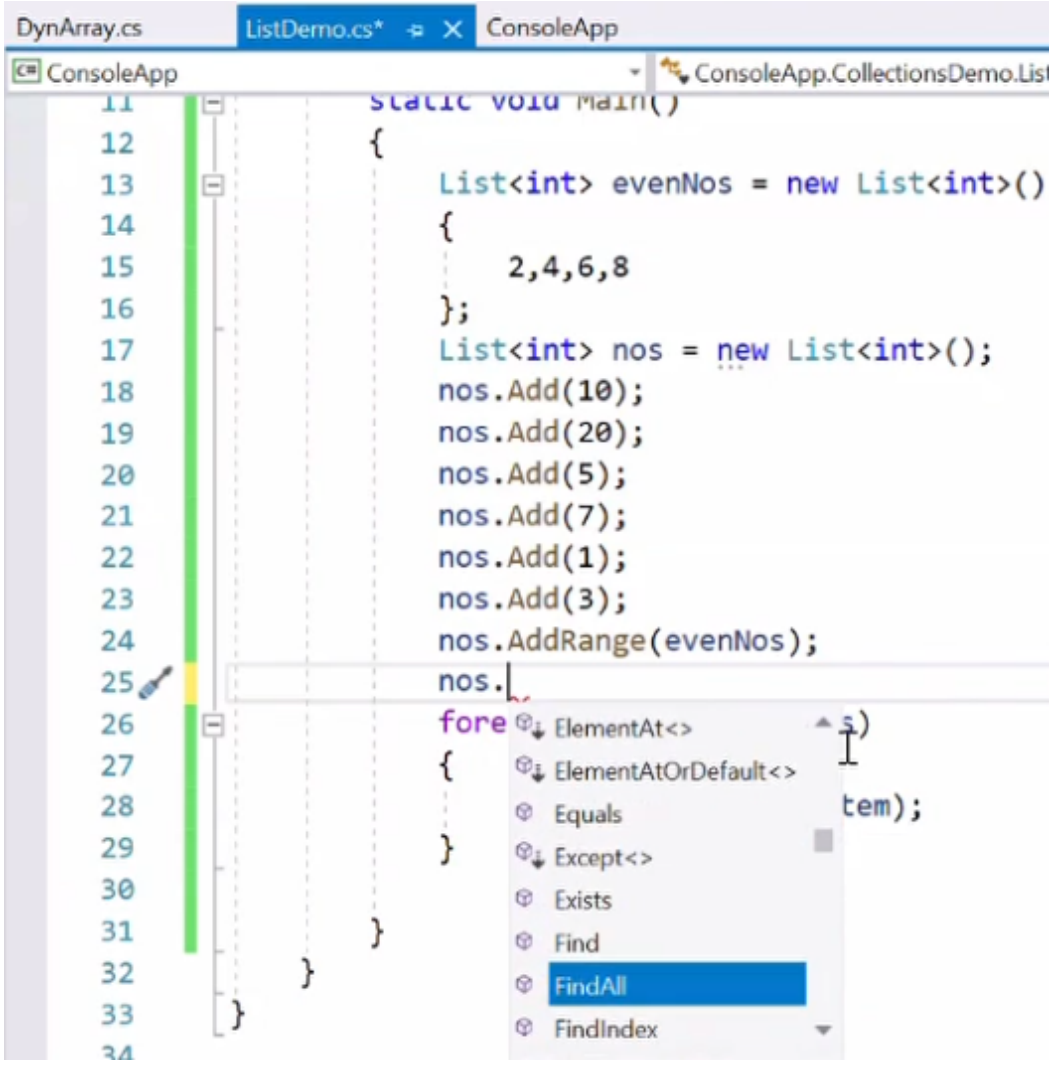
and this -1 is count



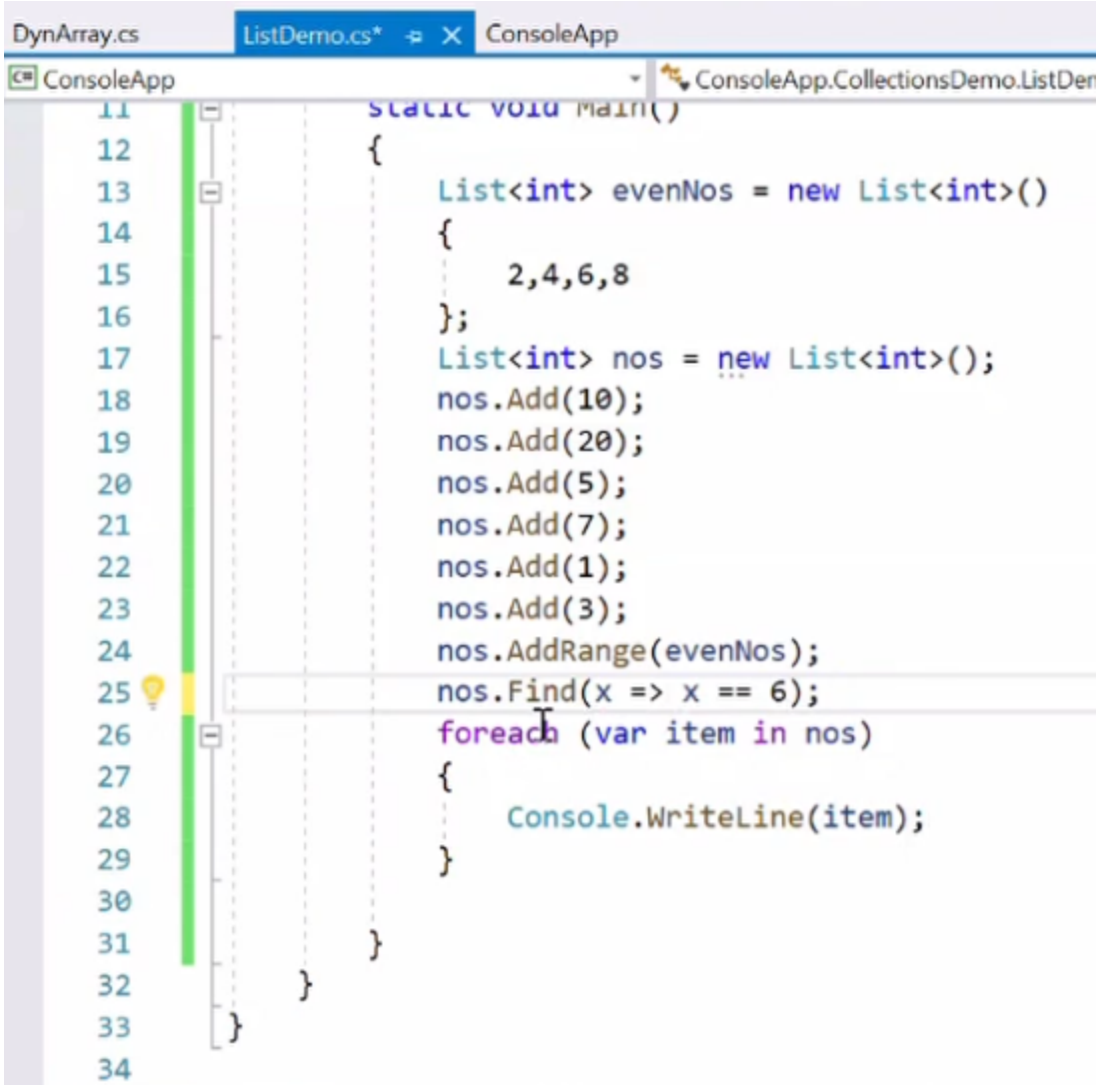
Element at ---- I want to know what is the element at a given position



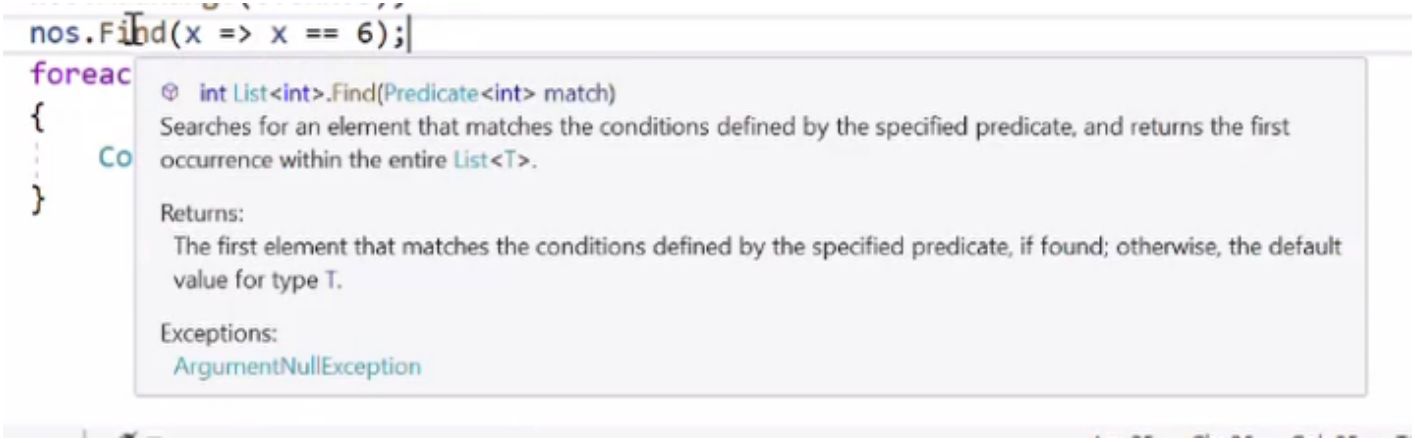
Find, FindAll -----



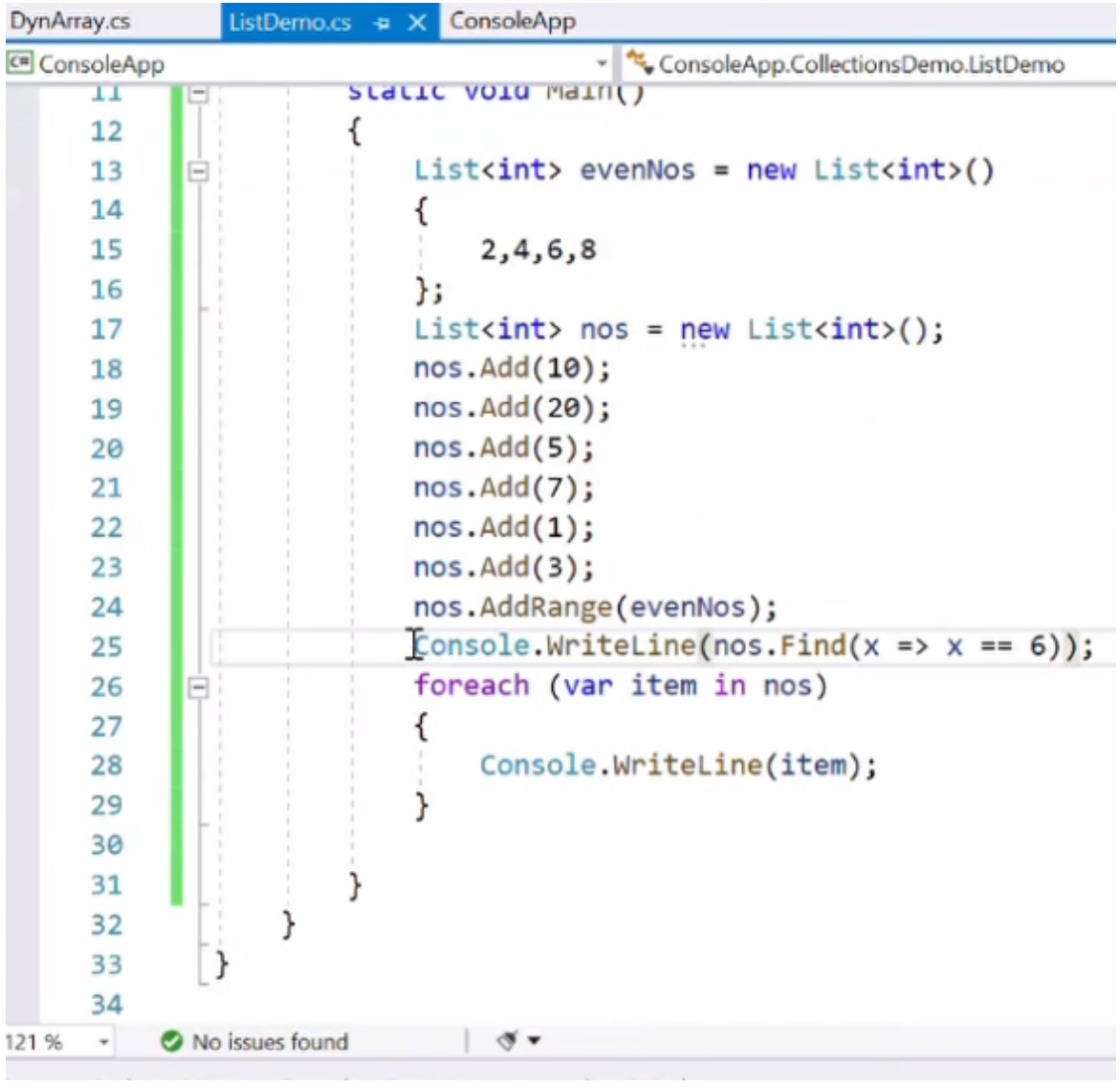
you can write predicate like this **x goes to x == i want to find 6**



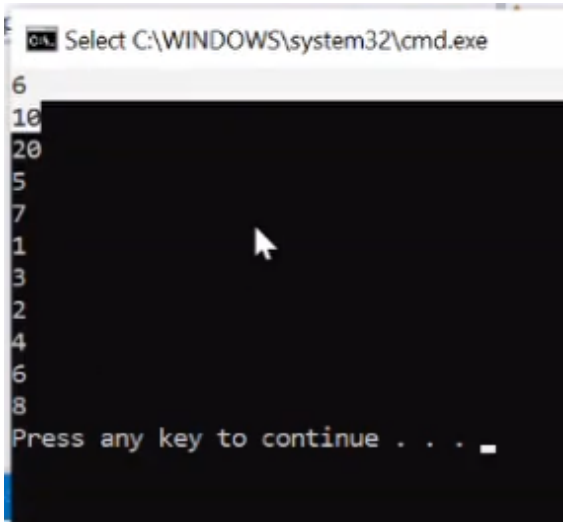
now what my **find** does is, it will return me a integer



you can put that in Console.WriteLine

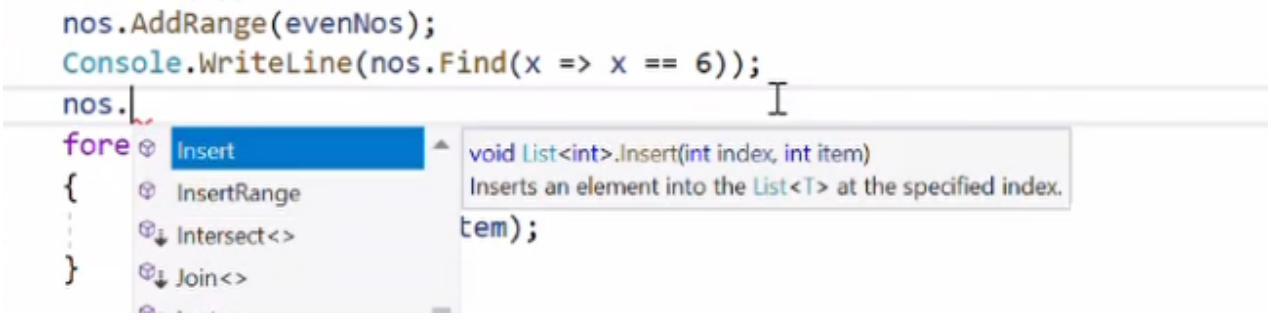


it will give me that particular number



Where ever you have the predicate, you can always check for the predicate, this will return weather the particular number exists or not

you have **insert**, **insertrange** ...



in Array list this will be very difficult, going to index and adding the particular value

you have **max**--- it will give me maximum number in the list

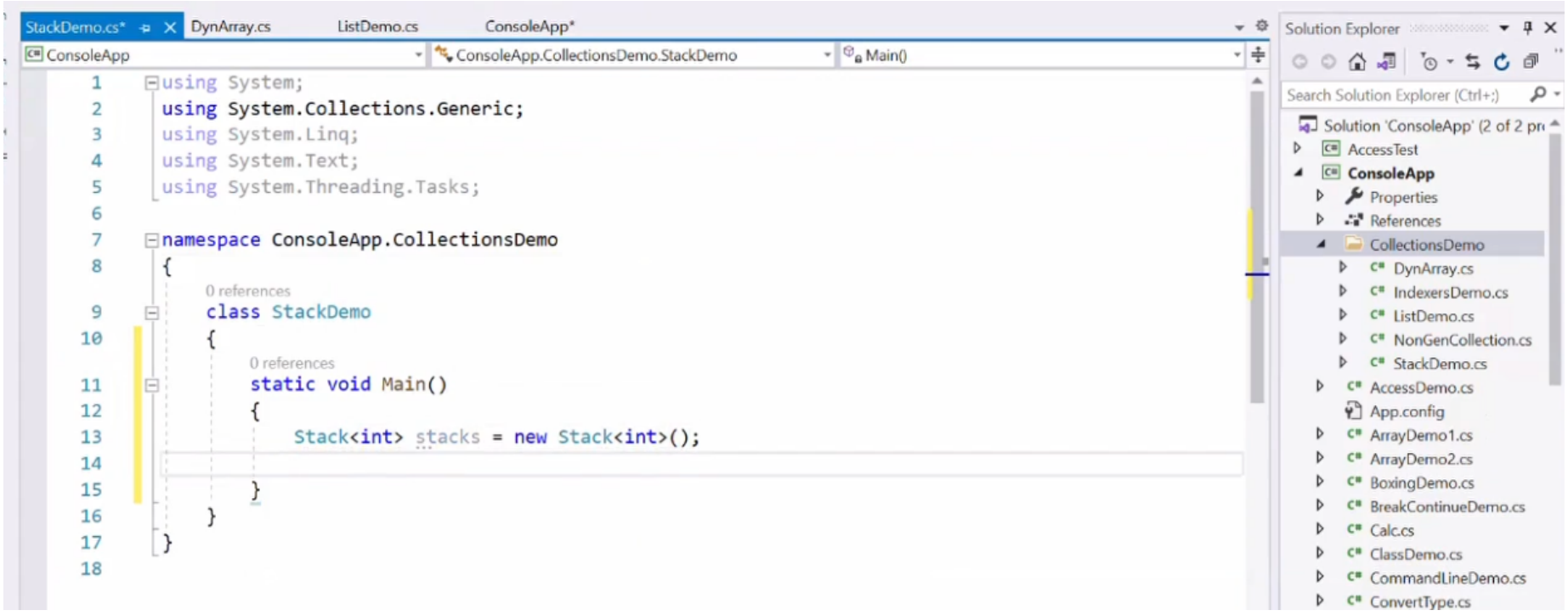
, **min** --- will give me the minimum number in the list

you have **Remove**, **Remove at** you can go to particular position and you can remove element from that position
you have **Reverse** ---------it will reverse the whole list
Sort --- it will sort the list
Sum --- it will sum all the list

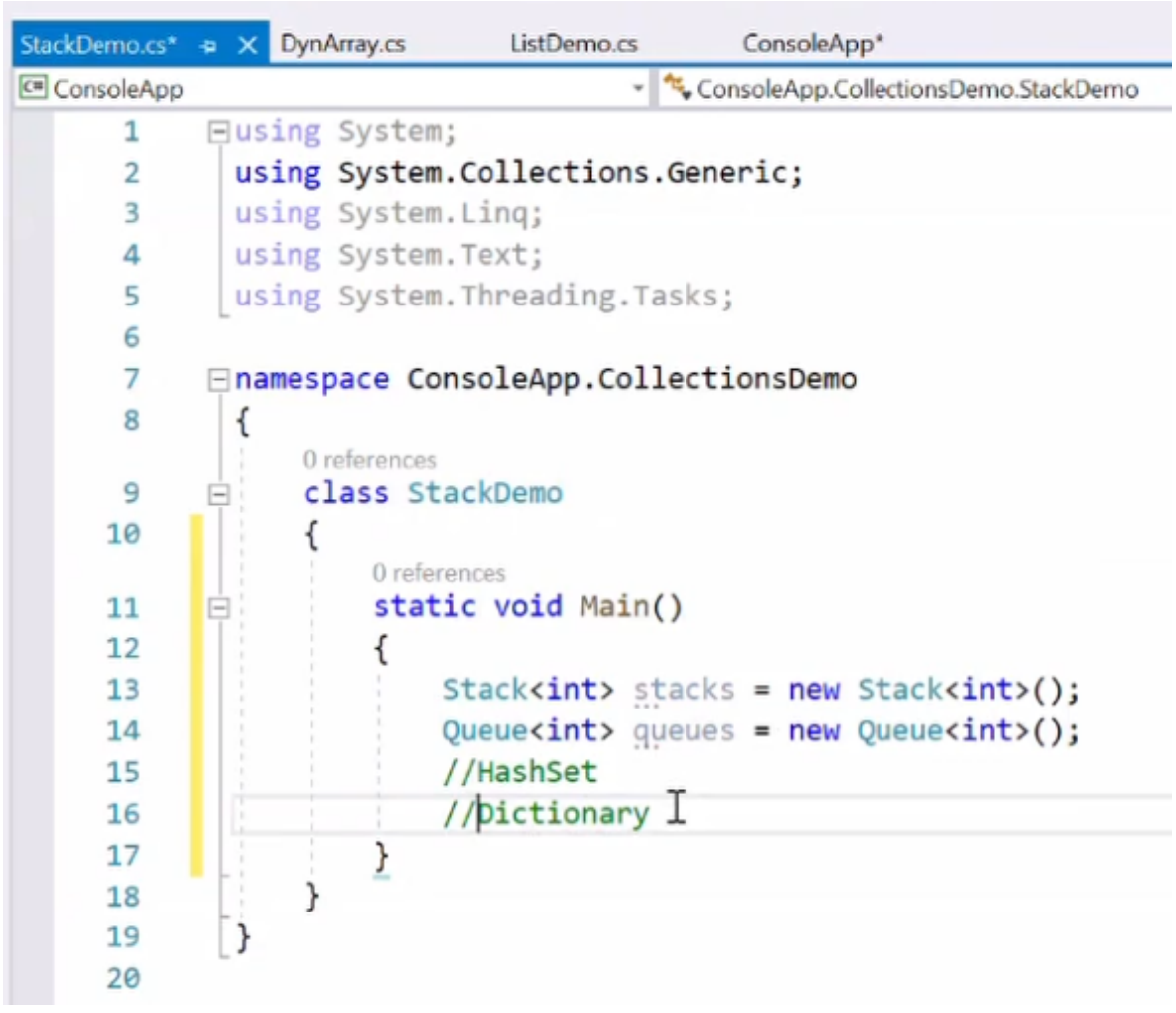
so this are some of the methods of list, you can explore more like this

Stack

Create a new stack, checkout for the methods

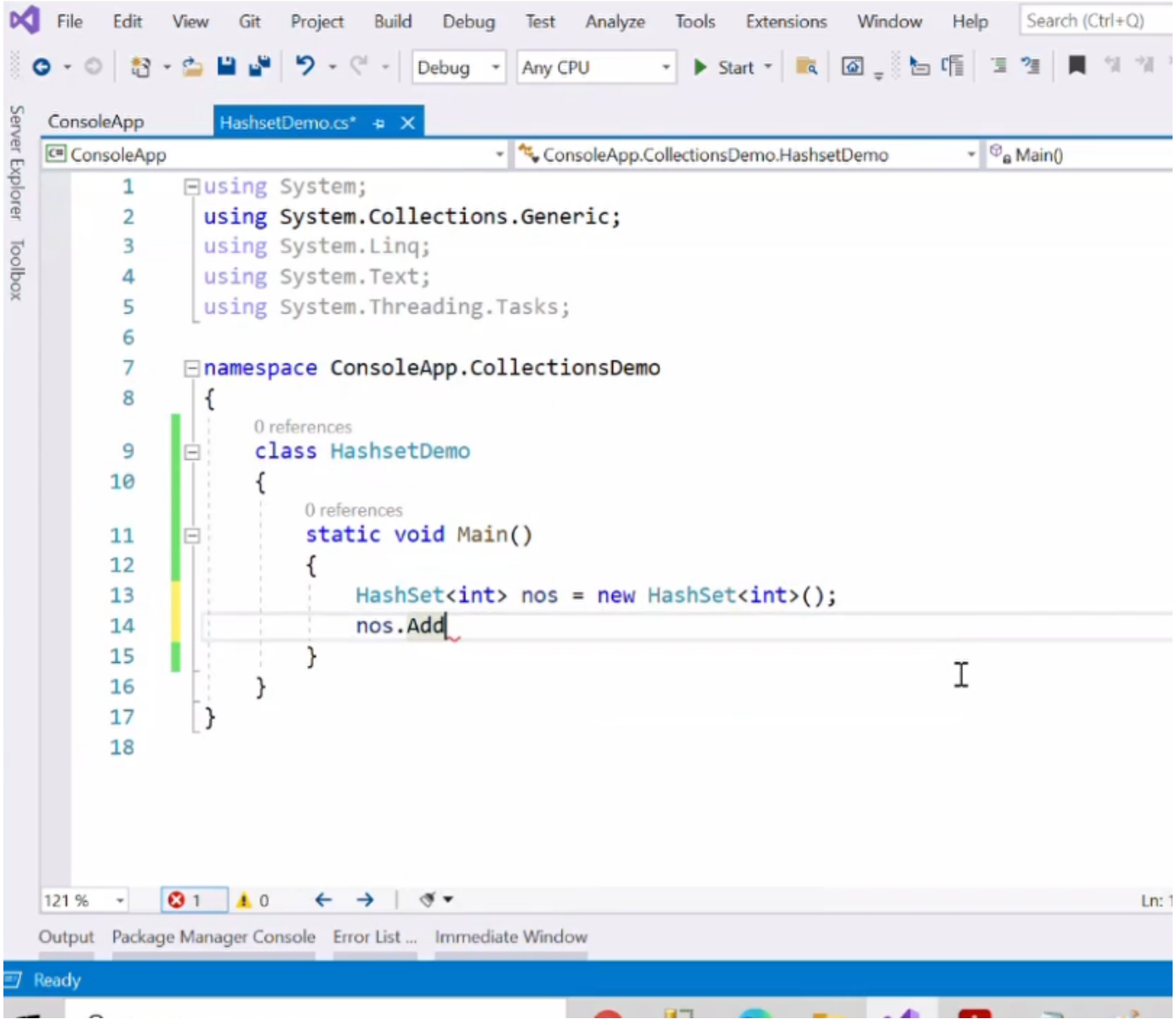


Similarly check for **Queue**

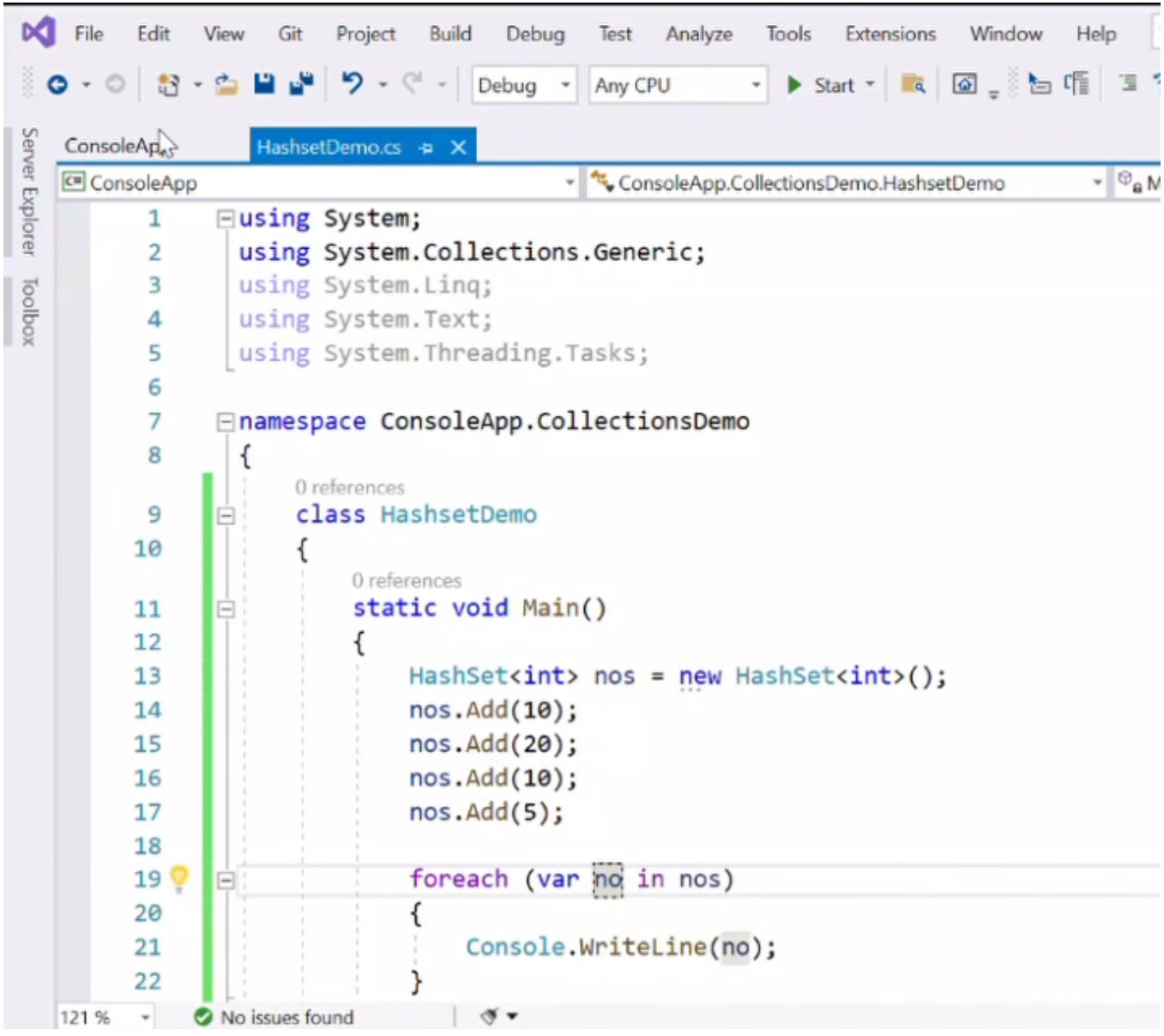

StackDemo.cs*
DynArray.cs
ListDemo.cs
ConsoleApp*
ConsoleApp
ConsoleApp.CollectionsDemo.StackDemo
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace ConsoleApp.CollectionsDemo
{
0 references
class StackDemo
{
0 references
static void Main()
{
Stack<int> stacks = new Stack<int>();
Queue<int> queues = new Queue<int>();
//HashSet
//Dictionary
}
}
}

HashSet

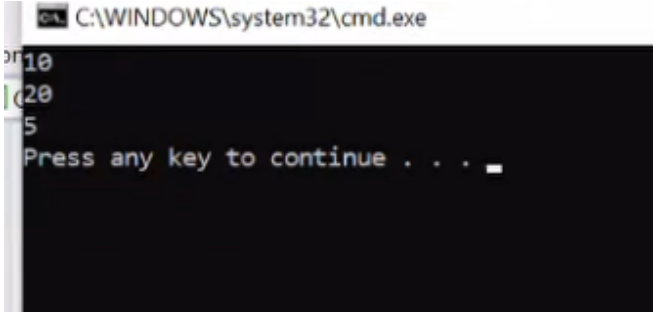
Now see If I don't know about hashset
just like any other collection i'll create a hashset and try methods on it, like what methods it will have


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search (Ctrl+Q)
Debug
Any CPU
Start
ConsoleApp
HashsetDemo.cs*
ConsoleApp
ConsoleApp.CollectionsDemo.HashsetDemo
Main()
Server Explorer
Toolbox
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace ConsoleApp.CollectionsDemo
8 {
0 references
9 class HashsetDemo
10 {
0 references
11 static void Main()
12 {
13 HashSet<int> nos = new HashSet<int>();
14 nos.Add
15 }
16 }
17 }
18
121 %
1
0
Ln:
Output
Package Manager Console
Error List ...
Immediate Window
Ready

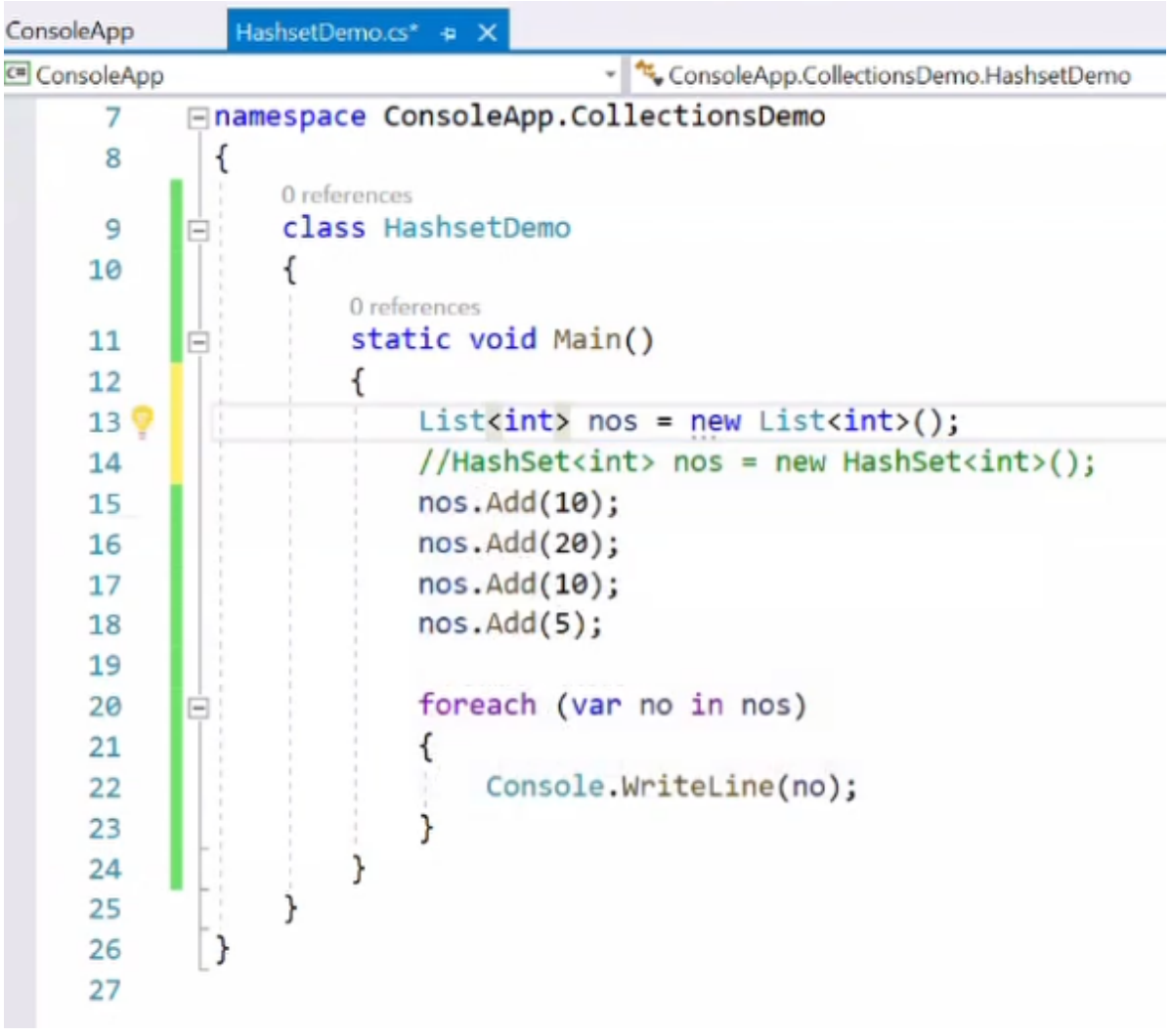
my hashset will also support add method


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Debug
Any CPU
Start
Server Explorer
Toolbox
ConsoleApp
HashsetDemo.cs
ConsoleApp
ConsoleApp.CollectionsDemo.HashsetDemo
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace ConsoleApp.CollectionsDemo
{
0 references
class HashsetDemo
{
0 references
static void Main()
{
HashSet<int> nos = new HashSet<int>();
nos.Add(10);
nos.Add(20);
nos.Add(10);
nos.Add(5);
foreach (var no in nos)
{
Console.WriteLine(no);
}
121 %
No issues found

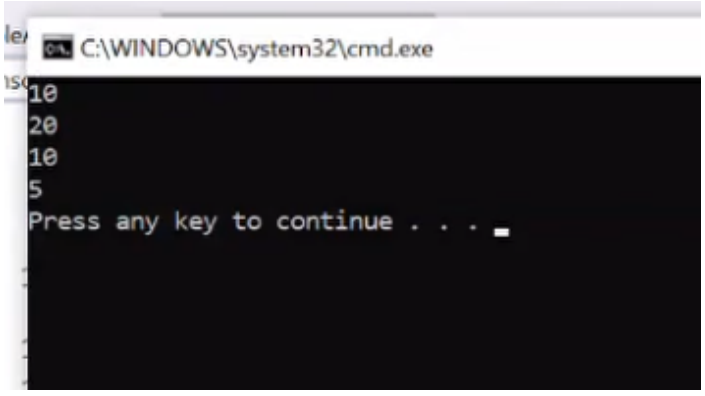
the expected output is 10 20 10 5
but i got this



So instead of me creating hashset lets check with list



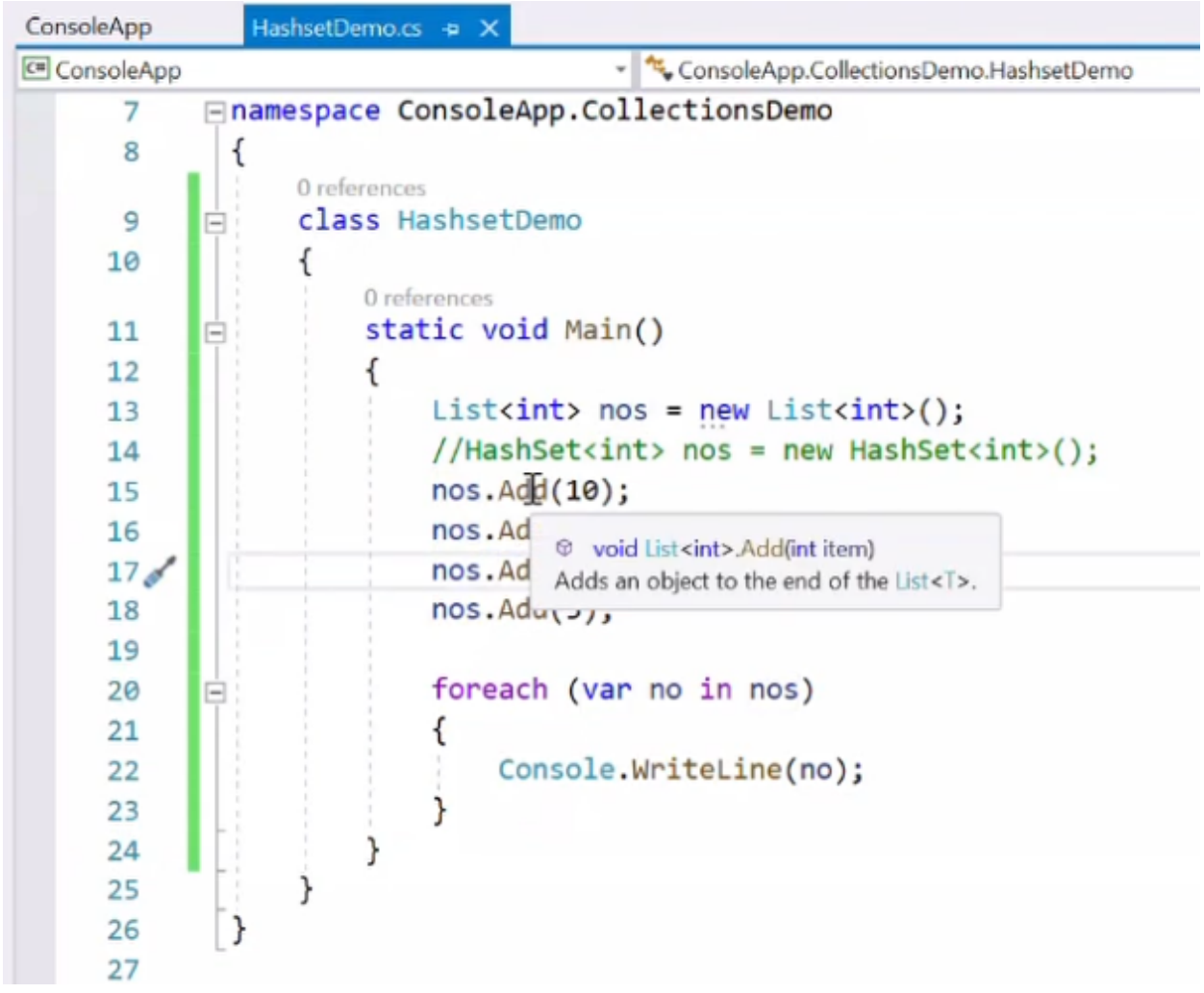
now i'm getting expected output



But in Hashset why am i not getting 10 twice

because hashset is a set of unique element, So it will always capture the unique data

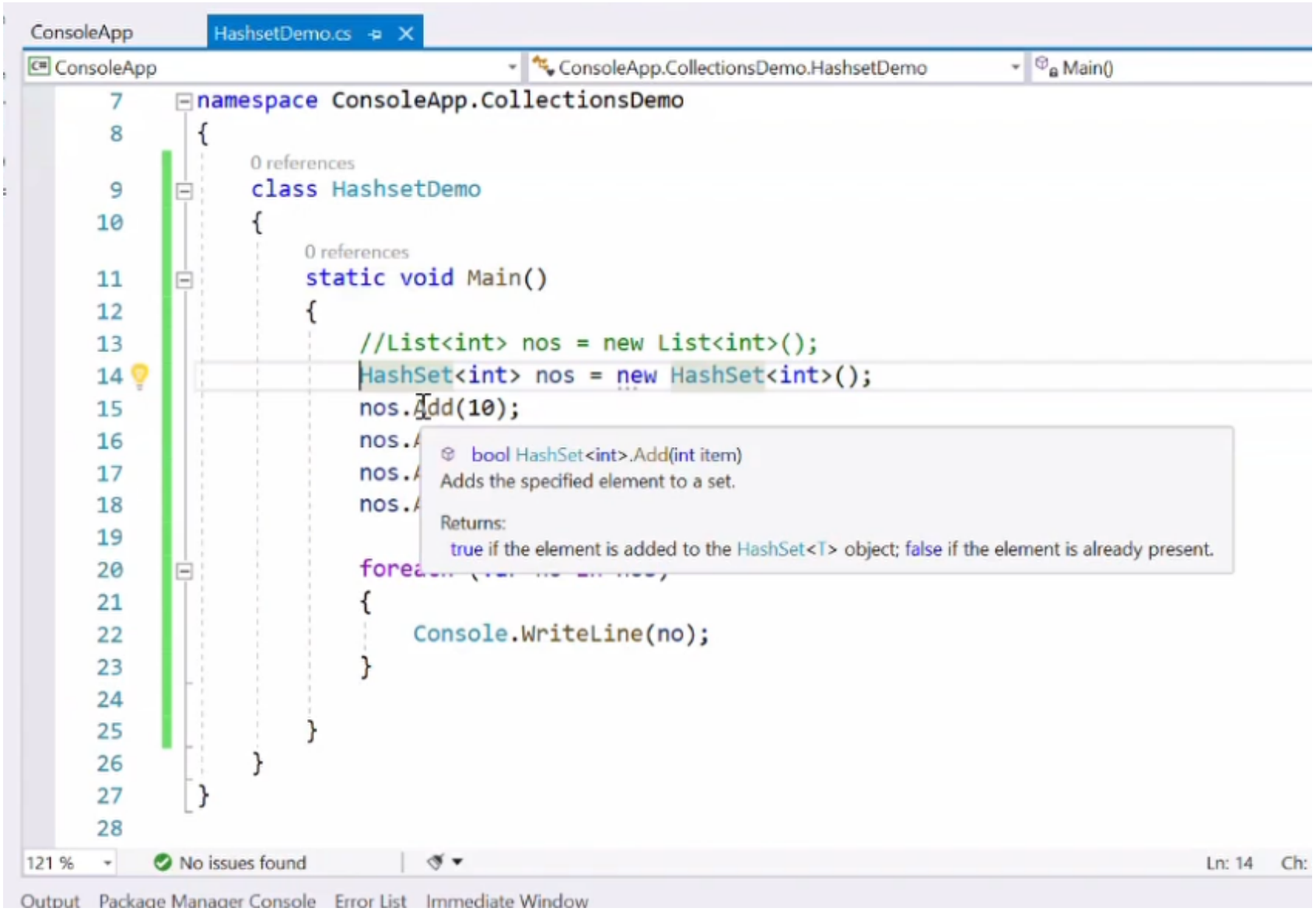
So how do you know 10 is stored twice or not
So look at my add method



in list add method is of void type, so you will not know weather it has added the particular element or not, so it will be difficult for you to know weather the particular element is added or not

so it's difficult to know with list

but look at this with hashset


ConsoleApp
HashsetDemo.cs
ConsoleApp
ConsoleApp.CollectionsDemo.HashsetDemo
Main()
namespace ConsoleApp.CollectionsDemo
{
0 references
class HashsetDemo
{
0 references
static void Main()
{
//List<int> nos = new List<int>();
HashSet<int> nos = new HashSet<int>();
nos.Add(10);
bool HashSet<int>.Add(int item)
Adds the specified element to a set.
Returns:
true if the element is added to the HashSet<T> object; false if the element is already present.
{
Console.WriteLine(no);
}
}
}
}
121 %
No issues found
Ln: 14
Ch:
Output
Package Manager Console
Error List
Immediate Window

with hashset add is of boolean type, that's how you will know weather the element is added or not in the hashset So you can always put a message,it is not added or it's a duplicate element

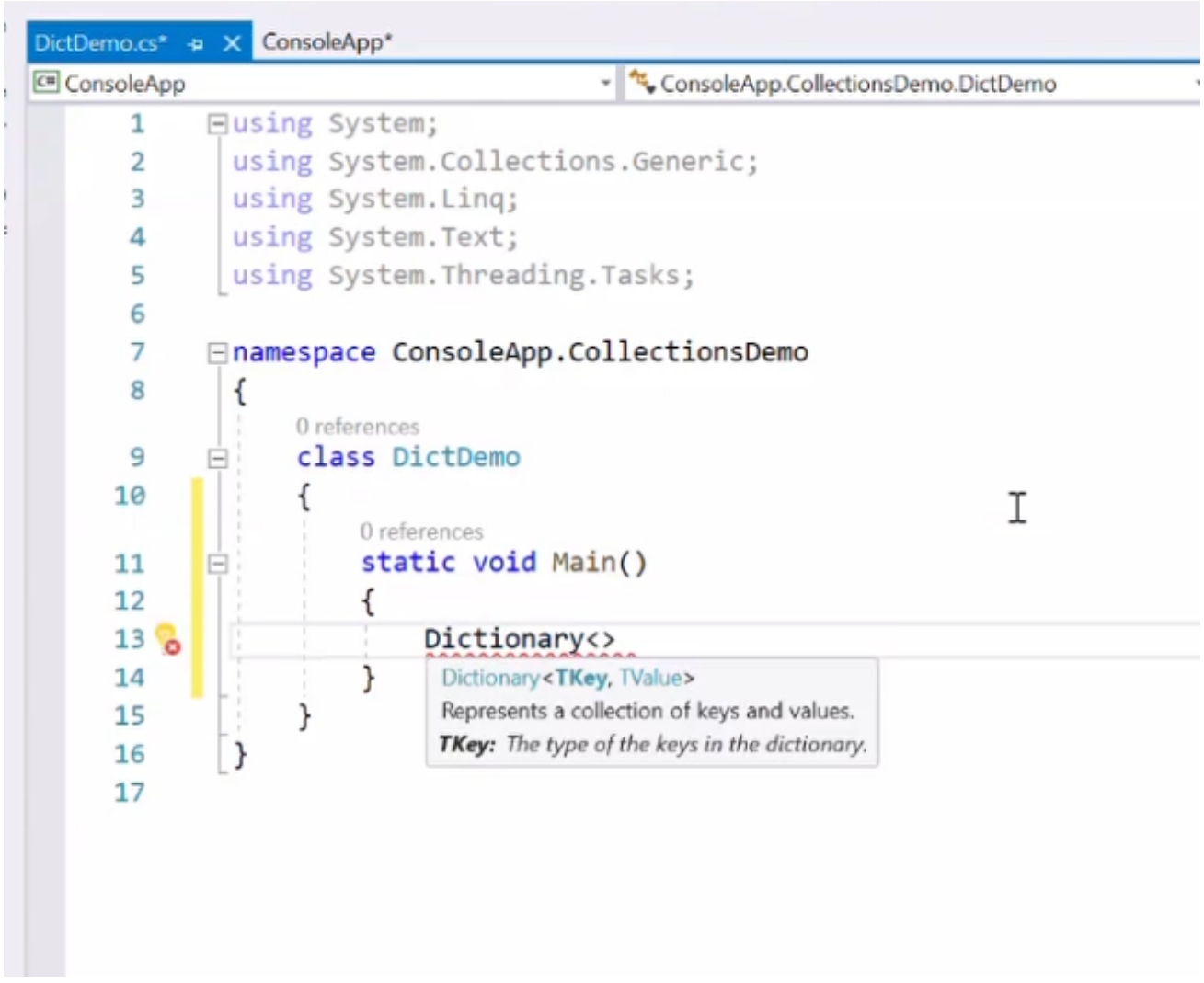
Apart from this you can see, we have a count method just like list, we have a Remove method , Contains , average, comparer, count .....

all this methods are still there just like your list

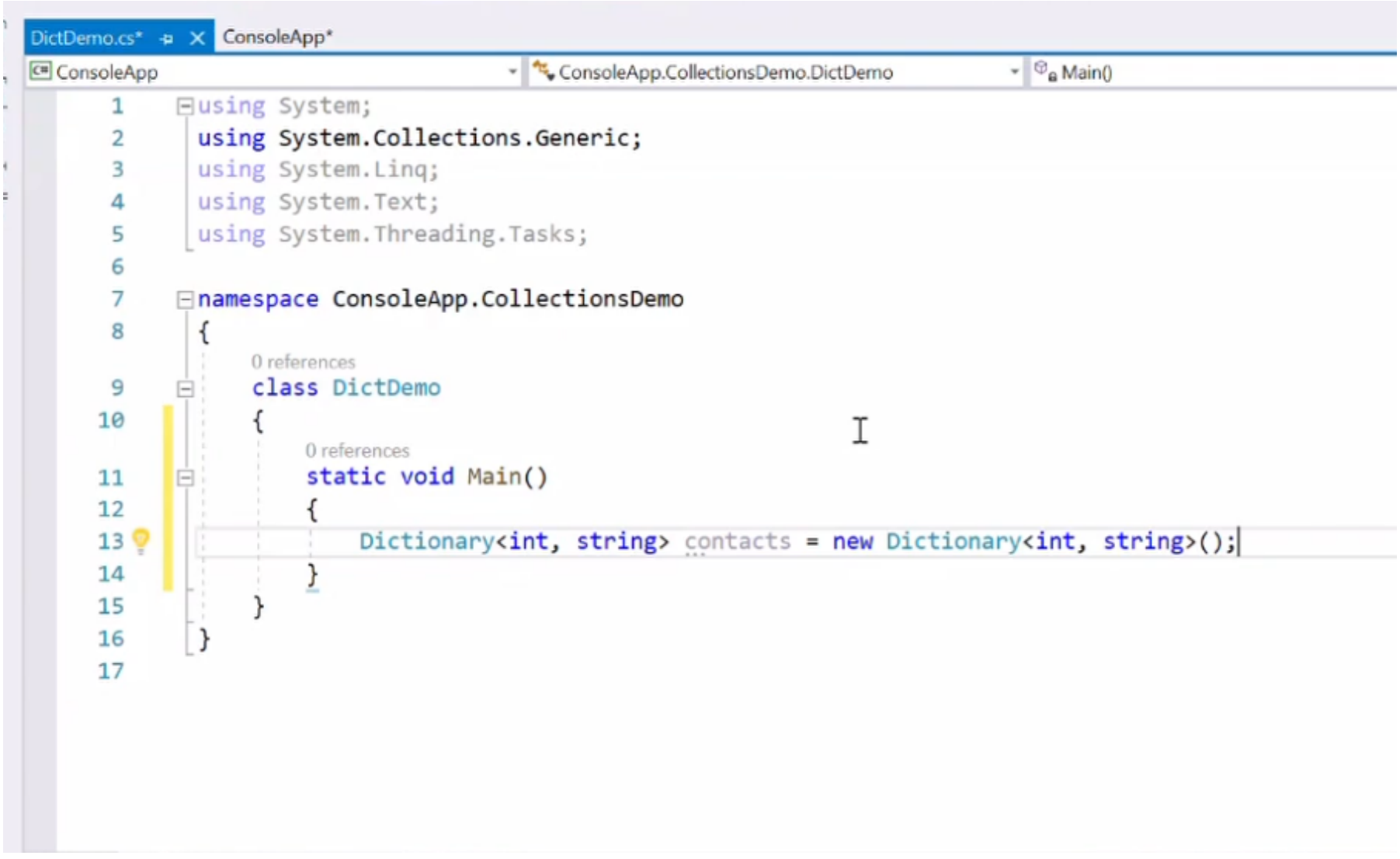
Dictionary

Speciality of dictionary is it will store the data in a key, value pair

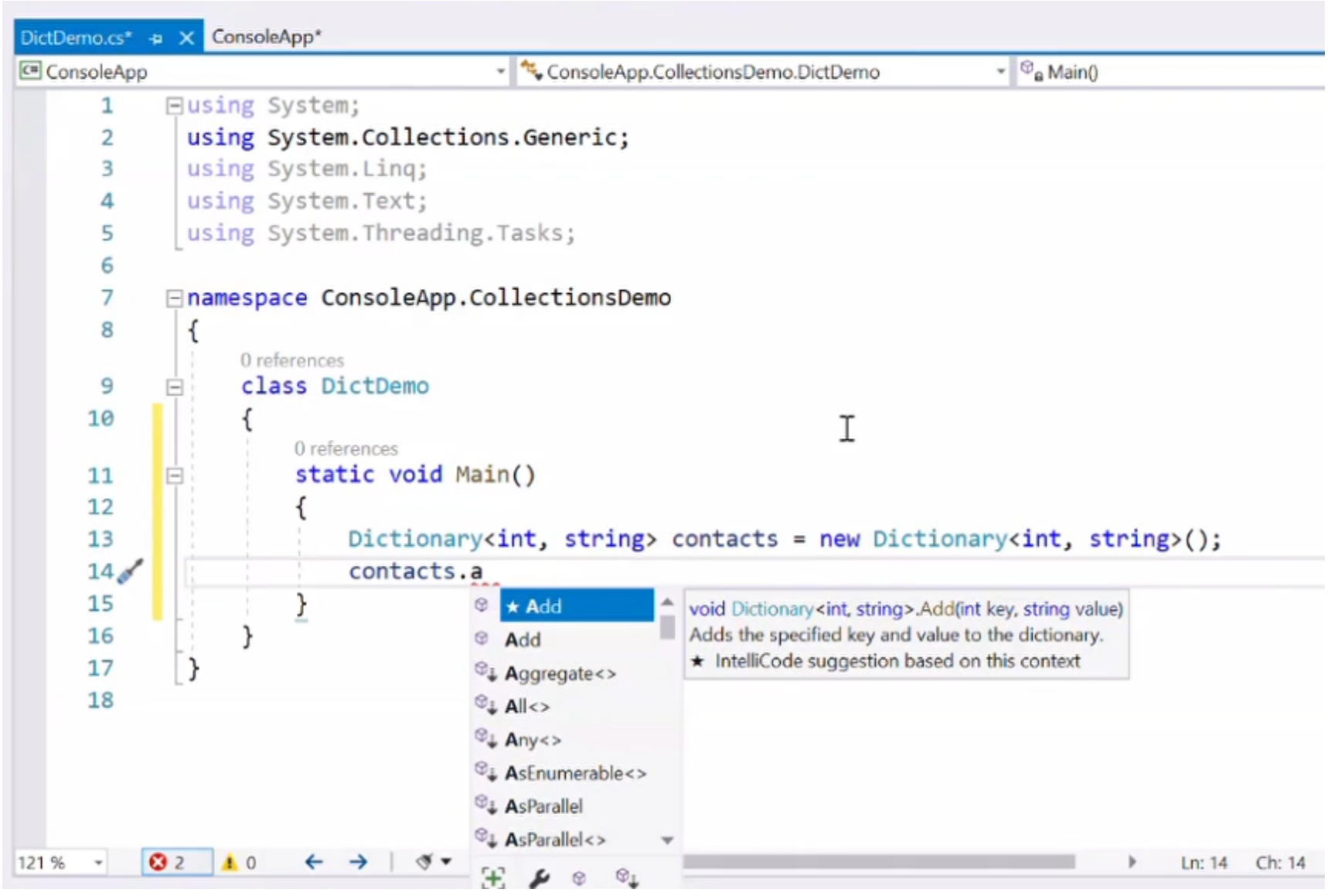
and dictionary is something which we usually use alot in project



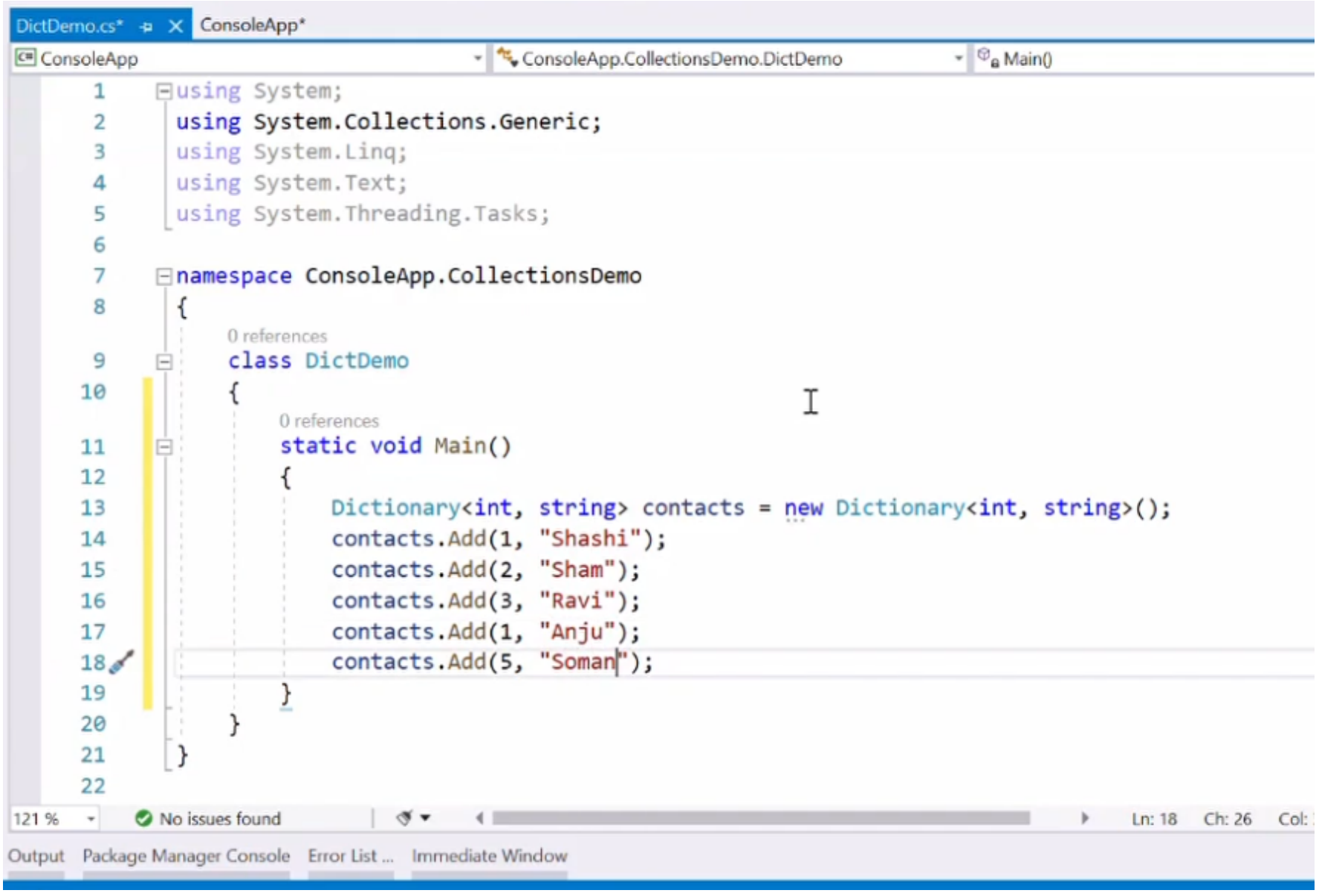
so dictionary of , i'll put my key as integer and my value as string



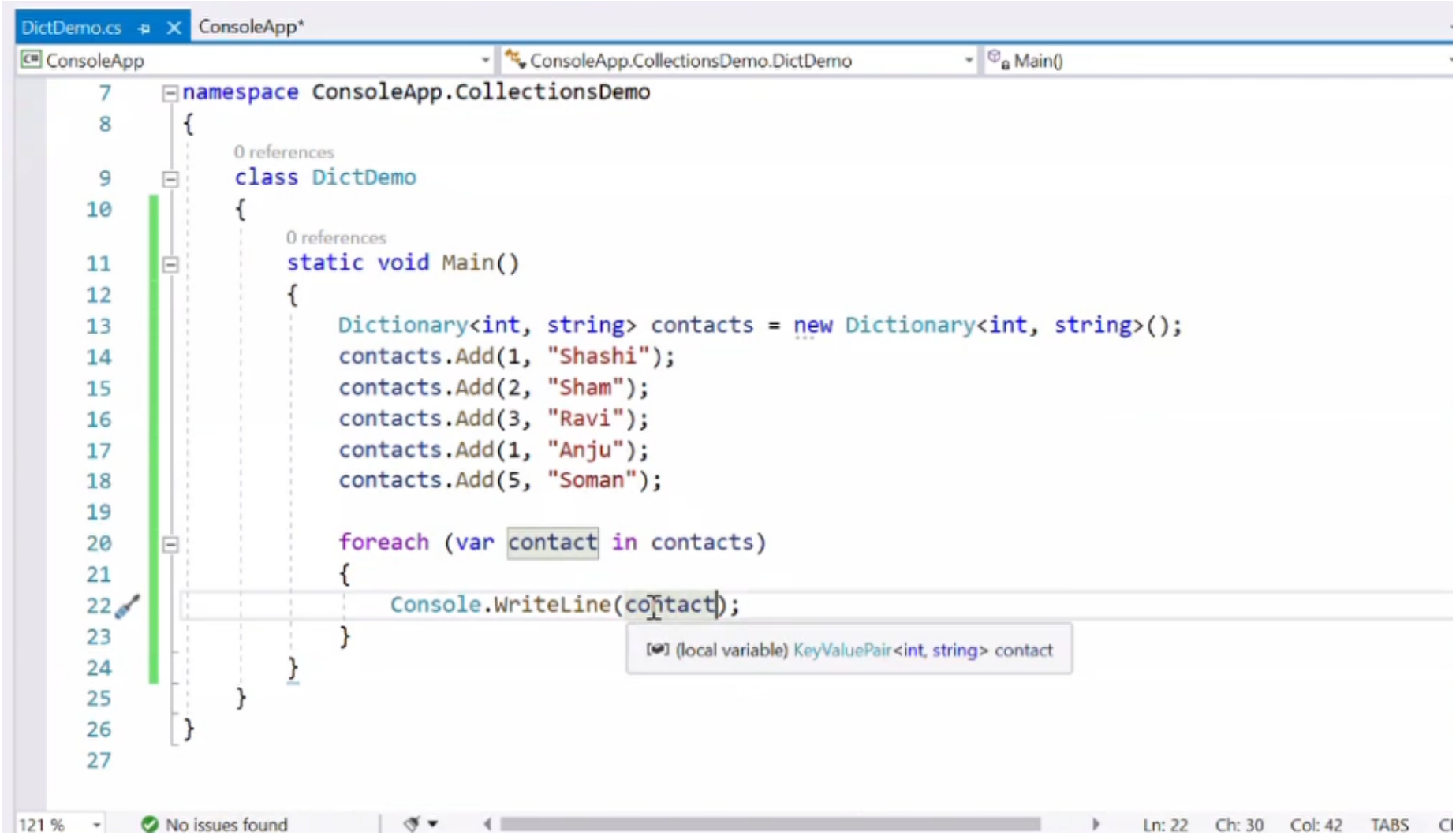
now how do you store the data, so we'll try
try with add


DictDemo.cs*
ConsoleApp*
ConsoleApp
ConsoleApp.CollectionsDemo.DictDemo
Main()
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace ConsoleApp.CollectionsDemo
{
0 references
class DictDemo
{
0 references
static void Main()
{
Dictionary<int, string> contacts = new Dictionary<int, string>();
contacts.a
}
}
}
★ Add
Add
Aggregate<>
All<>
Any<>
AsEnumerable<>
AsParallel
AsParallel<>
void Dictionary<int, string>.Add(int key, string value)
Adds the specified key and value to the dictionary.
★ IntelliCode suggestion based on this context
121 %
2
0
Ln: 14
Ch: 14

so yes dictionary will support add method which is of void type

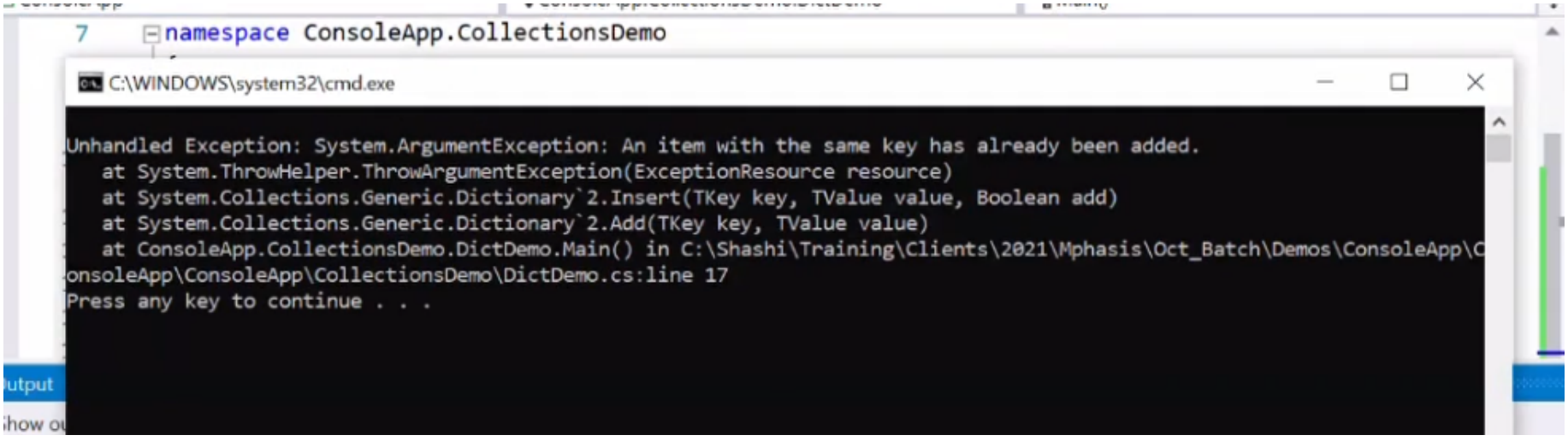

DictDemo.cs*
ConsoleApp*
ConsoleApp
ConsoleApp.CollectionsDemo.DictDemo
Main()
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace ConsoleApp.CollectionsDemo
{
0 references
class DictDemo
{
0 references
static void Main()
{
Dictionary<int, string> contacts = new Dictionary<int, string>();
contacts.Add(1, "Shashi");
contacts.Add(2, "Sham");
contacts.Add(3, "Ravi");
contacts.Add(1, "Anju");
contacts.Add(5, "Soman");
}
}
}
121 %
No issues found
Ln: 18
Ch: 26
Col:
Output
Package Manager Console
Error List ...
Immediate Window

so this is how we add the data
now to fetch the data again we'll use the foreach loop
keep this in your mind when ever you are creating a collection it should always be a plural , it cannot be a book, it should be books



when i move my mouse pointer on this, see contact is in key value pair

so when i execute this i'll get error



which says

An item with the same key has already been added.

we, already have data in the particular key

see by mistake we wrote 1 twice

So Remember whenever the data in key repeated it throws

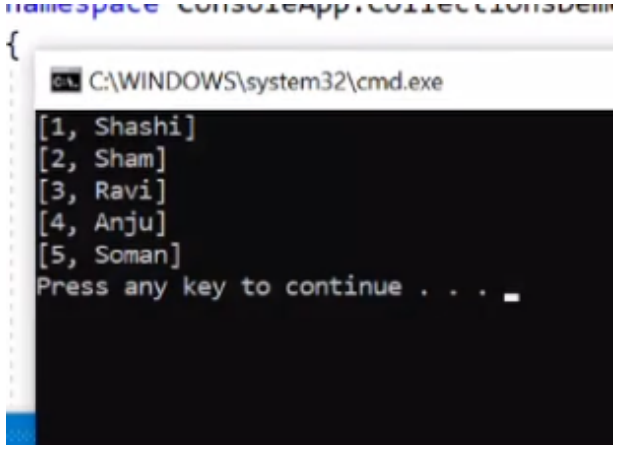
System.ArgumentException

it is a argument exception, not arguments exception

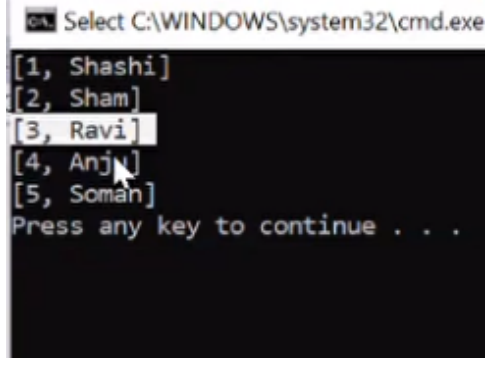
```
contacts.Add(1, "Shashi");
contacts.Add(2, "Sham");
contacts.Add(3, "Ravi");
contacts.Add(1, "Anju");
contacts.Add(5, "Soman");
```

so it should have been 4

now see we got the output

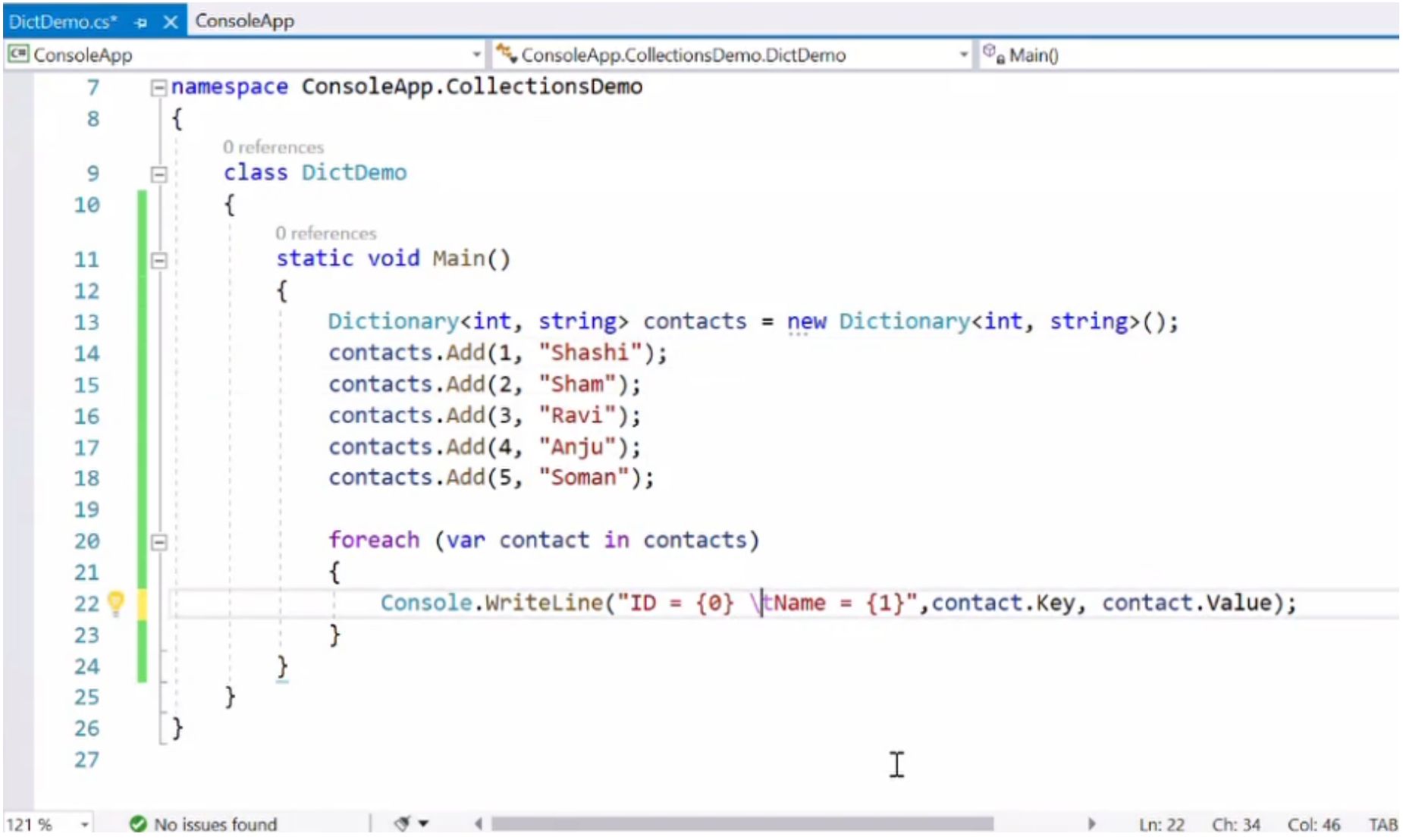


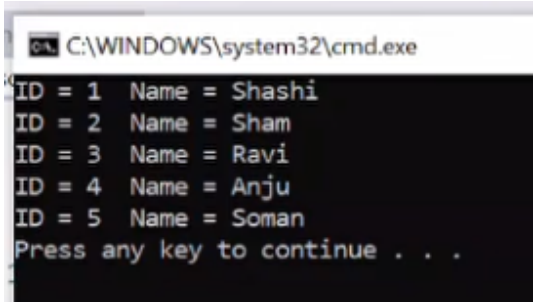
when you look into this there is a square bracket ,



so this is one object, this is one key, value pair what I'm getting

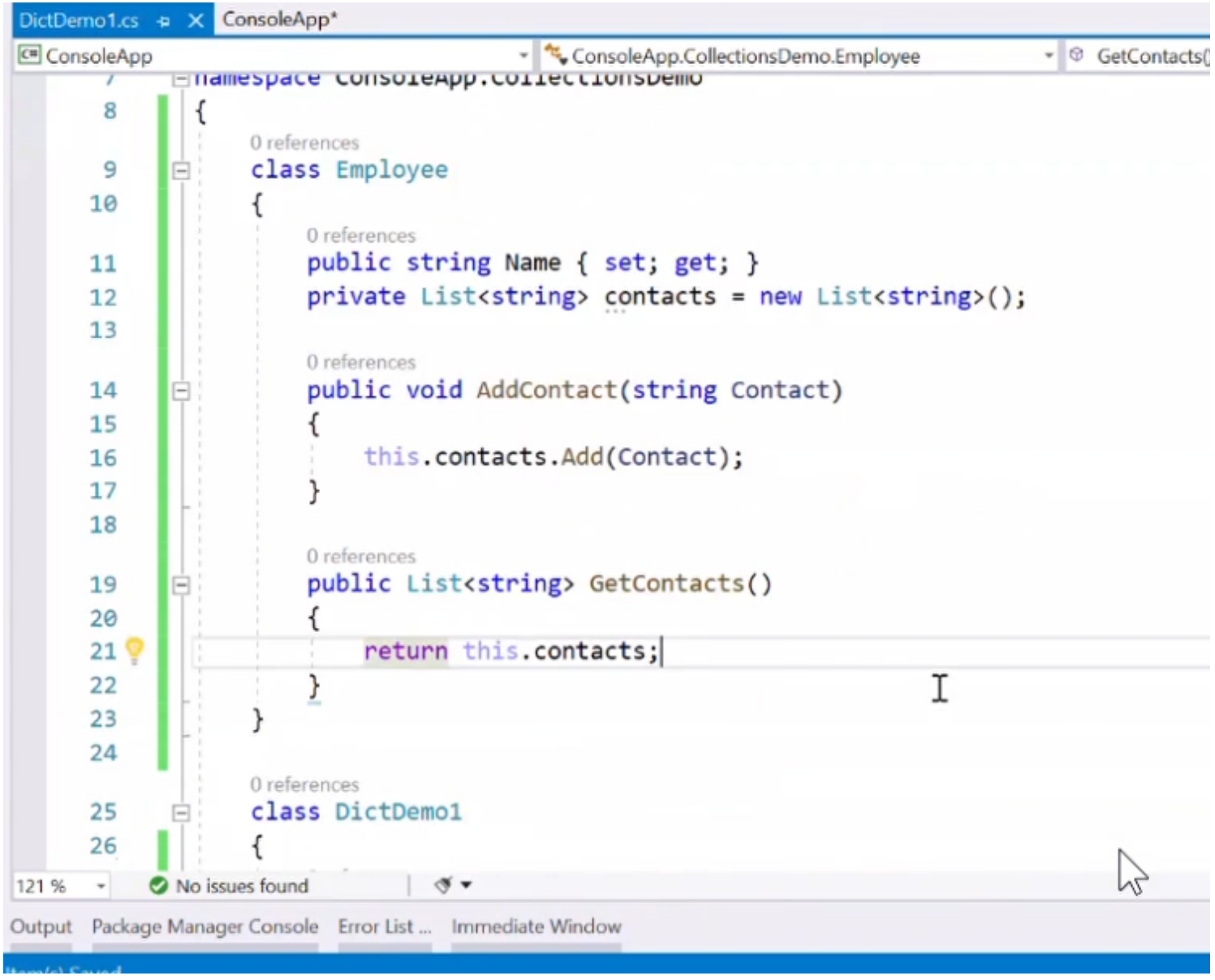
I want to differentiate between my key and value
so you can write like this


DictDemo.cs*
ConsoleApp
ConsoleApp
ConsoleApp.CollectionsDemo.DictDemo
Main()
namespace ConsoleApp.CollectionsDemo
{
0 references
class DictDemo
{
0 references
static void Main()
{
Dictionary<int, string> contacts = new Dictionary<int, string>();
contacts.Add(1, "Shashi");
contacts.Add(2, "Sham");
contacts.Add(3, "Ravi");
contacts.Add(4, "Anju");
contacts.Add(5, "Soman");
foreach (var contact in contacts)
{
Console.WriteLine("ID = {0} \tName = {1}",contact.Key, contact.Value);
}
}
}
}
121 %
No issues found
Ln: 22
Ch: 34
Col: 46
TAB

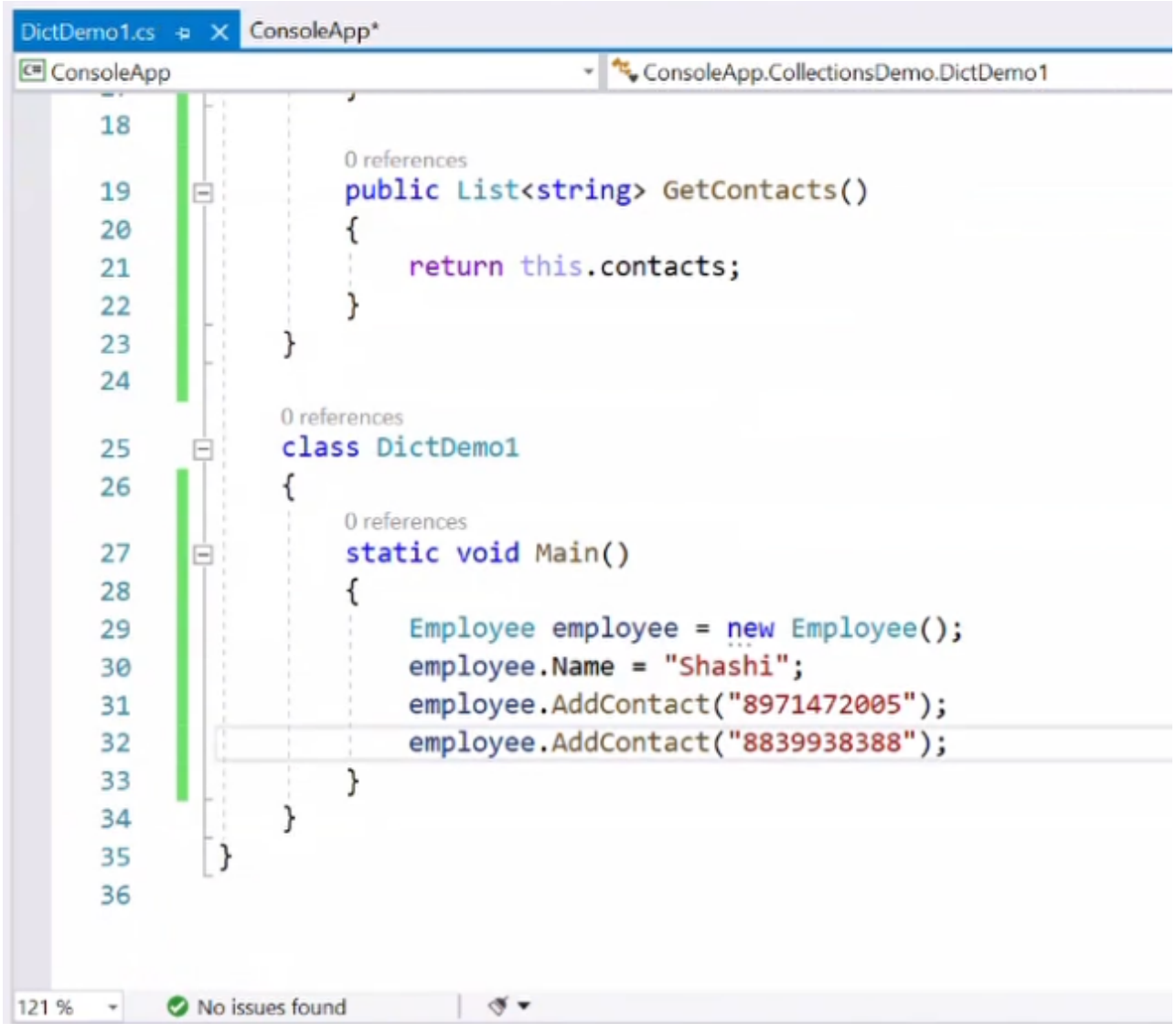


lets look into some complex thing in dictionary

Imagine I want to store **name with multiple contacts**
How do you store this
there were multiple solution to do this 1 employee can have many phone numbers
We already did this, in traier trainee app, 1 trainer can have many trainees


DictDemo1.cs
ConsoleApp*
ConsoleApp
ConsoleApp.CollectionsDemo.Employee
GetContacts()
namespace ConsoleApp.CollectionsDemo
{
0 references
class Employee
{
0 references
public string Name { set; get; }
private List<string> contacts = new List<string>();
0 references
public void AddContact(string Contact)
{
this.contacts.Add(Contact);
}
0 references
public List<string> GetContacts()
{
return this.contacts;
}
}
0 references
class DictDemo1
{
121 %
No issues found
Output
Package Manager Console
Error List ...
Immediate Window

now tomorrow i want to create a contact i'll just write

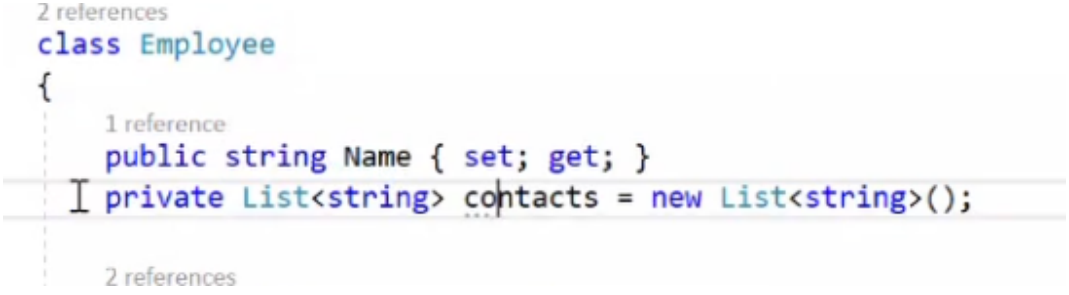


if it's a right thought - See its 2 min's job
wrong thought - it will be 2 years job and even after 2 years you cannot fix it because you are on wrong path, don't make the things complicated

we are taking about dictionary, then all of us have a common mind set , that shashi is talking about dictionary so solution is dictionary, yaa may be solution is dictionary but have you ever tried something else

Please keep thinking of something else there, then only you will get multiple solution, **don't stop at one solution**

So when i do this in list, so why should i go with dictionary
now here if you see, the employee is bounded with the name and the contact



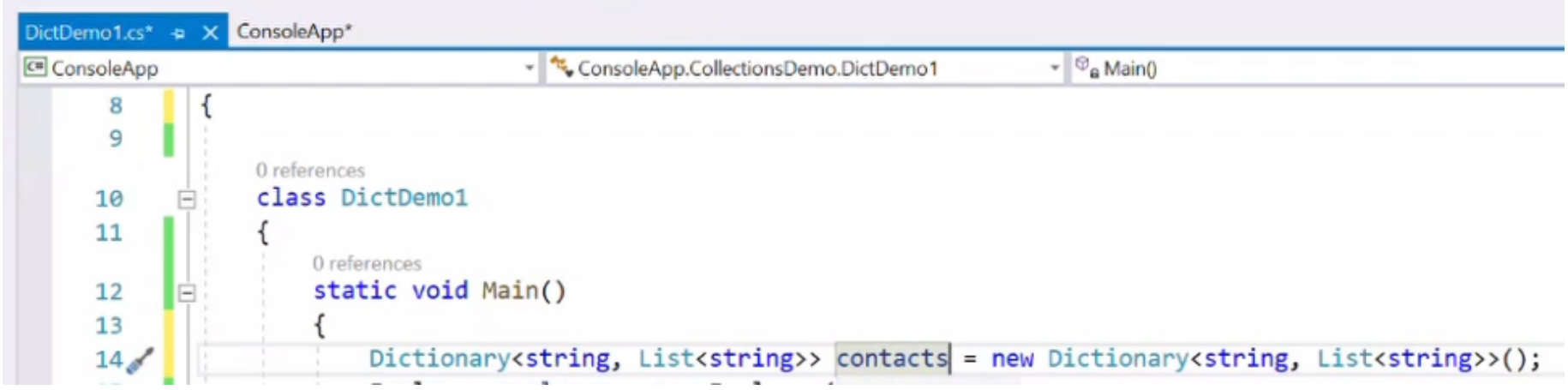
So contact is a part of employee, it is wrapped in the same box, called employee, we cannot segregate it
if i want to access this contact , i can go only through employee like employee.contacts

So there are chances where this design may be wrong for some time

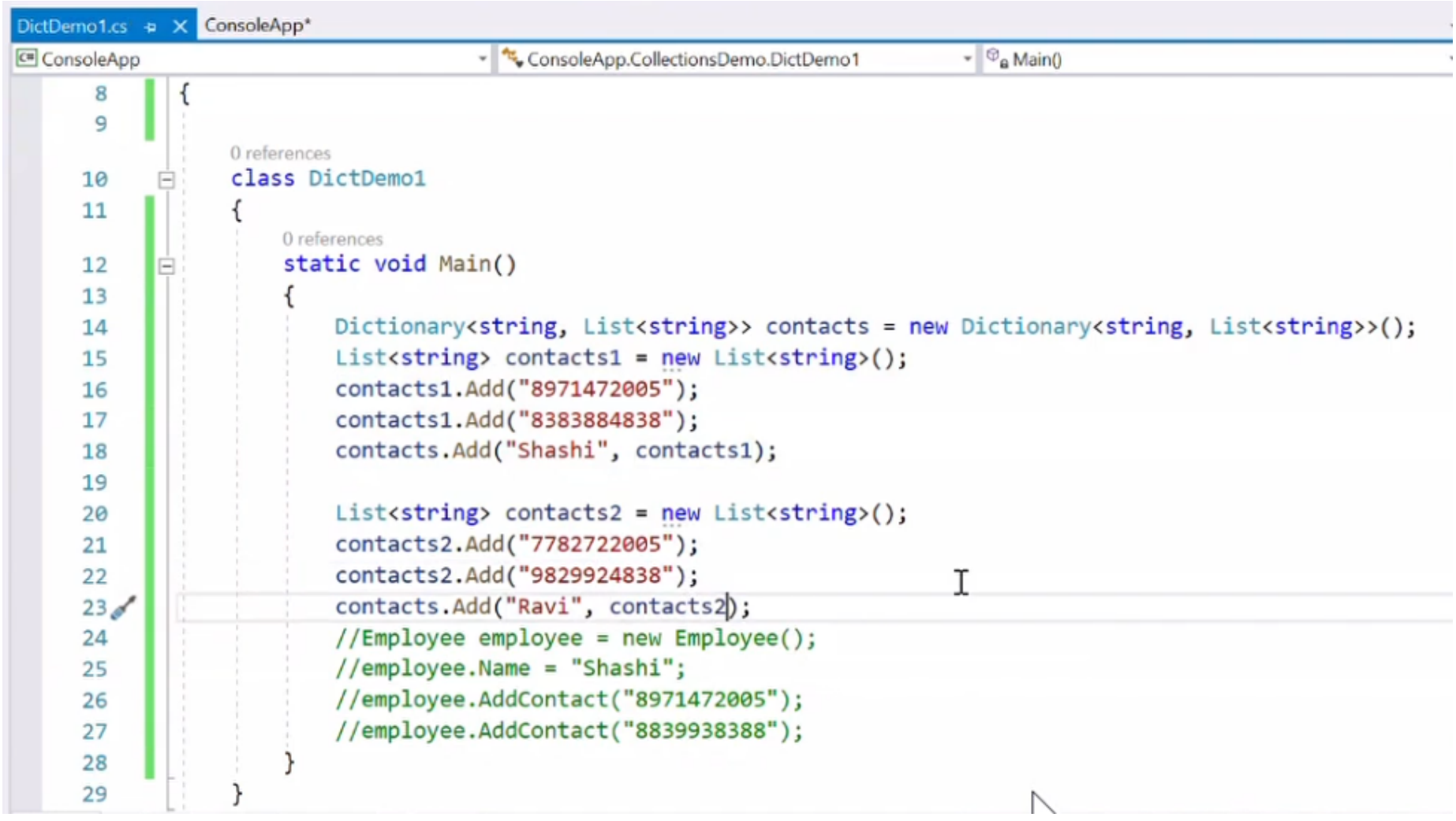
i want to loosely couple this 2 things, i want to keep employee name seprate and contact seprate , but they should be access via same pair , they should be club together

at that point of time you can think of a dictionary

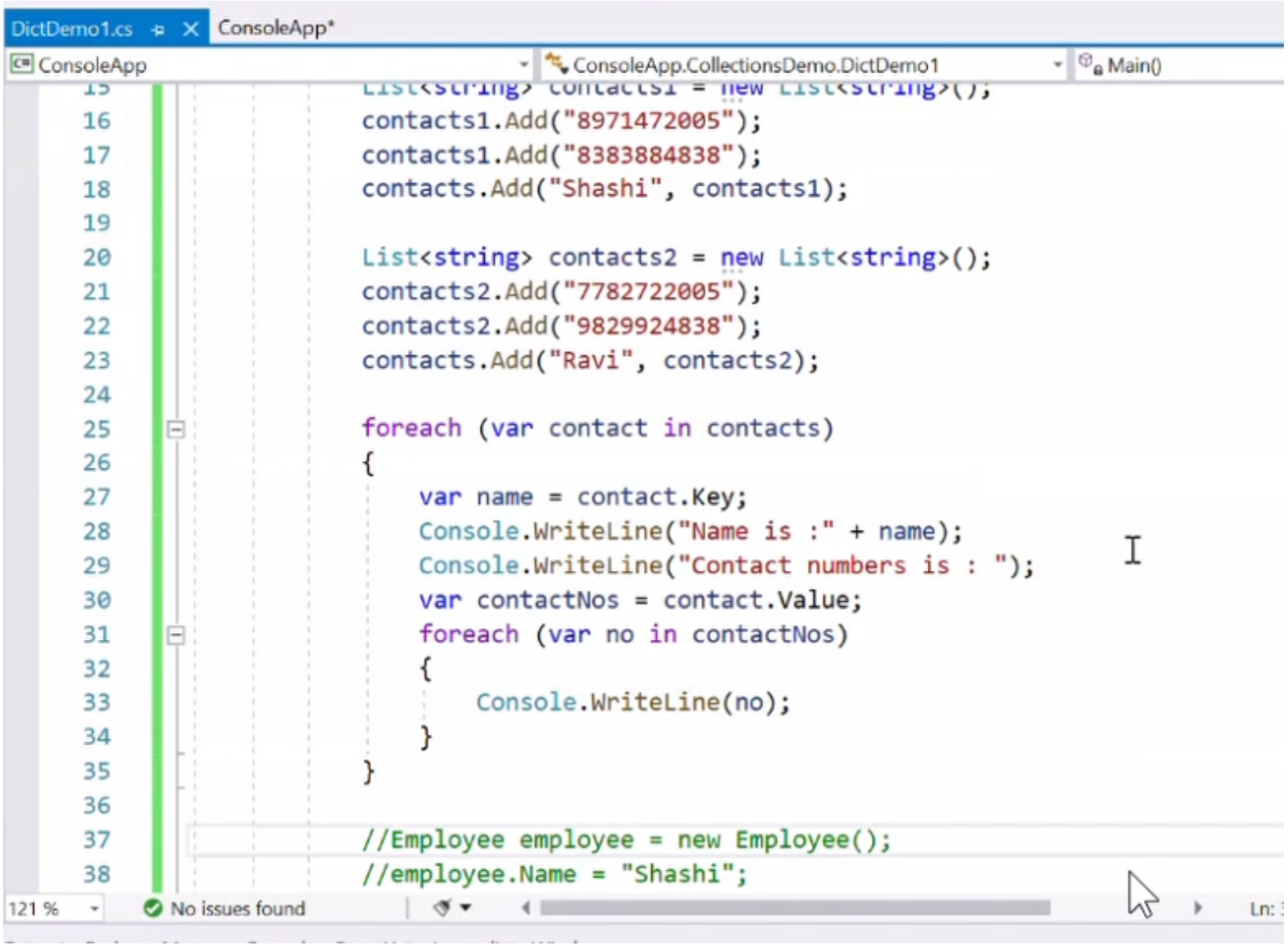
So in the dictionary i'll select the name as a key which will be my string and contact no as list because we have to store many contacts

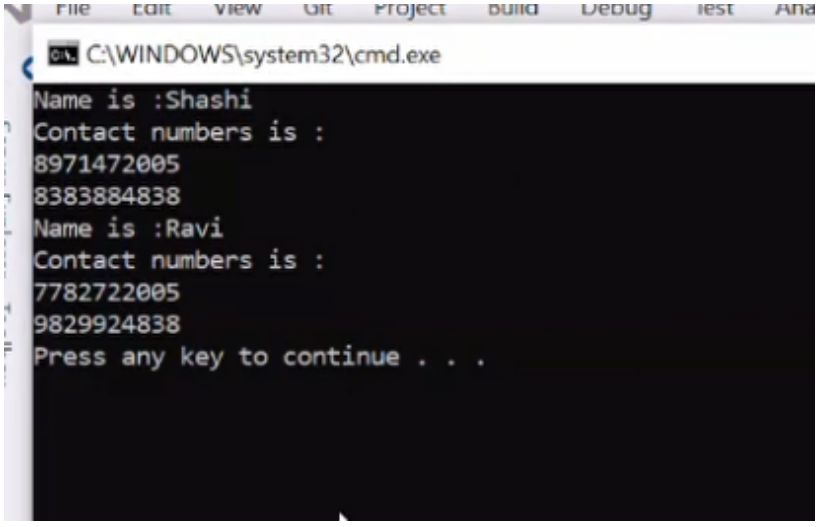


but what if you have 2 person with the same name, you cannot store that with the string, you should take employee id in that case as employee id will always be unique



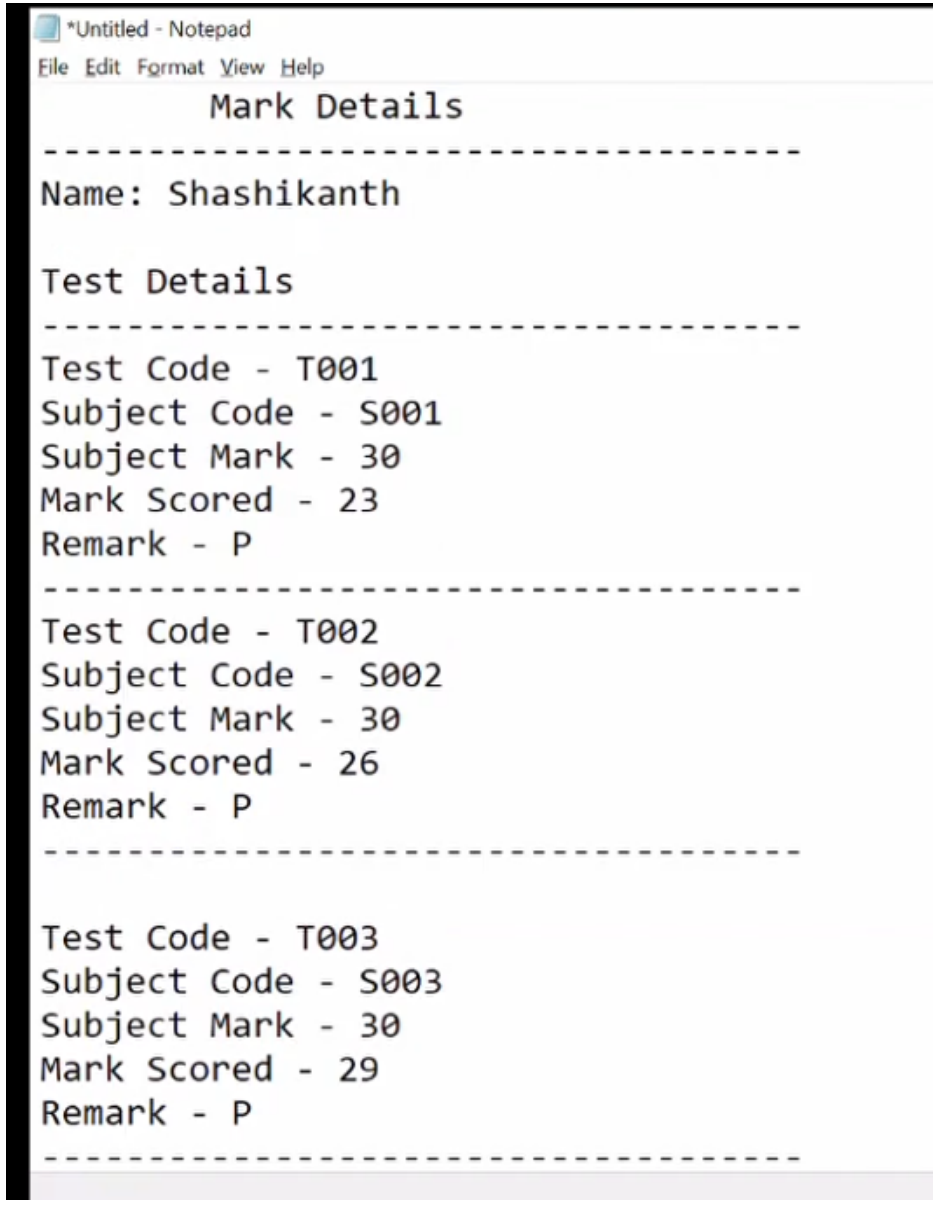
now to get data we use foreach loop
and as our contact is also a list we use foreach loop there too


DictDemo1.cs
ConsoleApp*
ConsoleApp
ConsoleApp.CollectionsDemo.DictDemo1
Main()
List<string> contacts1 = new List<string>();
contacts1.Add("8971472005");
contacts1.Add("8383884838");
contacts.Add("Shashi", contacts1);
List<string> contacts2 = new List<string>();
contacts2.Add("7782722005");
contacts2.Add("9829924838");
contacts.Add("Ravi", contacts2);
foreach (var contact in contacts)
{
var name = contact.Key;
Console.WriteLine("Name is :" + name);
Console.WriteLine("Contact numbers is : ");
var contactNos = contact.Value;
foreach (var no in contactNos)
{
Console.WriteLine(no);
}
}
//Employee employee = new Employee();
//employee.Name = "Shashi";
121 %
No issues found



Assignment :

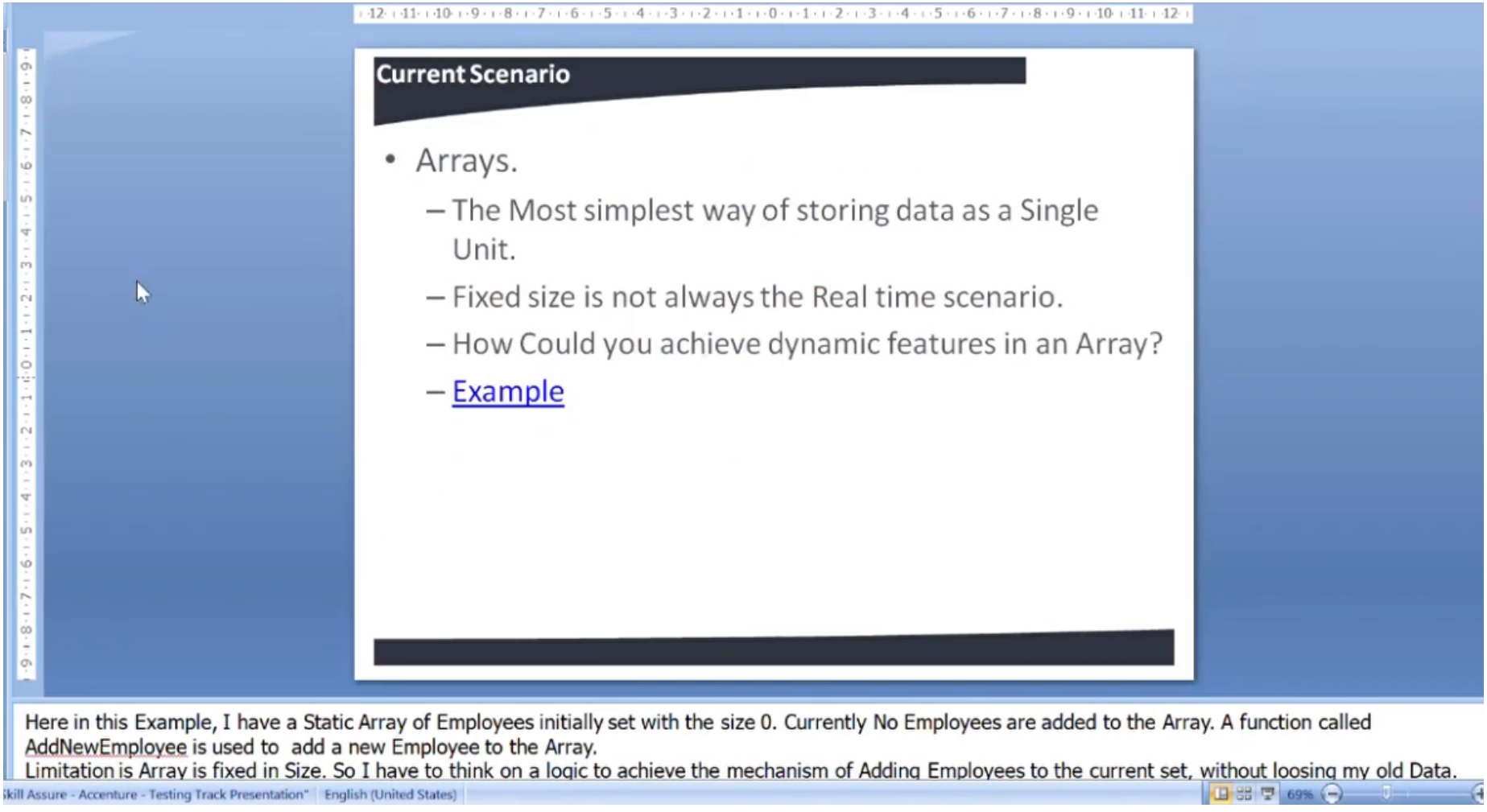
WAP to print mark report like this


*Untitled - Notepad
File Edit Format View Help
Mark Details
Name: Shashikanth
Test Details
Test Code - T001
Subject Code - S001
Subject Mark - 30
Mark Scored - 23
Remark - P
Test Code - T002
Subject Code - S002
Subject Mark - 30
Mark Scored - 26
Remark - P
Test Code - T003
Subject Code - S003
Subject Mark - 30
Mark Scored - 29
Remark - P

.NET Collections and Generics

Objectives

- After the end of the Session the participants should be able to:
  - Problems with Array.
  - Understand what are Collections.
    - What?
    - Why?
    - How?
  - Understand Generics- GenX Collections.
  - What are the built-in Classes of Generics
  - What are the Interfaces of Generics
  - Create Custom Generics
  - Implementing Custom generics.

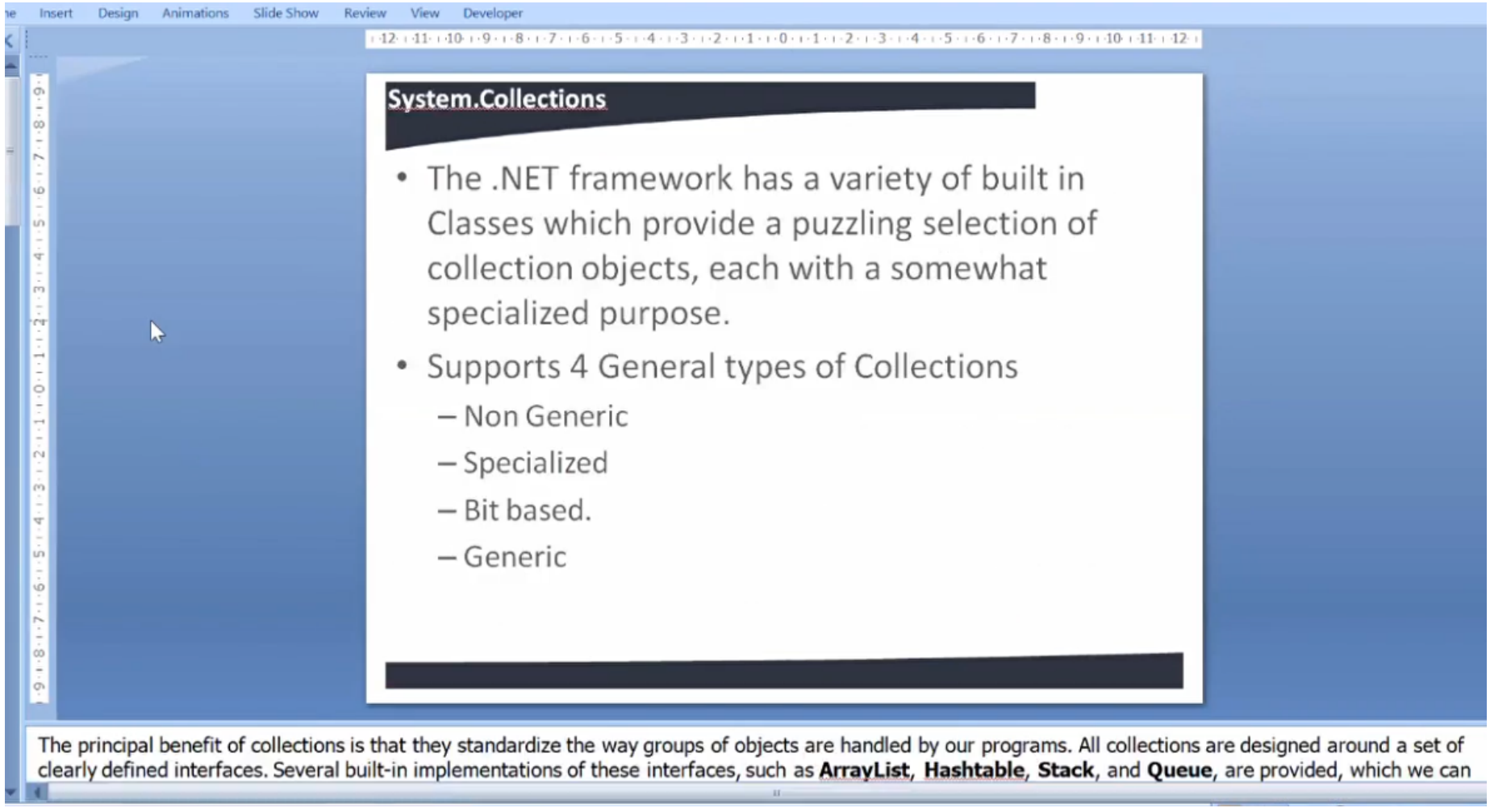

Current Scenario
• Arrays.
– The Most simplest way of storing data as a Single Unit.
– Fixed size is not always the Real time scenario.
– How Could you achieve dynamic features in an Array?
– Example
Here in this Example, I have a Static Array of Employees initially set with the size 0. Currently No Employees are added to the Array. A function called AddNewEmployee is used to add a new Employee to the Array.
Limitation is Array is fixed in Size. So I have to think on a logic to achieve the mechanism of Adding Employees to the current set, without loosing my old Data.
English (United States)
69%

Collections

- What?
  - An object that stores a group of elements together as one unit.
  - A group of data of similar type semantically grouped for a purpose.
  - Arrays are the most primitive type of Collections available in all Languages.
- Why?
  - Traditional Arrays are fixed in Size.
  - Real time Applications needs an ability to append, Remove or modify the existing Data as per their needs.
  - An Example without an Idea of Dynamic Data.
- How?
  - .NET framework provides a huge sets of classes which fulfill the issues of Collections.
  - Grouped under the namespace System.Collections.

Collections under .NET Terminology

- Anything that allows to refer each element within it is called Collection.
- Simply put, It's a class that implements an interface called IEnumerable.


Insert
Design
Animations
Slide Show
Review
View
Developer
System.Collections
• The .NET framework has a variety of built in Classes which provide a puzzling selection of collection objects, each with a somewhat specialized purpose.
• Supports 4 General types of Collections
– Non Generic
– Specialized
– Bit based.
– Generic
The principal benefit of collections is that they standardize the way groups of objects are handled by our programs. All collections are designed around a set of clearly defined interfaces. Several built-in implementations of these interfaces, such as **ArrayList**, **Hashtable**, **Stack**, and **Queue**, are provided, which we can

Generic vs. Non Generic

- Non Generic stores the data as Object
  - Not type safe
  - Available in v1.0 and v1.1
- Generic are type safe way of storing the objects as Data structures
  - Available since v2.0
  - Easy way of storing objects.
- Example of Non Generic Version.

Using Non Generic Class

```
ArrayList list = new ArrayList();
list.Add(123);//Stores the data as object(Boxed Value)
list.Add("SomeName");//Accepts this, but Wrong interpretation.
//Compiler does not throw Error.
```

Using Generic Class

```
List<int> list = new List<int>();
list.Add(123);//Stores the data as integer.
list.Add("SomeName");/*Does not Accepts this and
Compiler tells about this Error.*/
```

Retrieval Issues

- Using Non Generics:

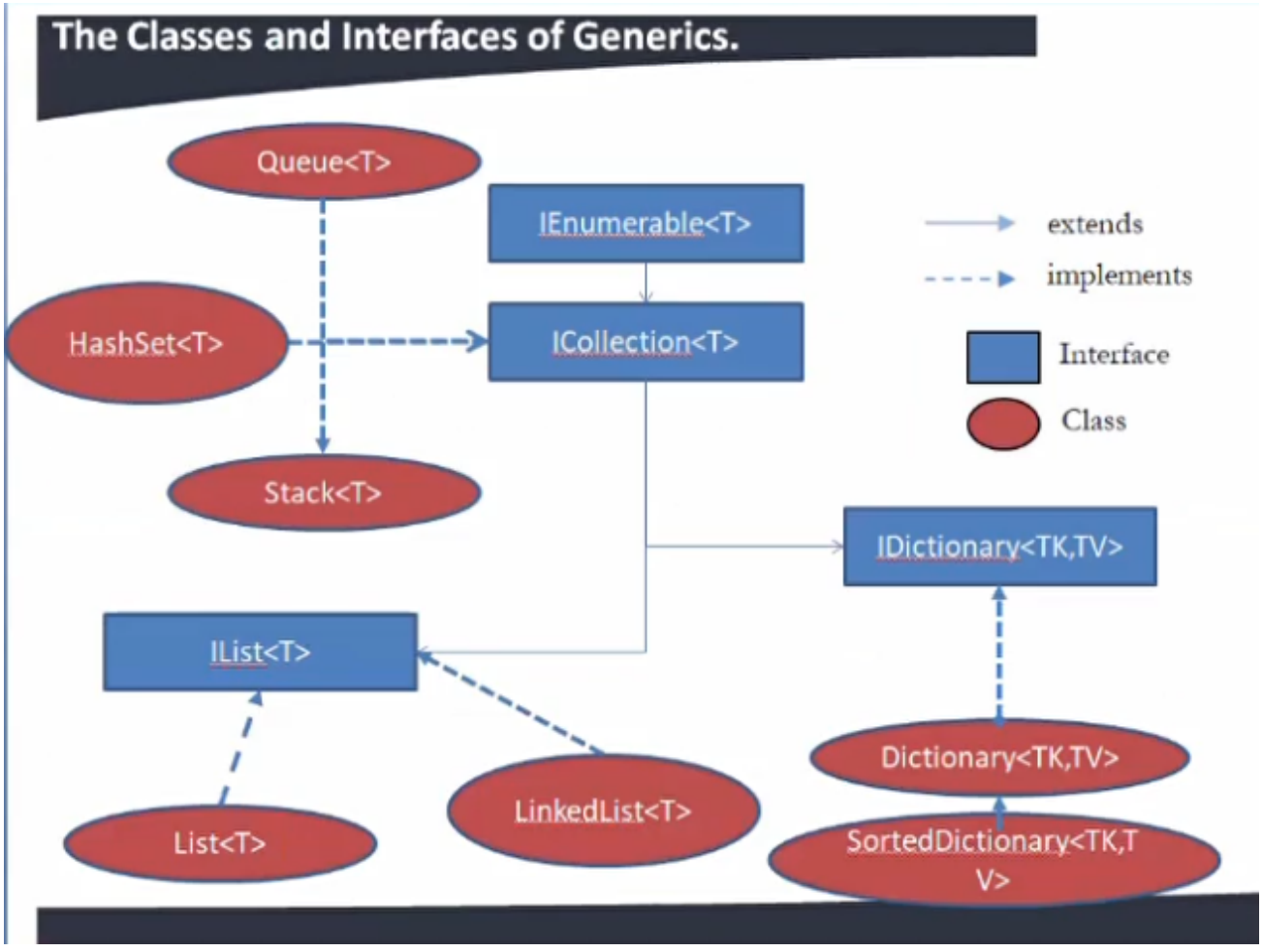
```
foreach(object element in list)
{
    if (element is int)//Check first for type safety.
    {
        //Integer Operations
    }
    else
    {
        //Non Integer Operations
    }
}
```

- Using Generics:

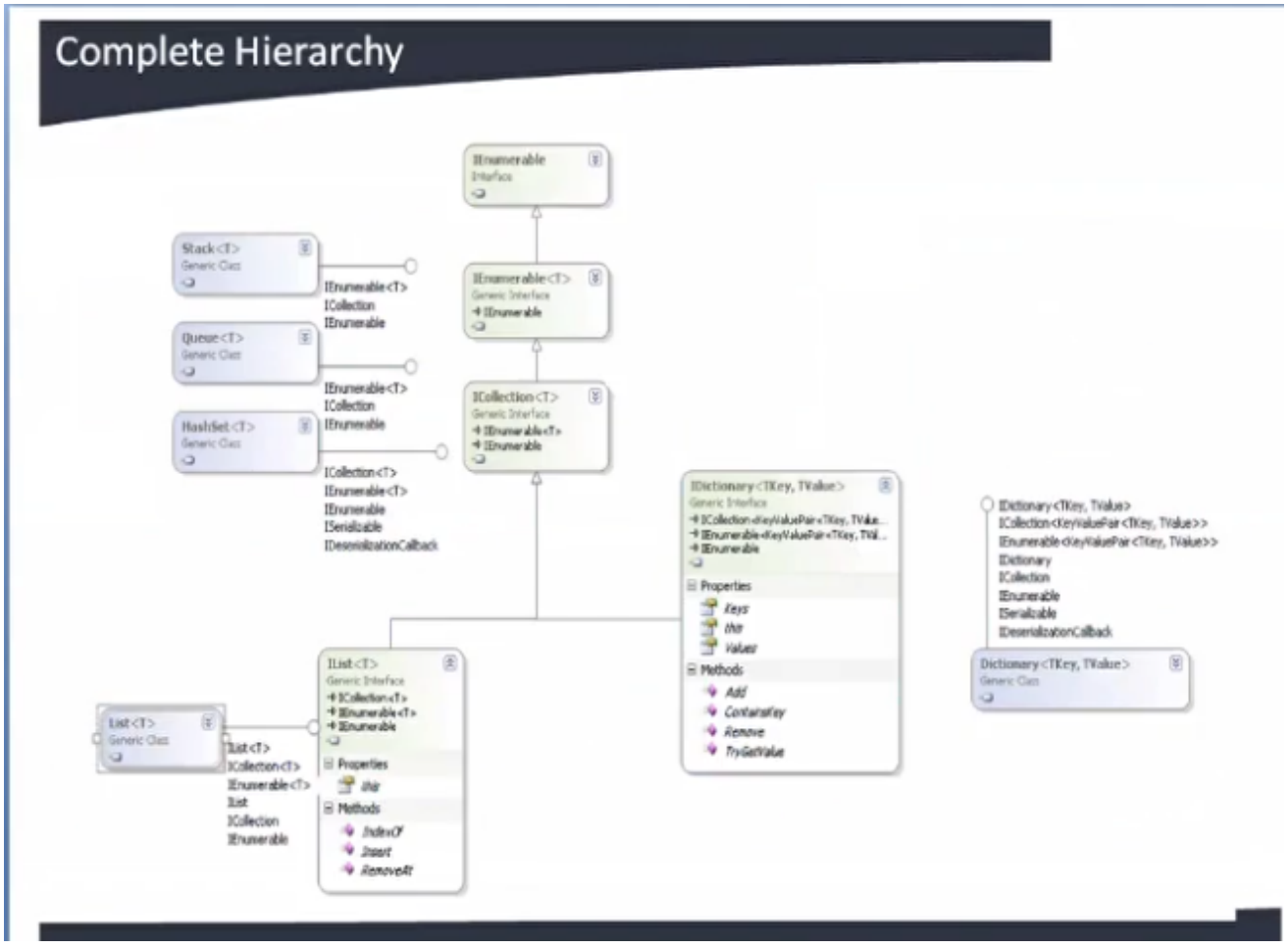
```
foreach(int element in list)//No  Need to check for Integer
{
    //Integer Operations
}
```

Generics

- GenX Collections.
- The whole idea is to avoid accidental Type mismatches.
- Similar to Templates of C++.
- Have Classes to perform type safe way of storing the data.


The Classes and Interfaces of Generics.
Queue<T>
IEnumerable<T>
extends
implements
HashSet<T>
ICollection<T>
Interface
Class
Stack<T>
IDictionary<TK,TV>
IList<T>
Dictionary<TK,TV>
LinkedList<T>
List<T>
SortedDictionary<TK,TV>

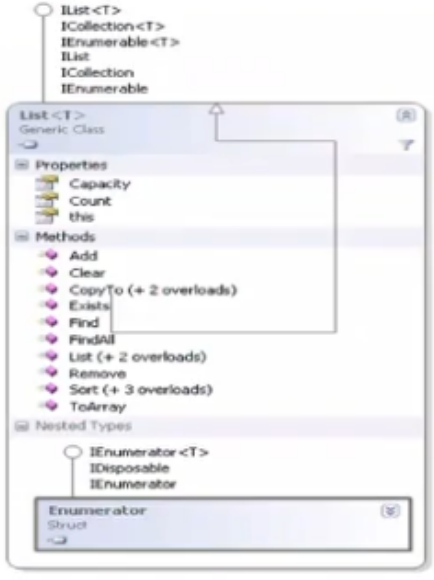
Complete Hierarchy



Classes under Generics

Class	Description
List<T>	A dynamic array. Provides functionality similar to that found in the non-generic ArrayList class.
Dictionary<TK, TV>	Stores key/value pairs. Provides functionality similar to that found in the non-generic Hashtable class.
HashSet<T>	Represents a set of values.
LinkedList<T>	Stores elements in a doubly linked list.
Queue<T>	A first-in, first-out list. Provides functionality similar to that found in the non-generic Queue class.
SortedDictionary<TK, TV>	A sorted list of key/value pairs.
SortedList<TK, TV>	A sorted list of key/value pairs. Provides functionality similar to that found in the non-generic SortedList class.
Stack<T>	A first-in, last-out list. Provides functionality similar to that found in the non-generic Stack class.

The List<T> Design

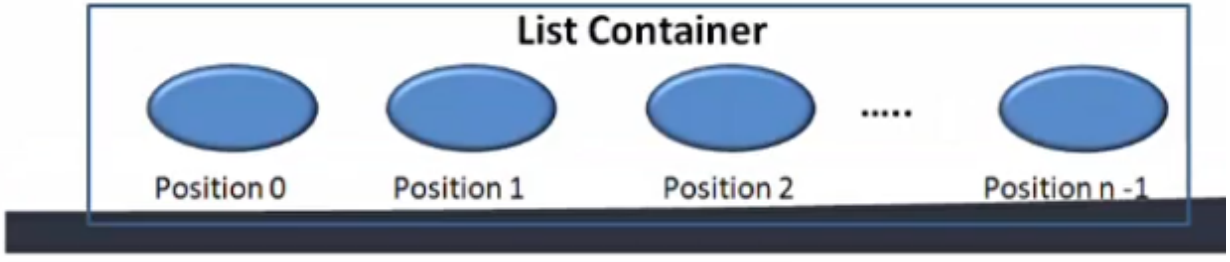


- Provides a Generic Dynamic Array kind of Data storage in it.
- Stores the data as First In Last Out.
- Simplest form of storing the data where we can modify the structure at any point of time.

Some of the Important Methods:
Add->Adds the Element to the Last of the list.
Remove->Removes the Specified Element from the List.

List<T> Usage

- Lists are Dynamic Array.
- Element is added to the Last of the list.
- The Count gives the number of elements within the List.



Code Snippet

```
class Phone...

class Program
{
    static void Main(string[] args)
    {
        List<Phone> mobiles = new List<Phone>();//Initially no Phones will be available
        mobiles.Add(new Phone { PhoneID = 111, PhoneModel = "N73", PhoneCost = 13200 });
        mobiles.Add(new Phone { PhoneID = 112, PhoneModel = "xPhoria", PhoneCost = 18000 });
        mobiles.Add(new Phone { PhoneID = 113, PhoneModel = "5800Xpress", PhoneCost = 12200 });
        //Phones are added using Add Method...

        mobiles.Remove(mobiles[2]);
    }
}
```