DATA STRUCTURES

Lab 3 : Array 2

Lecturer: Hussien Omer AL_Baiti

Previous Home work

Write **algorithm** and **C# code** of search integer item in array:

- user read array size.
- user read array items.
- user read item that search about it.
- If exist in array print message called **is found** and print it’s index .
- Else print message called **is not found .**

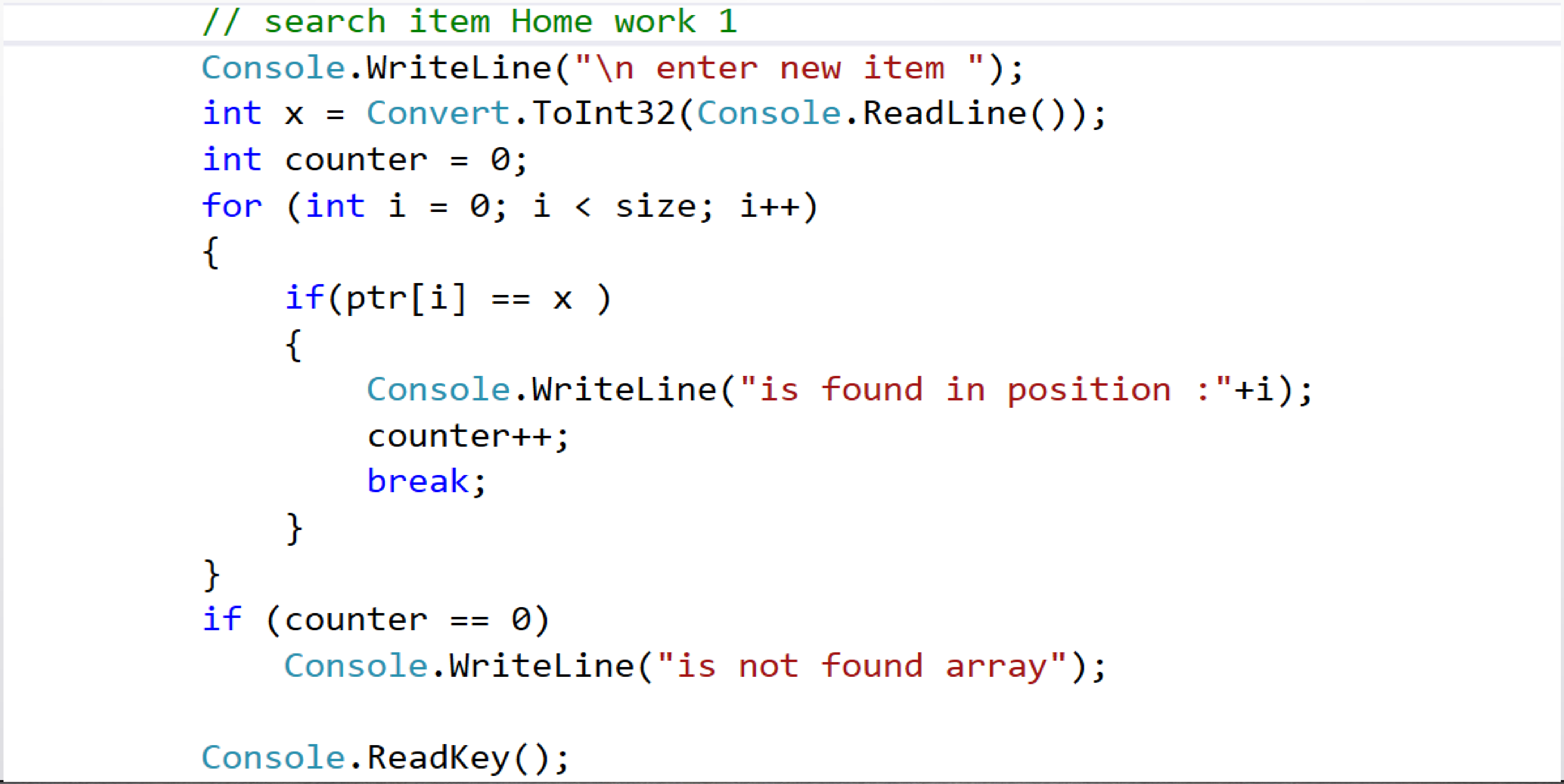
Build an Array Algorithm

```
Step 1 : start
Step 2 : read size of array (size)      ⎫
Step 3 : build array  called (ptr)      ⎬ Input
Step 4 : init i = 0                     ⎭
Step 5 : while i < size                 ⎫
Step 5.1 : ptr [ i ] = read new item    ⎬ Process
Step 5.2 : i++                          ⎭
Step 5.3 : print ptr array  ──────────→  Output
Step 6 : end
```

Search Algorithm

```
Step 1 : start
Step 2 : read item  x
Step 3: init counter = 0 , i=0
Step 4 : loop until i == size
Step 4.1 : if  ( ptr [i] = x )
Step 4.1.1: print is found in i position
Step 4.1.2: counter ++
Step 4.1.2 : break
Step 4.2 : i++
Step 5: if counter == 0
Step 5.1 : print x is not found
Step 6 : end
```

Search Code



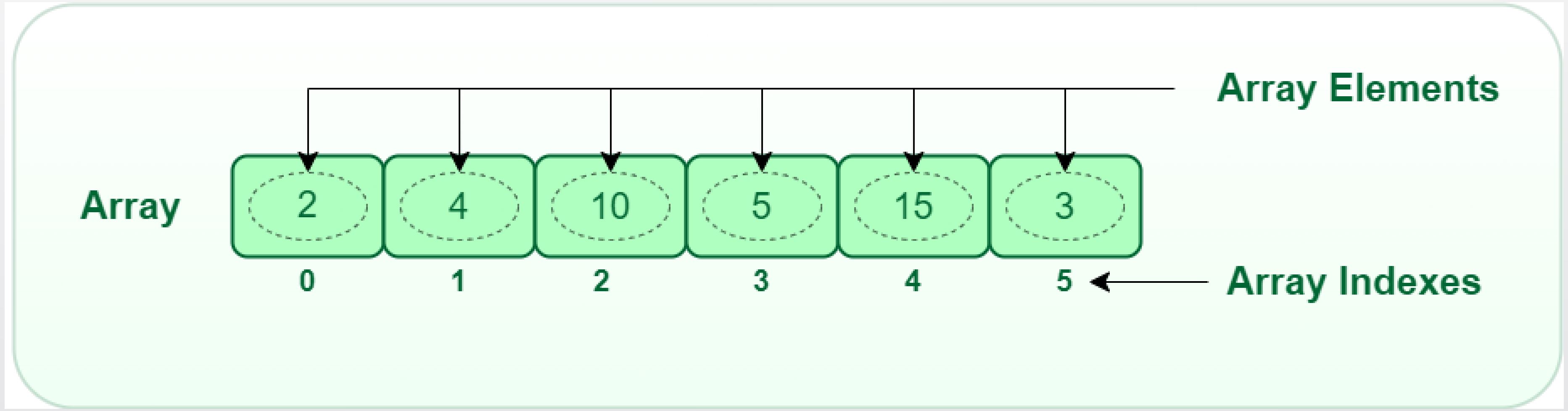
```
// search item Home work 1
Console.WriteLine("\n enter new item ");
int x = Convert.ToInt32(Console.ReadLine());
int counter = 0;
for (int i = 0; i < size; i++)
{
    if(ptr[i] == x )
    {
        Console.WriteLine("is found in position :"+i);
        counter++;
        break;
    }
}
if (counter == 0)
    Console.WriteLine("is not found array");

Console.ReadKey();
```

Exercise 1

exchange the searched item into 55.

Basic Array Operations

- Traverse
- Search
- Update
- Insertion
- Deletion



Insertion

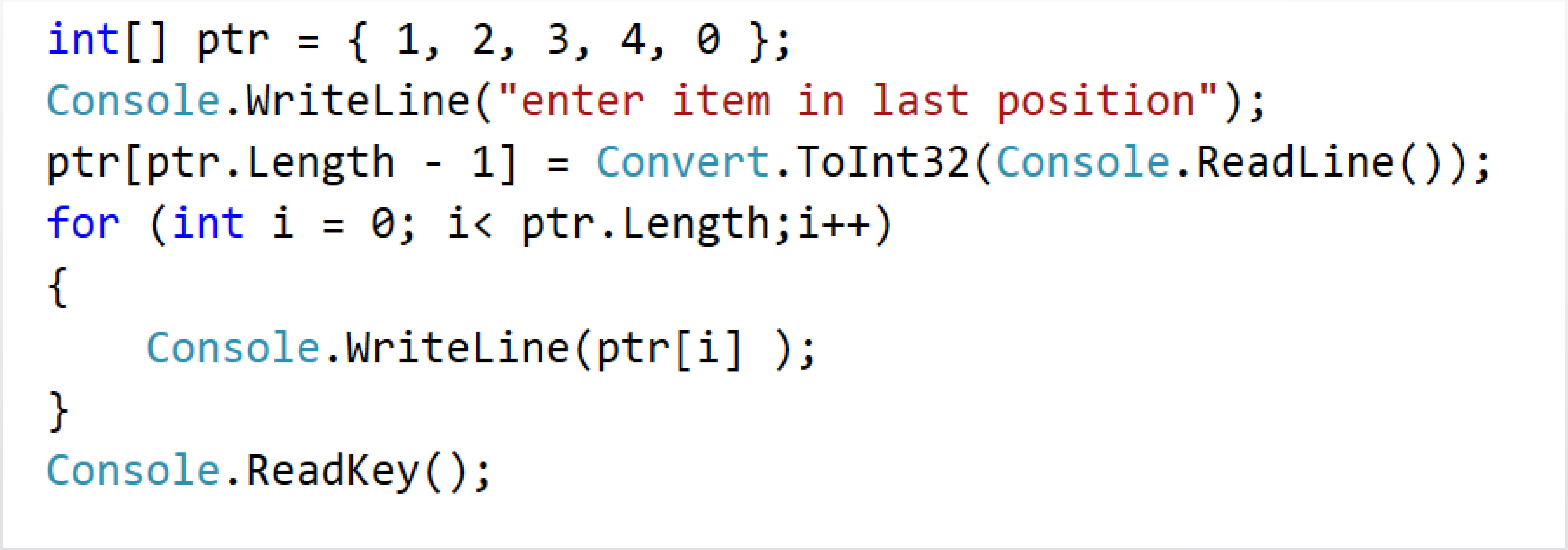
- To execute this operation most be specific size of array, and number of items , the array size most be greater than items number for shifting .
- Insertion in an array can be done in Three ways.

- First Position : shift all items into right position, then add new item in first position
- End Position: add new item in end position
- Any position: most be specific please through (Position or Value), into right position use this rule (size – position)

Add first position

```
int[] ptr = { 1, 2, 3, 4, 0 };
int num = ptr.Length - 1;
// befor add
for (int i =0;i<ptr.Length-1;i++)
{
    Console.Write(ptr[i] + "\t");
}
for (int i = 0; i < ptr.Length-1; i++)
{
    ptr[num] = ptr[num - 1];
    num--;
}
Console.WriteLine("\n enter new item");
ptr[0] = Convert.ToInt32(Console.ReadLine());
// after add
for (int i = 0; i < ptr.Length ; i++)
{
    Console.Write(ptr[i] + "\t");
}
Console.ReadKey();
```

Add end position



```
int[] ptr = { 1, 2, 3, 4, 0 };
Console.WriteLine("enter item in last position");
ptr[ptr.Length - 1] = Convert.ToInt32(Console.ReadLine());
for (int i = 0; i< ptr.Length;i++)
{
    Console.WriteLine(ptr[i] );
}
Console.ReadKey();
```

Delete

- Delete in an array can be done in Three ways.

➢ First Position : shift all items into Left position, then add zero item in Last position.

➢ End Position: add zero item in Last position .

➢ Any position: most be specific please through (Position or Value), into right position use this rule (size – 1) - position .

Delete first position

```
int[] ptr = { 1, 2, 3, 4, 8 };
int num = 0;
//// befor delete
Console.WriteLine("befor");

for (int i = 0; i < ptr.Length; i++)
{
    Console.Write(ptr[i] + "\t");
}
for (int i = 0; i < ptr.Length - 1; i++)
{
    ptr[num] = ptr[num + 1];
    num++;
}
ptr[num] = 0;
Console.WriteLine("after");

// after delete
for (int i = 0; i < ptr.Length - 1; i++)
{
    Console.Write(ptr[i] + "\t");
}

Console.ReadKey();
```

Delete end position

```
int[] ptr = { 1, 2, 3, 4, 8 };

ptr[ptr.Length - 1] = 0;

// after delete
for (int i = 0; i < ptr.Length - 1; i++)
{
    Console.Write(ptr[i] + "\t");
}

Console.ReadKey();
```

Home work 1

Write **algorithm** and **C# code** of add and delete at any position in array:

- user read array size.
- user read array items.
- user read array position .

THE END