**Task-3**

**Exercise - 1**

using System.Collections.Generic;

using System;

public class Exercise1

{

public static List<string> ProcessToKill(List<string> process)

{

List<string> processToKill = new List<string>(3);

Console.WriteLine(string.Format("Capacity {0}", processToKill.Capacity));

// Show number of items ; here : 0.

Console.WriteLine(string.Format("Count {0}", processToKill.Count));

foreach(var p in process){

if(!p.Equals("Explorer.exe")){

processToKill.Add(p);

}

}

foreach (var p in processToKill)

{

Console.WriteLine(p);

}

return processToKill;

}

public static void Main(){

List<string> process = new List<string>();

process.Add("ui.exe");

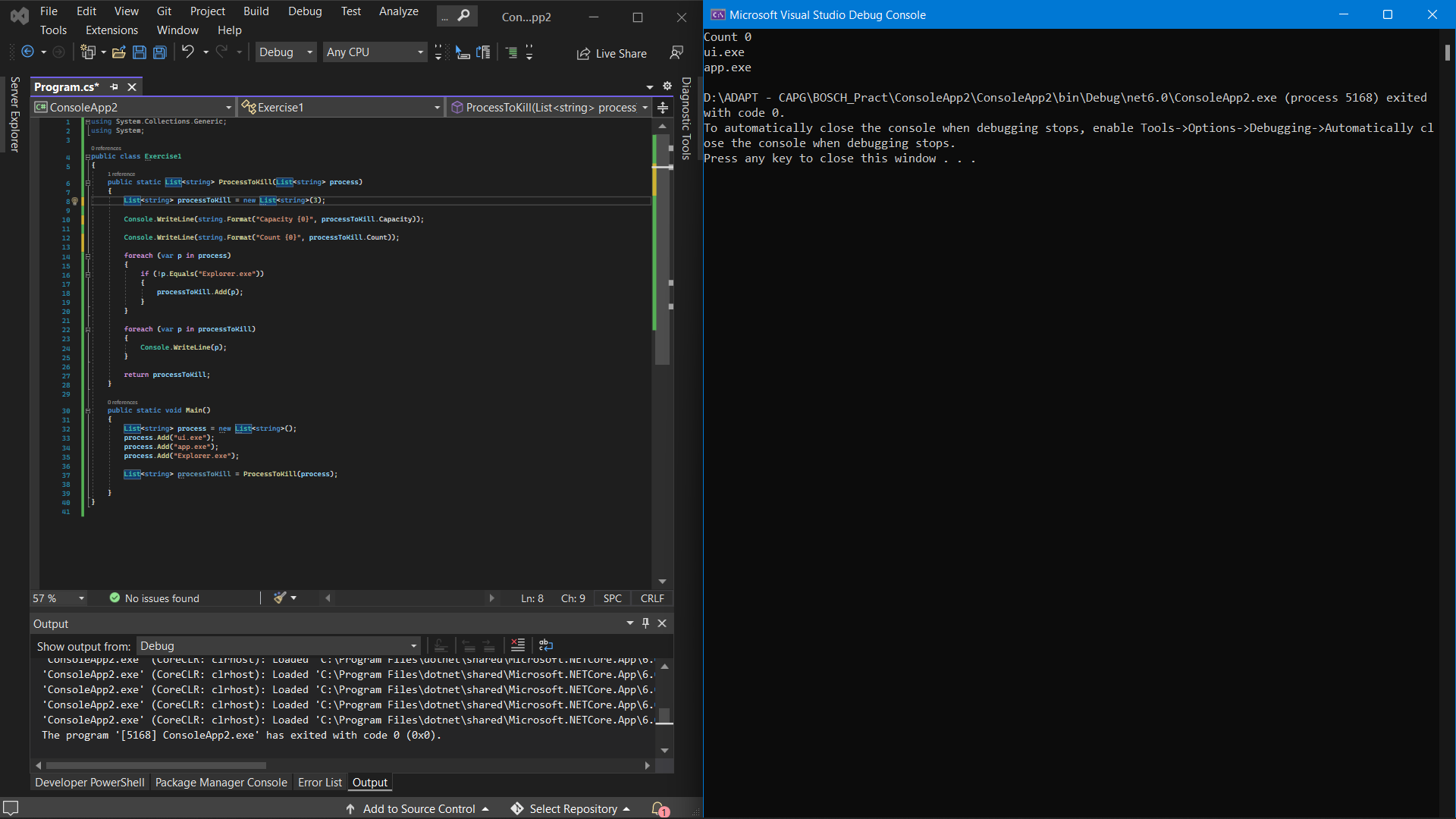
process.Add("app.exe");

process.Add("Explorer.exe");

List<string> processToKill = ProcessToKill(process);

}

}

****

**Exercise – 2**

using System;

using System.Collections.Generic;

using System.Text;

public class DictionaryStub

{

public class Employee : IEquatable<Employee>

{

public int Age;

public string Name;

public Employee(string name, int age)

{

this.Age = age;

this.Name = name;

}

public bool Equals(Employee other)

{

return (this.Age == other.Age && this.Name == other.Name);

}

}

public static Dictionary<int, List<string>> GetEmployeesByAge(List<Employee> employees)

{

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

foreach (var e in employees)

{

if(result.ContainsKey(e.Age)){

result[e.Age].Add(e.Name);

}

else{

result.Add(e.Age,new List<string>(){e.Name});

}

}

return result;

}

public static void Main(){

List<Employee> Employees = new List<Employee>();

Employees.Add( new Employee("Sam",25));

Employees.Add(new Employee("Sam",25));

Employees.Add(new Employee("Chris",35));

Dictionary<int, List<string>> employeeByAge= GetEmployeesByAge(Employees);

foreach(KeyValuePair<int, List<string>> ele1 in employeeByAge)

{

Console.WriteLine(ele1.Key);

foreach(var e in ele1.Value){

Console.WriteLine(e+",");

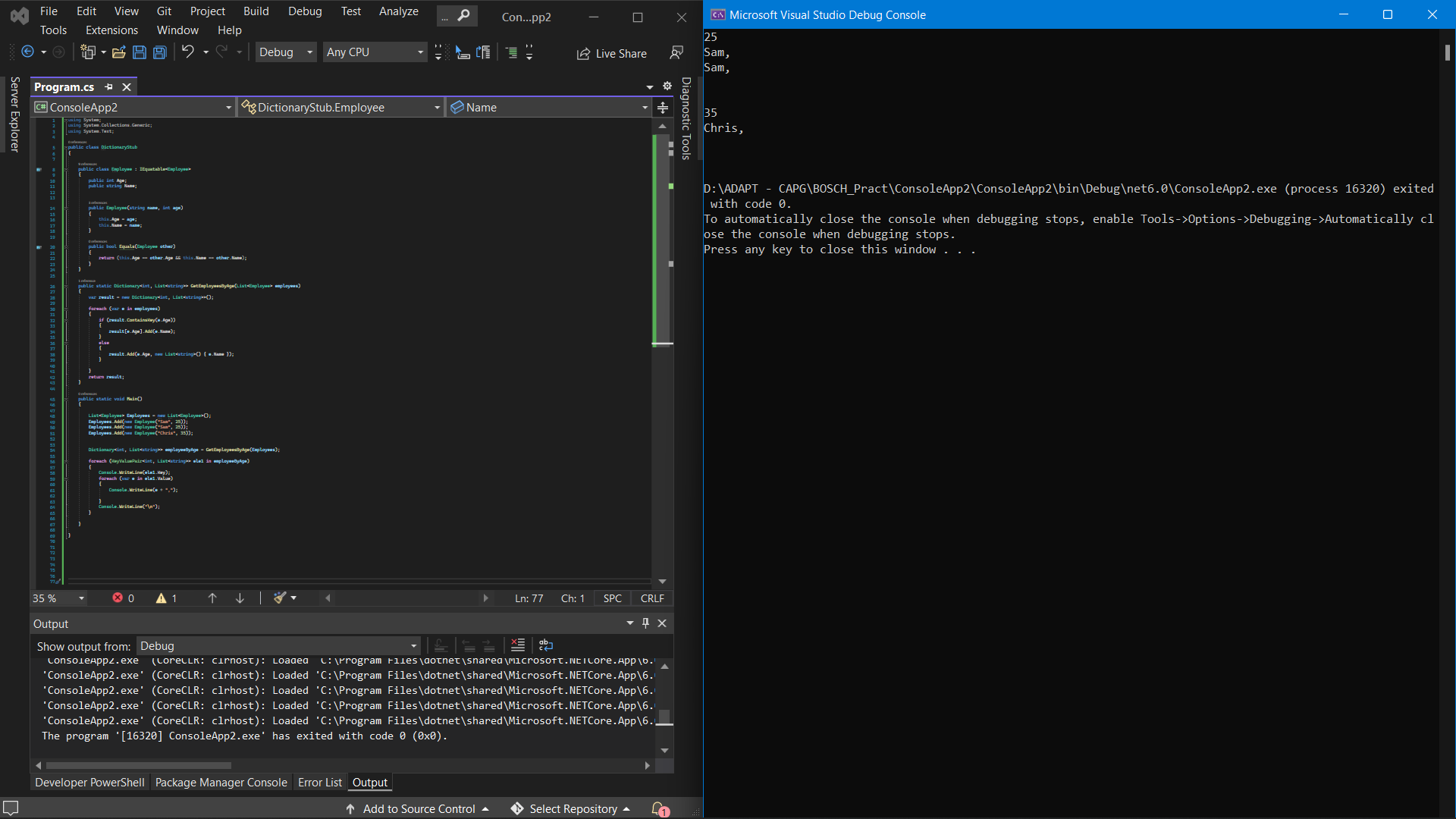
}

Console.WriteLine("\n");

}

}

}



**Exercise – 3**

using System;

using System.Collections.Generic;

using System.Text;

public class DictionaryStub

{

public static void SortDictionary(Dictionary<int, string> dict){

SortedDictionary<int, string> sdict =

new SortedDictionary<int, string>();

foreach(KeyValuePair<int, string> pair in dict)

{

sdict.Add(pair.Key,pair.Value);

}

foreach(KeyValuePair<int, string> pair in sdict)

{

Console.WriteLine("Rank: {0} and Name: {1}",

pair.Key, pair.Value);

}

}

public static void Main(){

Dictionary<int,string> dict = new Dictionary<int,string>();

dict.Add(001,"Code");

dict.Add(004,"Decode");

dict.Add(002,"Fault");

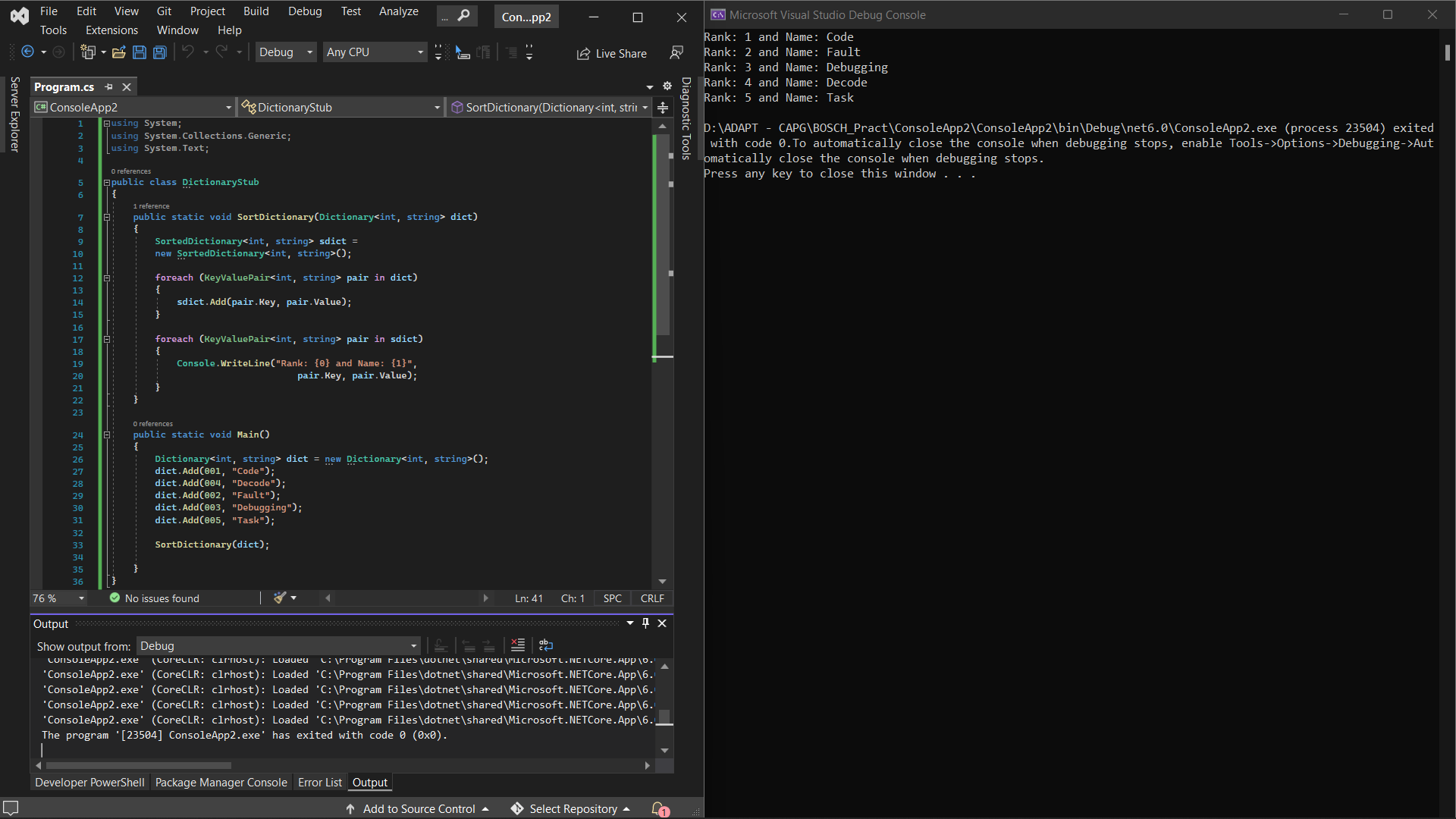
dict.Add(003,"Debugging");

dict.Add(005,"Task");

SortDictionary(dict);

}

}



**Exercise – 4**

//Remove last and add it to first.

using System;

using System.Collections.Generic;

using System.Text;

public class DictionaryStub

{

private static void Display(LinkedList<string> words, string test)

{

Console.WriteLine(test);

foreach (string word in words)

{

Console.Write(word + " ");

}

Console.WriteLine();

Console.WriteLine();

}

public static void RemoveLastAddFirst(LinkedList<string> words){

string last = words.Last.Value;

words.RemoveLast();

words.AddFirst(last);

}

public static void Main(){

string[] words = { "the", "fox", "jumps", "over", "the", "dog" };

LinkedList<string> sentence = new LinkedList<string>(words);

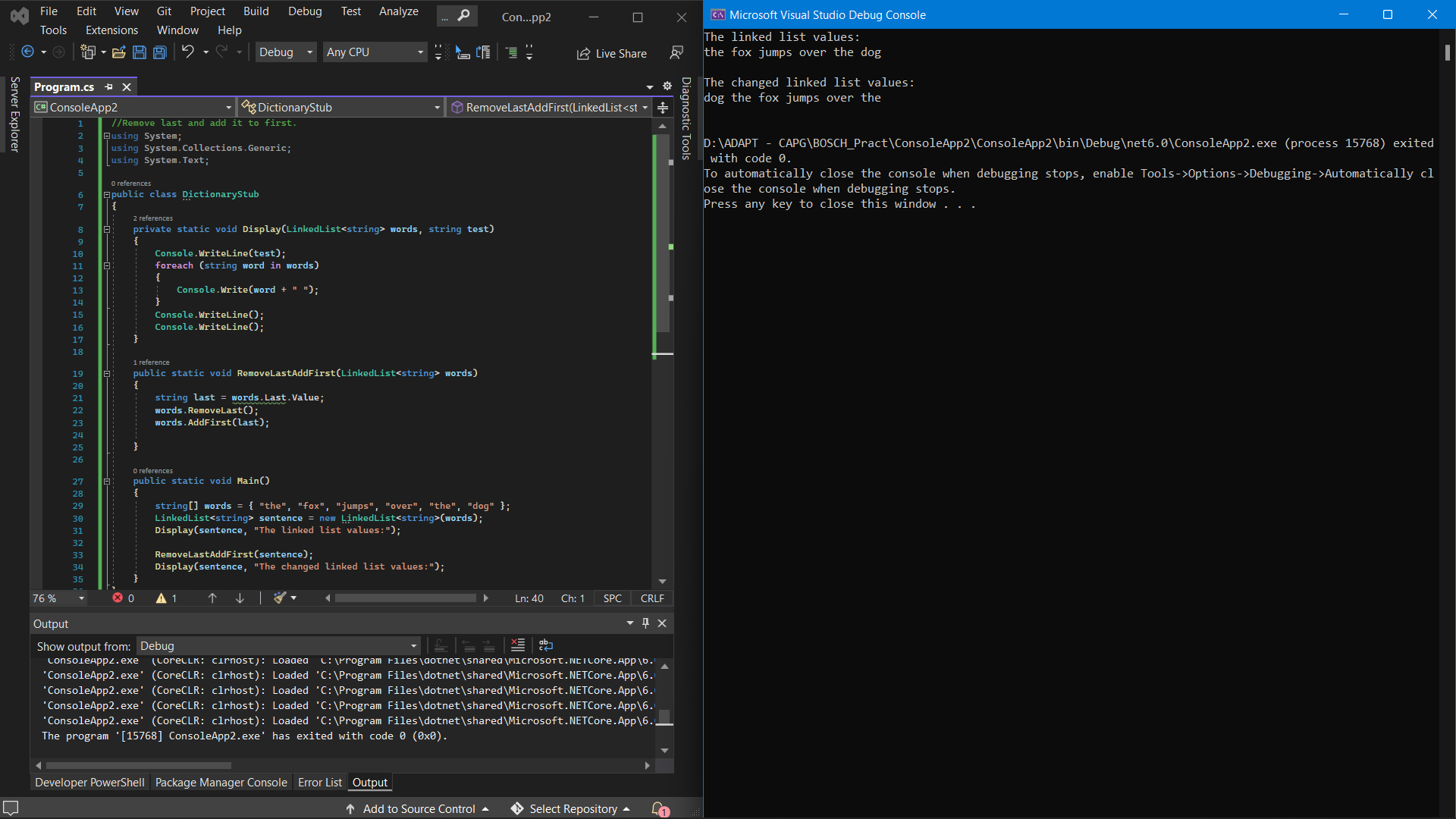
Display(sentence, "The linked list values:");

RemoveLastAddFirst(sentence);

Display(sentence, "The changed linked list values:");

}

}



**Exercise – 5**

using System;

using System.Collections;

using System.Text;

public class Exec{

public static void StackToQueue(Stack myStack){

int n = myStack.Count;

Queue myQueue = new Queue();

for(int i = 0;i<n;i++){

myQueue.Enqueue(myStack.Pop());

}

lock(myQueue.SyncRoot)

{

// foreach loop to display

// the elements in q1

foreach(Object i in myQueue)

Console.WriteLine(i);

}

}

public static void Main(){

Stack myStack = new Stack();

myStack.Push("E");

myStack.Push("D");

myStack.Push("C");

myStack.Push("B");

myStack.Push("A");

Console.WriteLine("Number of elements in the Stack: {0}",

myStack.Count);

StackToQueue(myStack);

}

}

