Assignment: Simple Generic and CompositionIn this assignment, you will create a simple C# program that utilizes generic and composition concepts. Your program should consist of two classes, each with its own properties and methods.**The first class** should be a generic class that can store any type of data (e.g., integers, strings, etc.). This class should have the following properties and methods:Properties:Capacity: an integer value that determines the maximum number of items that the class can storeData: an array of the generic data type that the class can storeCount: an integer value that represents the current number of items in the Data arrayMethods:AddData: a method that adds a new item to the Data arrayRemoveData: a method that removes an item from the Data arrayGetData: a method that returns the entire Data array

**The second class** should be a composed class that uses the generic class to store a specific type of data (e.g., strings). This class should have the following properties and methods:Properties:Capacity: an integer value that determines the maximum number of strings that the class can storeStringData: an array of strings that stores the data as stringsCount: an integer value that represents the current number of strings in the StringData arrayMethods:AddString: a method that adds a new string to the StringData array by using the AddData method of the generic classRemoveString: a method that removes a string from the StringData array by using the RemoveData method of the generic classGetStringData: a method that returns the entire StringData array by using the GetData method of the generic classYour program should demonstrate the use of both classes by performing the following actions:Create an instance of the composed class that uses the generic class to store strings with a capacity of 5.Add several strings to the StringData array using the AddString method.Display the entire StringData array using the GetStringData method.Remove one of the strings from the StringData array using the RemoveString method.Display the updated StringData array using the GetStringData method.

using System;

public class GenericClass<T>

{

private T[] data;

private int count;

private int capacity;

public GenericClass(int capacity)

{

this.capacity = capacity;

data = new T[capacity];

count = 0;

}

public void AddData(T item)

{

if (count < capacity)

{

data[count] = item;

count++;

}

else

{

Console.WriteLine("Error: The array is already at maximum capacity.");

}

}

public void RemoveData(T item)

{

int index = Array.IndexOf(data, item);

if (index >= 0)index

{

for (int i = index; i < count - 1; i++)

{

data[i] = data[i + 1];

}

count--;

}

else

{

Console.WriteLine("Error: The item was not found in the array.");

}

}

public T[] GetData()

{

return data;

}

}

public class StringDataClass

{

private GenericClass<string> data;

public StringDataClass(int capacity)

{

data = new GenericClass<string>(capacity);

}

public void AddString(string item)

{

data.AddData(item);

}

public void RemoveString(string item)

{

data.RemoveData(item);

}

public string[] GetStringData()

{

return data.GetData();

}

}

public class Program

{

public static void Main()

{

StringDataClass stringData = new StringDataClass(5);

stringData.AddString("Hello");

stringData.AddString("World");

stringData.AddString("C#");

stringData.AddString("Programming");

stringData.AddString("Assignment");

Console.WriteLine("Initial StringData array:");

foreach (string s in stringData.GetStringData())

{

Console.WriteLine(s);

}

stringData.RemoveString("C#");

Console.WriteLine("Updated StringData array:");

foreach (string s in stringData.GetStringData())

{

Console.WriteLine(s);

}

}

}

using System;

using System.Collections;

public class GenericClass<T>

{

public int Capacity { get; set; }

public ArrayList Data { get; set; }

public int Count { get { return Data.Count; } }

public GenericClass(int capacity)

{

Capacity = capacity;

Data = new ArrayList();

}

public void AddData(T item)

{

if (Count < Capacity)

{

Data.Add(item);

}

else

{

throw new IndexOutOfRangeException("Data array is full");

}

}

public void RemoveData(T item)

{

if (Data.Contains(item))

{

Data.Remove(item);

}

else

{

throw new ArgumentException("Data array does not contain the item");

}

}

public T[] GetData()

{

return (T[])Data.ToArray(typeof(T));

}

}

public class ComposedClass

{

public int Capacity { get; set; }

public GenericClass<string> StringData { get; set; }

public int Count { get { return StringData.Count; } }

public ComposedClass(int capacity)

{

Capacity = capacity;

StringData = new GenericClass<string>(capacity);

}

public void AddString(string item)

{

StringData.AddData(item);

}

public void RemoveString(string item)

{

StringData.RemoveData(item);

}

public string[] GetStringData()

{

return StringData.GetData();

}

}

class Program

{

static void Main(string[] args)

{

ComposedClass composedClass = new ComposedClass(5);

// add strings

composedClass.AddString("apple");

composedClass.AddString("banana");

composedClass.AddString("cherry");

// display all strings

Console.WriteLine("All strings:");

foreach (string s in composedClass.GetStringData())

{

Console.WriteLine(s);

}

// remove a string

composedClass.RemoveString("banana");

// display updated strings

Console.WriteLine("Updated strings:");

foreach (string s in composedClass.GetStringData())

{

Console.WriteLine(s);

}

}

}