Demonstrate the ICharQ interface: A character queue interface

```
/*
C# A Beginner's Guide
By Schildt

Publisher: Osborne McGraw-Hill
ISBN: 0072133295
*/
/*
Project 9-1

Demonstrate the ICharQ interface.
*/
using System;

// A character queue interface.
public interface ICharQ {
// Put a characer into the queue.
void put(char ch);

// Get a character from the queue.
char get();
}

// A fixed-size queue class for characters.
class FixedQueue : ICharQ {
char[] q; // this array holds the queue
int putloc, getloc; // the put and get indices

// Construct an empty queue given its size.
public FixedQueue(int size) {
q = new char[size+1]; // allocate memory for queue
putloc = getloc = 0;
```

```
}

// Put a character into the queue.
public void put(char ch) {
if(putloc==q.Length-1) {
Console.WriteLine(” – Queue is full.”);
return;
}

putloc++;
q[putloc] = ch;
}

// Get a character from the queue.
public char get() {
if(getloc == putloc) {
Console.WriteLine(” – Queue is empty.”);
return (char) 0;
}

getloc++;
return q[getloc];
}
}

// A circular queue.
class CircularQueue : ICharQ {
char[] q; // this array holds the queue
int putloc, getloc; // the put and get indices

// Construct an empty queue given its size.
public CircularQueue(int size) {
q = new char[size+1]; // allocate memory for queue
putloc = getloc = 0;
}

// Put a character into the queue.
public void put(char ch) {
/* Queue is full if either putloc is one less than
```

```
getloc, or if putloc is at the end of the array
and getloc is at the beginning. */
if(putloc+1==getloc |
((putloc==q.Length-1) & (getloc==0))) {
Console.WriteLine(” – Queue is full.”);
return;
}

putloc++;
if(putloc==q.Length) putloc = 0; // loop back
q[putloc] = ch;
}

// Get a character from the queue.
public char get() {
if(getloc == putloc) {
Console.WriteLine(” – Queue is empty.”);
return (char) 0;
}

getloc++;
if(getloc==q.Length) getloc = 0; // loop back
return q[getloc];
}
}

// A dynamic queue.
class DynQueue : ICharQ {
char[] q; // this array holds the queue
int putloc, getloc; // the put and get indices

// Construct an empty queue given its size.
public DynQueue(int size) {
q = new char[size+1]; // allocate memory for queue
putloc = getloc = 0;
}

// Put a character into the queue.
public void put(char ch) {
```

```
if(putloc==q.Length-1) {
// increase queue size
char[] t = new char[q.Length * 2];

// copy elements into new queue
for(int i=0; i < q.Length; i++) t[i] = q[i]; q = t; } putloc++; q[putloc] = ch; } // Get a character from the queue. public char get() { if(getloc == putloc) { Console.WriteLine(" -- Queue is empty."); return (char) 0; } getloc++; return q[getloc]; } } // Demonstrate the queues. public class IQDemo { public static void Main() { FixedQueue q1 = new FixedQueue(10); DynQueue q2 = new DynQueue(5); CircularQueue q3 = new CircularQueue(10); ICharQ iQ; char ch; int i; iQ = q1; // Put some characters into fixed queue. for(i=0; i < 10; i++) iQ.put((char) ('A' + i)); // Show the queue. Console.Write("Contents of fixed queue: "); for(i=0; i < 10; i++) { ch = iQ.get(); Console.Write(ch); } Console.WriteLine(); iQ = q2; // Put some characters into dynamic queue. for(i=0; i < 10; i++) iQ.put((char) ('Z' - i)); // Show the queue. Console.Write("Contents of dynamic queue: "); for(i=0; i < 10; i++) { ch = iQ.get(); Console.Write(ch); } Console.WriteLine(); iQ = q3; // Put some characters into circular queue. for(i=0; i < 10; i++) iQ.put((char) ('A' + i)); // Show the queue. Console.Write("Contents of circular queue: "); for(i=0; i < 10; i++) { ch = iQ.get(); Console.Write(ch); } Console.WriteLine(); // Put more characters into circular queue. for(i=10; i < 20; i++) iQ.put((char) ('A' + i)); // Show the queue. Console.Write("Contents of circular queue: "); for(i=0; i < 10; i++) { ch = iQ.get(); Console.Write(ch); } Console.WriteLine(" Store and consume from" + " circular queue."); // Use and consume from circular queue. for(i=0; i < 20; i++) { iQ.put((char) ('A' + i)); ch = iQ.get(); Console.Write(ch); } } } [/csharp]
```