КПІ ім. Ігоря Сікорського

Кафедра ІПІ

ЗВІТ

про виконання комп’ютерного практикуму № 6

з кредитного модуля

«Основи програмування-2. Методології програмування»

Тема: Структури даних

Варіант №3

Виконала:

студентка 1-го курсу

гр. ІП-з21 ФІОТ

Гавриленко Даяна Юріївна

Київ 2023

1. Умова завдання:

Изображение выглядит как текст, снимок экрана, Шрифт, алгебра

Автоматически созданное описание

1. Текст програми на мові C#:

Program.cs

using System;  
using System.Collections.Generic;  
using Lab6;  
  
public class Program  
{  
 static void Main(string[] args)  
 {  
 ConsoleManager consoleManager = new ConsoleManager();  
 CircularQueue<int> circularQueue = new CircularQueue<int>(5);  
  
 Console.WriteLine($"Queue is empty? {circularQueue.IsEmpty}");  
  
 consoleManager.AddElementToQueue(circularQueue, 4);  
 Console.WriteLine("Queue elements after adding a new element:");  
 circularQueue.DisplayInternal();  
  
 circularQueue.Dequeue();  
 Console.WriteLine("Queue elements after removing a new element:");  
 circularQueue.DisplayInternal();  
  
 consoleManager.AddElementToQueue(circularQueue, 2);  
 Console.WriteLine("Queue elements after adding a new element:");  
 circularQueue.DisplayInternal();  
   
 Console.WriteLine("Queue elements using iterator:");  
 foreach (int item in circularQueue)  
 {  
 Console.Write($"{item} ");  
 }  
  
 Console.WriteLine();  
   
 Console.WriteLine($"Queue is empty? {circularQueue.IsEmpty}");  
  
 Console.WriteLine($"\nClear the queue.");  
 circularQueue.Clear();  
 Console.WriteLine($"Queue is empty? {circularQueue.IsEmpty}");  
 }  
}

ConsoleManager.cs

namespace Lab6;  
  
public class ConsoleManager  
{  
 private readonly Random \_random = new Random();  
 public void AddElementToQueue(CircularQueue<int> queue, int count) // додавання нових елементів у чергу  
 {  
 for (int i = 1; i <= count; i++)  
 {  
 int randomNumber = \_random.Next(1, 100);  
 queue.Enqueue(randomNumber);  
 // queue.DisplayInternal();  
 }  
 }  
}

CircularQueue.cs

using System.Collections;  
using System.Text;  
  
namespace Lab6;  
  
public class CircularQueue<T> : IEnumerable<T>  
{  
 private T?[] \_items;  
 private int \_frontIndex = 0;  
 private int \_rearIndex = -1;  
  
 private readonly int \_max;  
  
 private int \_count = 0;  
  
 public int FrontIndex => \_frontIndex;  
 public int RearIndex => \_rearIndex;  
 public bool IsEmpty => \_count == 0; // перевірка черги на пустоту  
 public CircularQueue(int size)  
 {  
 \_items = new T[size];  
 \_max = size;  
 }  
  
 public void Enqueue(T item) // додавання нового елемента у чергу  
 {  
 if (\_count == \_max)  
 MoveToNext(ref \_frontIndex, \_max);  
  
 MoveToNext(ref \_rearIndex, \_max);  
 \_items[\_rearIndex] = item;  
  
 if (\_count < \_max)  
 \_count++;  
 }  
   
 public T? Dequeue() // видалення елемента із черги  
 {  
 if (\_count == 0)  
 throw new InvalidOperationException("Queue is empty.");  
  
 T? elem = \_items[\_frontIndex];  
  
 \_items[\_frontIndex] = default;  
  
 MoveToNext(ref \_frontIndex, \_max);  
  
 \_count--;  
  
 return elem;  
 }  
   
 public void Clear() // очищення черги  
 {  
 \_items = new T[\_max];  
 \_count = 0;  
 \_frontIndex = 0;  
 \_rearIndex = -1;  
 }  
  
 public void DisplayInternal() // вивід черги на консоль  
 {  
 for (int i = 0; i < \_items.Length; i++)  
 {  
 StringBuilder sb = new StringBuilder();  
 sb.Append($"[{\_items[i]}]");  
 if (i == \_frontIndex)  
 sb.Append('F');  
 if (i == \_rearIndex)  
 sb.Append('R');  
  
 Console.WriteLine(sb.ToString());  
 }  
 Console.WriteLine();  
 }  
  
 public IEnumerator<T> GetEnumerator() // ітератор для доступу до елементів черги  
 {  
 int index = \_frontIndex;  
 for (int count = 0; count < \_count; count++)  
 {  
 if (\_items[index] != null)  
 yield return \_items[index]!;  
  
 index = (index + 1) % \_items.Length;  
 }  
 }  
   
 private void MoveToFront(int limit) // установка початку черги  
 {  
 \_frontIndex = 0;  
 }  
  
 private void MoveToRear(int count) // установка на кінець черги  
 {  
 \_rearIndex = count - 1;  
 }  
  
 private void MoveToNext(ref int value, int limit) // перехід до наступного елемента черги  
 {  
 value = (value + 1) % limit;  
 }  
  
 private void MoveToPrevious(int limit) // перехід до попереднього елемента черги  
 {  
 \_frontIndex = (\_frontIndex - 1 + limit) % limit;  
 }  
   
 IEnumerator IEnumerable.GetEnumerator()  
 {  
 return GetEnumerator();  
 }  
}

1. Відеокопія результатів роботи програми:

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание