Министерство образования и науки Украины

Национальный аэрокосмический университет

им. Н.Е. Жуковского «ХАИ»

Кафедра №302

Отчет по лабораторной работе №6

**По предмету:** «ООП»

**Тема:** «Обработка исключительных ситуаций в C#»

**Выполнил:**

студент группы 316ст

Волковский В.В.

**Проверил:**

Момот М.А.

Харьков 2014

**Постановка задачи**

Для задания из Лабораторной работы № 4, добавить обработку исключений при возникновении ошибок. Обработку реализовать на уровне методов и демонстрационной программы. Обработать некоторые стандартные исключения. Создать свой собственный класс исключения и обработать его.

**Теоретические сведенья**

Блок [try](http://msdn.microsoft.com/ru-ru/library/0yd65esw.aspx) используется программистами C# для разбиения на разделы кода, который может затрагиваться исключением. Связанные с ним блоки [поймать](http://msdn.microsoft.com/ru-ru/library/0yd65esw.aspx) используются для обработки возможных исключений. Блок [finally](http://msdn.microsoft.com/ru-ru/library/zwc8s4fz.aspx), содержащий код, выполняемый вне зависимости от того, вызвано ли исключение в блоке **try**, например освобождение ресурсов, выделенных блоку **try**. Блоку **try** требуется один или несколько связанных блоков **catch** или блок **finally** (либо и то, и другое).

**Исходный код**

Program.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace DEQUE

{

class TimeException : Exception

{

public override string Message

{

get

{

return "Ошибка, неправильный формат ввода времени";

}

}

}

class Program

{

//Template for integer value

static Deque<int> deque = new Deque<int>();

static void Input()

{

Console.Clear();

Console.Write("Count = ");

int n = Convert.ToInt32(Console.ReadLine());

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

{

Console.Write("mas[" + (i + 1) + "]=");

deque.push\_e(Convert.ToInt32(Console.ReadLine()));

}

Console.WriteLine("Ok!");

Console.ReadLine();

}

static void Push\_f()

{

try

{

List<int> a = deque.getList();

Console.Clear();

Console.WriteLine("Old mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i]);

Console.Write("Push input element: ");

deque.push\_f(Convert.ToInt32(Console.ReadLine()));

a = deque.getList();

Console.WriteLine("New mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i]);

Console.ReadLine();

}

catch(FormatException)

{

Console.Clear();

Console.WriteLine("Ошибка, возможен ввод только целочисленных значений");

Console.ReadLine();

}

}

static void Push\_e()

{

List<int> a = deque.getList();

Console.Clear();

Console.WriteLine("Old mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i]);

Console.Write("Push input element: ");

deque.push\_e(Convert.ToInt32(Console.ReadLine()));

a = deque.getList();

Console.WriteLine("New mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i]);

Console.ReadLine();

}

static void Pop\_e()

{

List<int> a = deque.getList();

Console.Clear();

Console.WriteLine("Old mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i]);

Console.WriteLine("Poped last element " + deque.pop\_e());

a = deque.getList();

Console.WriteLine("New mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i]);

Console.ReadLine();

}

static void Pop\_f()

{

List<int> a = deque.getList();

Console.Clear();

Console.WriteLine("Old mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i]);

Console.WriteLine("Poped first element " + deque.pop\_f());

a = deque.getList();

Console.WriteLine("New mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i]);

Console.ReadLine();

}

static void Sort()

{

List<int> a = deque.getList();

Console.Clear();

Console.WriteLine("Old mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i]);

deque.SortDesc();

a = deque.getList();

Console.WriteLine("New mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i]);

Console.ReadLine();

}

static void Rotate()

{

List<int> a = deque.getList();

Console.Clear();

Console.WriteLine("Old mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i]);

Console.Write("Number of rotates: ");

deque.Rotate(Convert.ToInt32(Console.ReadLine()));

a = deque.getList();

Console.WriteLine("New mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i]);

Console.ReadLine();

}

static void Replace()

{

List<int> a = deque.getList();

Console.Clear();

Console.WriteLine("Old mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i]);

Console.Write("Position replacement: ");

int k = Convert.ToInt32(Console.ReadLine());

k--;

Console.Write("Replace number: ");

deque.Replace(Convert.ToInt32(Console.ReadLine()), k);

a = deque.getList();

Console.WriteLine("New mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i]);

Console.ReadLine();

}

static void Reverse()

{

List<int> a = deque.getList();

Console.Clear();

Console.WriteLine("Old mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i]);

deque.Reverse();

a = deque.getList();

Console.WriteLine("New mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i]);

Console.ReadLine();

}

static void Remove()

{

List<int> a = deque.getList();

Console.Clear();

Console.WriteLine("Old mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i]);

Console.Write("Remove position: ");

deque.Remove(Convert.ToInt32(Console.ReadLine()) - 1);

a = deque.getList();

Console.WriteLine("New mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i]);

Console.ReadLine();

}

static void Fill()

{

List<int> a = deque.getList();

Console.Clear();

Console.WriteLine("Old mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i]);

Console.Write("Fill value: ");

deque.Fill(Convert.ToInt32(Console.ReadLine()));

a = deque.getList();

Console.WriteLine("New mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i]);

Console.ReadLine();

}

static void Unique()

{

List<int> a = deque.getList();

Console.Clear();

Console.WriteLine("Number of array elements: " + deque.Count());

Console.WriteLine("Old mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i]);

deque.Unique();

a = deque.getList();

Console.WriteLine("New mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i]);

Console.ReadLine();

}

static void isEmpty()

{

List<int> a = deque.getList();

Console.Clear();

Console.WriteLine("isEmpty: " + deque.isEmpty());

Console.WriteLine("Old mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i]);

deque.Clear();

a = deque.getList();

Console.WriteLine("isEmpty: " + deque.isEmpty());

Console.WriteLine("New mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i]);

Console.ReadLine();

}

//Template for Time value

static Deque<Time> dequeTime = new Deque<Time>();

static void TInput()

{

Console.Clear();

Console.Write("Count = ");

int n = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Enter the time in the format (hh:mm:ss):");

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

{

Console.Write("mas[" + (i + 1) + "]=");

string sTime = Console.ReadLine();

dequeTime.push\_e(new Time(

Convert.ToInt32(sTime.Substring(0, 2)),

Convert.ToInt32(sTime.Substring(3, 2)),

Convert.ToInt32(sTime.Substring(6, 2))

)); }

Console.WriteLine("Ok!");

Console.ReadLine();

}

static void TPush\_f()

{

try

{

List<Time> a = dequeTime.getList();

Console.Clear();

Console.WriteLine("Old mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i].getAll());

Console.Write("Push input element: ");

string sTime = Console.ReadLine();

bool f = false;

Time t = null;

try

{

t = new Time(

Convert.ToInt32(sTime.Substring(0, 2)),

Convert.ToInt32(sTime.Substring(3, 2)),

Convert.ToInt32(sTime.Substring(6, 2))

);

}

catch

{

f = true;

}

if (sTime[2] == ':' && sTime[5] == ':' && sTime.Length == 8) f = false;

else f = true;

if (f) throw new TimeException();

else dequeTime.push\_f(t);

a = dequeTime.getList();

Console.WriteLine("New mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i].getAll());

Console.ReadLine();

}

catch(TimeException ex)

{

Console.Clear();

Console.WriteLine(ex.Message);

Console.ReadLine();

} }

static void TPush\_e()

{

List<Time> a = dequeTime.getList();

Console.Clear();

Console.WriteLine("Old mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i].getAll());

Console.Write("Push input element: ");

string sTime = Console.ReadLine();

dequeTime.push\_e(new Time(

Convert.ToInt32(sTime.Substring(0, 2)),

Convert.ToInt32(sTime.Substring(3, 2)),

Convert.ToInt32(sTime.Substring(6, 2))

));

a = dequeTime.getList();

Console.WriteLine("New mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i].getAll());

Console.ReadLine();

}

static void TPop\_e()

{

List<Time> a = dequeTime.getList();

Console.Clear();

Console.WriteLine("Old mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i].getAll());

Console.WriteLine("Poped last element " + dequeTime.pop\_e().getAll());

a = dequeTime.getList();

Console.WriteLine("New mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i].getAll());

Console.ReadLine();

}

static void TPop\_f()

{

List<Time> a = dequeTime.getList();

Console.Clear();

Console.WriteLine("Old mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i].getAll());

Console.WriteLine("Poped first element " + dequeTime.pop\_f().getAll());

a = dequeTime.getList();

Console.WriteLine("New mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i].getAll());

Console.ReadLine();

}

static void TSort()

{

List<Time> a = dequeTime.getList();

Console.Clear();

Console.WriteLine("Old mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i].getAll());

dequeTime.SortDesc();

a = dequeTime.getList();

Console.WriteLine("New mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i].getAll());

Console.ReadLine();

}

static void TRotate()

{

List<Time> a = dequeTime.getList();

Console.Clear();

Console.WriteLine("Old mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i].getAll());

Console.Write("Number of rotates: ");

dequeTime.Rotate(Convert.ToInt32(Console.ReadLine()));

a = dequeTime.getList();

Console.WriteLine("New mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i].getAll());

Console.ReadLine();

}

static void TReplace()

{

List<Time> a = dequeTime.getList();

Console.Clear();

Console.WriteLine("Old mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i].getAll());

Console.Write("Position replacement: ");

int k = Convert.ToInt32(Console.ReadLine());

k--;

Console.Write("Replace time: ");

string sTime = Console.ReadLine();

dequeTime.Replace(new Time(

Convert.ToInt32(sTime.Substring(0, 2)),

Convert.ToInt32(sTime.Substring(3, 2)),

Convert.ToInt32(sTime.Substring(6, 2))

), k);

a = dequeTime.getList();

Console.WriteLine("New mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i].getAll());

Console.ReadLine();

}

static void TReverse()

{

List<Time> a = dequeTime.getList();

Console.Clear();

Console.WriteLine("Old mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i].getAll());

dequeTime.Reverse();

a = dequeTime.getList();

Console.WriteLine("New mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i].getAll());

Console.ReadLine();

}

static void TRemove()

{

List<Time> a = dequeTime.getList();

Console.Clear();

Console.WriteLine("Old mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i].getAll());

Console.Write("Remove position: ");

dequeTime.Remove(Convert.ToInt32(Console.ReadLine()) - 1);

a = dequeTime.getList();

Console.WriteLine("New mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i].getAll());

Console.ReadLine();

}

static void TFill()

{

List<Time> a = dequeTime.getList();

Console.Clear();

Console.WriteLine("Old mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i].getAll());

Console.Write("Fill time: ");

string sTime = Console.ReadLine();

dequeTime.Fill(new Time(

Convert.ToInt32(sTime.Substring(0, 2)),

Convert.ToInt32(sTime.Substring(3, 2)),

Convert.ToInt32(sTime.Substring(6, 2))

));

a = dequeTime.getList();

Console.WriteLine("New mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i].getAll());

Console.ReadLine();

}

static void TUnique()

{

List<Time> a = dequeTime.getList();

Console.Clear();

Console.WriteLine("Number of array elements: " + dequeTime.Count());

Console.WriteLine("Old mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i].getAll());

dequeTime.Unique();

a = dequeTime.getList();

Console.WriteLine("New mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i].getAll());

Console.ReadLine();

}

static void TisEmpty()

{

List<Time> a = dequeTime.getList();

Console.Clear();

Console.WriteLine("isEmpty: " + dequeTime.isEmpty());

Console.WriteLine("Old mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i].getAll());

dequeTime.Clear();

a = dequeTime.getList();

Console.WriteLine("isEmpty: " + dequeTime.isEmpty());

Console.WriteLine("New mas: ");

for (int i = 0; i < a.Count(); i++) Console.WriteLine("a[" + (i + 1) + "]=" + a[i].getAll());

Console.ReadLine();

}

static void menuInt()

{

bool b = true;

while (b)

{

Console.Clear();

Console.WriteLine("Int");

Console.WriteLine("1 - Input elements");

Console.WriteLine("2 - push\_f()");

Console.WriteLine("3 - push\_e()");

Console.WriteLine("4 - pop\_f()");

Console.WriteLine("5 - pop\_e()");

Console.WriteLine("6 - sort()");

Console.WriteLine("7 - rotate()");

Console.WriteLine("8 - replace()");

Console.WriteLine("9 - reverse()");

Console.WriteLine("10 - remove()");

Console.WriteLine("11 - fill()");

Console.WriteLine("12 - unique(), count()");

Console.WriteLine("13 - is\_empty(), clear()");

Console.WriteLine("0 - Exit");

Console.Write("Your choise: ");

int choice = Convert.ToInt32(Console.ReadLine());

switch (choice)

{

case 1: Input(); break;

case 2: Push\_f(); break;

case 3: Push\_e(); break;

case 4: Pop\_f(); break;

case 5: Pop\_e(); break;

case 6: Sort(); break;

case 7: Rotate(); break;

case 8: Replace(); break;

case 9: Reverse(); break;

case 10: Remove(); break;

case 11: Fill(); break;

case 12: Unique(); break;

case 13: isEmpty(); break;

case 0: b = false; break;

default:

Console.Clear();

Console.WriteLine("Wrong operation!");

Console.ReadLine();

break;

} } }

static void menuTime()

{

bool b = true;

while (b)

{

Console.Clear();

Console.WriteLine("Time");

Console.WriteLine("1 - Input elements");

Console.WriteLine("2 - push\_f()");

Console.WriteLine("3 - push\_e()");

Console.WriteLine("4 - pop\_f()");

Console.WriteLine("5 - pop\_e()");

Console.WriteLine("6 - sort()");

Console.WriteLine("7 - rotate()");

Console.WriteLine("8 - replace()");

Console.WriteLine("9 - reverse()");

Console.WriteLine("10 - remove()");

Console.WriteLine("11 - fill()");

Console.WriteLine("12 - unique(), count()");

Console.WriteLine("13 - is\_empty(), clear()");

Console.WriteLine("0 - Exit");

Console.Write("Your choise: ");

int choice = Convert.ToInt32(Console.ReadLine());

switch (choice)

{

case 1: TInput(); break;

case 2: TPush\_f(); break;

case 3: TPush\_e(); break;

case 4: TPop\_f(); break;

case 5: TPop\_e(); break;

case 6: TSort(); break;

case 7: TRotate(); break;

case 8: TReplace(); break;

case 9: TReverse(); break;

case 10: TRemove(); break;

case 11: TFill(); break;

case 12: TUnique(); break;

case 13: TisEmpty(); break;

case 0: b = false; break;

default:

Console.Clear();

Console.WriteLine("Wrong operation!");

Console.ReadLine();

break;

} } }

static void Main(string[] args)

{

bool b = true;

while(b)

{

Console.Clear();

Console.WriteLine("1 - Int");

Console.WriteLine("2 - Time");

Console.WriteLine("0 - Exit");

Console.Write("Your choise: ");

int choice = Convert.ToInt32(Console.ReadLine());

switch (choice)

{

case 1: menuInt(); break;

case 2: menuTime(); break;

case 0: b = false; break;

default:

Console.Clear();

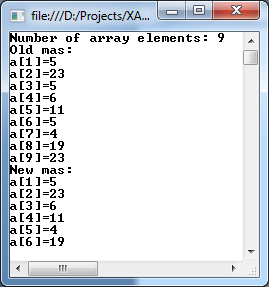
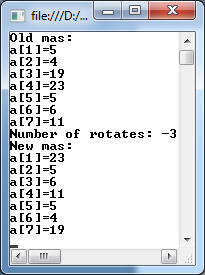
Console.WriteLine("Wrong operation!");

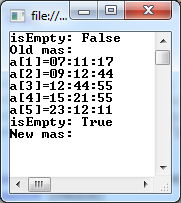
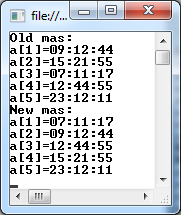
Console.ReadLine();

break;

} } } }}

**Скриншоты работы программы**





**Вывод**

На данной лабораторной работе я закрепил теоретические знания, а также приобрел практические навыки по реализации шаблонов в программировании.