Ministerul Educației al Republicii Moldova

Universitatea Tehnică a Moldovei

Facultatea Calculatoare, Informatică și Microelectronică

Departamentul Automatică și Inginerie Software

**RAPORT**

**Disciplina:** Analiza, proiectarea și programarea orientată pe obiecte

Lucrare de laborator Nr. 3

**Tema:** Tipurile de date generice in C#

A efectuat : studentul gr. TI-143 Lutenco Petru

A verificat: lector superior Balan Mihaela

Chișinău 2015

**1.Scopurile lucrării**

Însușirea tehnologiei de programare folosind tipurile de date generice in limbajul C#.

**2.Sarcina lucrării**

**Programul 1:**

1.Crearea unui obiect-container de tip Dictionary

2.Vizualizarea containerului

3.Modificarea containerului, stergind din el careva elemente si schimarea a altora

4.Vizulizarea folosind iteratorul

5.Crearea celui de-al 2-lea container si popularea lui cu date

6.Stergerea din primul container n elemente si popularea lui cu elementele din cel de-al doilea

7.Vizualizarea ambelor containere

**Programul 2:**

1.Creati un container de tip Multimap cu date de tip utilizator

2.Sortarea elementelor

3.Vizualizarea containerului

4.Folosind algoritmul potrivit, sa se gaseasca elementele care satusfac conditiei

5.Copierea elementelor care satisfac conditia in un al doilea container de tip LinkedList

6.Vizualizarea celui de-al 2-lea container

7. Sortarea crescatoare a containerelor

8.Vizualizarea containrelor

9.Obtinerea celui de-al 3-lea container prin metoda alipirii celor doua

10.Vizualizarea celui de-al 3-lea container

11.Stabilirea nr de elemente care satisfac conditia din containeru 3

Pentru realizarea sarcinii am folosit clasa Student.

public class Student

{

public double NotaMedie;

public string Nume;

public Student() { }

public Student(string \_Nume, int \_Grupa)

{

Nume = \_Nume;

NotaMedie = \_Grupa;

}

public static Student ReadStudent(Student \_Student)

{

Console.Write("Introduceti Numele : ");

\_Student.Nume = Console.ReadLine();

Console.Write("Introduceti Nota Medie : ");

\_Student.NotaMedie = Convert.ToDouble(Console.ReadLine());

return \_Student;

}

public static void WriteStudent(Student \_Student)

{

Console.WriteLine("\nNumele : "+\_Student.Nume);

Console.WriteLine("Nota Medie : "+\_Student.NotaMedie+ "\n");

}

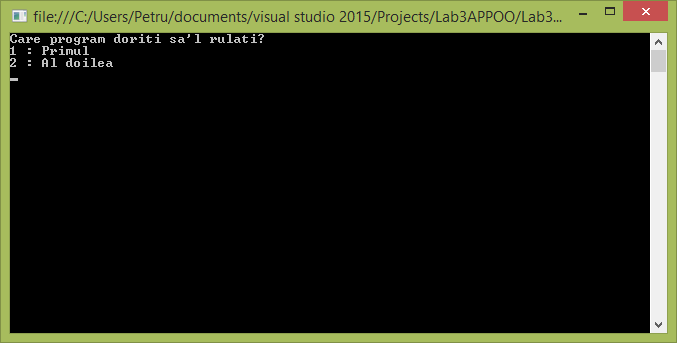
}

**3.Realizarea sarcinii**

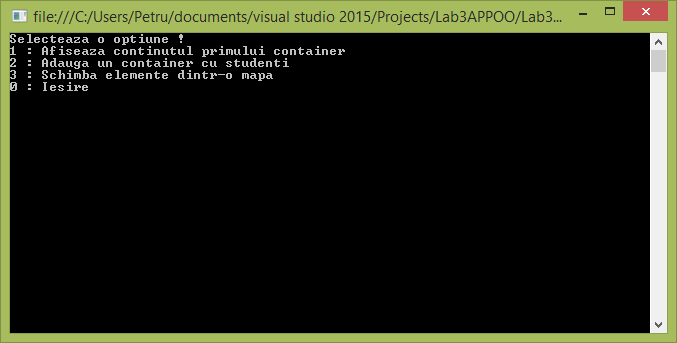
Codul sursa a programelor poate fi vizualizat in anexa.

Rezultatele programeleor

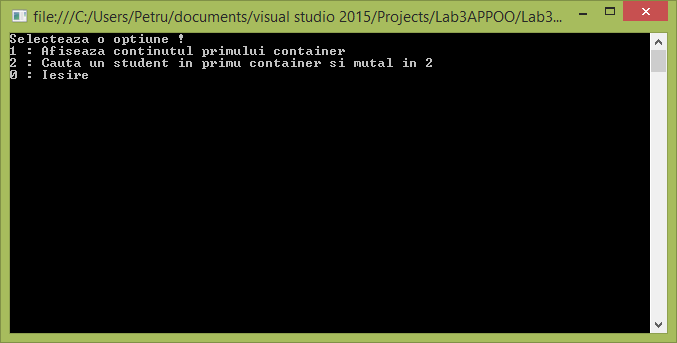
**Meniul de selectare a programului:**



**Program1:Meniul Principal**



**Program 2:Meniul Principal**



**Concluzie:**

In urma realizarii acestei lucrari de laborator am obtinut deprinderi de a lucra cu elemnte din limbajul C# si anume unele metode din biblioteca LINQ si unele tipuri de date generice ca Dictionary si LinkedList.

Anexa:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab3APPOO

{

class Program

{

Dictionary<Student, string> RemoveRangeAndAddSecondMap(Dictionary<Student, string> \_Dictionary1, Dictionary<Student, string> \_Dictionary2)

{

int ItemToDelete, ItemsToBeDeleted;

var names = ShowElements(\_Dictionary1);

Console.Clear();

var i = 0;

foreach (var item in names)

Console.WriteLine(i++ + " : "+item);

Console.WriteLine("De la care elemnet doriti sa incepeti stergerea? : ");

ItemToDelete = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Cite elemente doriti sa mai stergeti? : ");

ItemsToBeDeleted = Convert.ToInt32(Console.ReadLine());

if (ItemsToBeDeleted >= (\_Dictionary1.Count() - ItemToDelete))

{

Console.WriteLine("Nu puteti sterge mai multe elemente decit are mapa,\n Introduceti un numar mai mic! : ");

ItemsToBeDeleted = Convert.ToInt32(Console.ReadLine());

}

var \_itemFrist = \_Dictionary1.FirstOrDefault(el=>el.Key.Nume == names.ElementAtOrDefault(ItemToDelete));

var \_lastItem = \_Dictionary1.FirstOrDefault(el => el.Key.Nume == names.ElementAtOrDefault(ItemsToBeDeleted));

bool \_tempFlag = false;

foreach(var item in \_Dictionary1.ToList())

{

if (item.Key == \_itemFrist.Key) \_tempFlag = true;

if (\_tempFlag)

{

\_Dictionary1.Remove(item.Key);

}

if (item.Key == \_lastItem.Key) \_tempFlag = false;

}

\_Dictionary1.Remove(\_itemFrist.Key);

foreach(var item in \_Dictionary2)

\_Dictionary1.Add(item.Key,item.Value);

return \_Dictionary1;

}

public static Dictionary< Student, string> ModificaDictionar(Dictionary<Student,string> \_Dictionary)

{

int option = 1;

while (option != 0)

{

Console.Clear();

Console.WriteLine("Selecteaza o optiune!");

Console.WriteLine("1 : Stergeti datele despre un Student");

Console.WriteLine("2 : Modifica datele despre un Student");

Console.WriteLine("3 : Adauga datele despre un Student");

Console.WriteLine("0 : Inapoi");

option = Convert.ToInt32(Console.ReadLine());

switch (option)

{

case 1:

{

int ItemToDelete = 0;

var names = ShowElements(\_Dictionary);

Console.Clear();

var i = 0;

foreach (var item in names)

Console.WriteLine(i++ + " : " + item);

Console.Write("Sterge datele despre studentul : ");

ItemToDelete = Convert.ToInt32(Console.ReadLine());

\_Dictionary.Remove(\_Dictionary.FirstOrDefault(element => element.Key.Nume == names[ItemToDelete]).Key);

}

break;

case 2:

{

int ItemToModify = 0;

var names = ShowElements(\_Dictionary);

Console.Clear();

var i = 0;

foreach (var item in names)

Console.WriteLine(i++ + " : " + item);

Console.WriteLine("\nModificati datele despre studentul : ");

ItemToModify = Convert.ToInt32(Console.ReadLine());

var StudentToModify = \_Dictionary.FirstOrDefault(element => element.Key.Nume == names[ItemToModify]).Key;

StudentToModify = ModifyItem(StudentToModify);

ShowElements(\_Dictionary);

}

break;

case 3:

{

Student student = new Student();

student = Student.ReadStudent(student);

\_Dictionary.Add(student, student.Nume);

}

break;

}

}

return \_Dictionary;

}

private static Student ModifyItem(Student student)

{

int ItemToModify = 1;

while (ItemToModify != 0)

{

Console.Clear();

Student.WriteStudent(student);

Console.WriteLine("Alegeti o optiune!");

Console.WriteLine("1 : Modifica Numele");

Console.WriteLine("2 : Modifica Nota Medie");

Console.WriteLine("0 : Inapoi");

ItemToModify = Convert.ToInt32(Console.ReadLine());

switch (ItemToModify)

{

case 1:

{

Console.WriteLines("Introduceti noul nume : ");

student.Nume = Console.ReadLine();

}

break;

case 2:

{

Console.WriteLine("Introduceti Nota Medie : ");

student.NotaMedie = Convert.ToInt32(Console.ReadLine()) ;

}

break;

}

}

return student;

}

public static LinkedList<Student> FindElementsMove(Dictionary<Student, string> \_Dictionary)

{

string Nume;

LinkedList<Student> \_ListNewLocal = new LinkedList<Student>();

Console.WriteLine("Introduceti numele pentru cautare! : ");

Nume = Console.ReadLine();

foreach(var element in \_Dictionary)

{

if (element.Key.Nume == Nume)

{

Console.WriteLine("A fost gasit un element!");

Student.WriteStudent(element.Key);

Console.ReadKey();

\_ListNewLocal.AddLast(element.Key);

\_Dictionary.Remove(element.Key);

return \_ListNewLocal;

}

else if (element.Equals(\_Dictionary.Last()) && element.Key.Nume != Nume && \_ListNewLocal.Count == 0)

{

Console.WriteLine("Nu a fost gasit");

Console.ReadKey();

}

}

return \_ListNewLocal;

}

public static List<string> ShowElements(Dictionary<Student, string> \_Dictionary)

{

List<string> \_names = new List<string>();

foreach (var elements in \_Dictionary)

{

Student.WriteStudent(elements.Key);

\_names.Add(elements.Key.Nume);

}

return \_names;

}

public static void ShowElements(LinkedList<Student> \_List)

{

for(LinkedListNode<Student> \_it =\_List.First; \_it!=null; \_it= \_it.Next)

Student.WriteStudent(\_it.Value);

}

private Dictionary<Student, string> PopulateDictionary(Dictionary<Student, string> \_Dictionary)

{

int Counter = 1;

Console.Write("Introduceti elementele primului container(minimum 4)");

Console.Write("\n");

while (Counter != 0)

{

string \_tempGrupa;

Student \_temp = new Student();

Console.Write("Introduceti numarul grupei studentului : ");

\_tempGrupa = Console.ReadLine();

\_temp = Student.ReadStudent(\_temp);

\_Dictionary.Add(\_temp, "TI-" + \_tempGrupa);

Counter++;

if (Counter >= 5)

{

char \_decision;

Console.Write("Doriti sa mai introduceti un student? y/n");

\_decision = Convert.ToChar(Console.ReadLine());

if (\_decision == 'n' || \_decision == 'N')

{

Counter = 0;

}

}

}

return \_Dictionary;

}

static void Main(string[] args)

{

bool \_showChangeElementsP1 = true;

int \_Program = 0;

Console.Clear();

Program \_this = new Program();

Dictionary<Student, string> \_Dictionary = new Dictionary<Student, string>();

Dictionary<Student, string> \_Dictionary2 = new Dictionary<Student, string>();

\_Dictionary = \_this.PopulateDictionary(\_Dictionary);

Console.Clear();

Console.WriteLine("Care program doriti sa'l rulati?");

Console.WriteLine("1 : Primul");

Console.WriteLine("2 : Al doilea");

\_Program = Convert.ToInt32(Console.ReadLine());

switch (\_Program)

{

case 2:

{

LinkedList<Student> \_List1 = new LinkedList<Student>();

LinkedList<Student> \_List2 = new LinkedList<Student>();

LinkedList<Student> \_List3 = new LinkedList<Student>();

LinkedList<Student> \_ListPromovati = new LinkedList<Student>();

int option = 1;

while (option != 0)

{

Console.Clear();

Console.WriteLine("Selecteaza o optiune !");

Console.WriteLine("1 : Afiseaza continutul primului container");

if (\_List1.Count == 0)

Console.WriteLine("2 : Cauta un student in primu container si mutal in 2 ");

else Console.WriteLine("2 : Afiseaza continutul celui de al doilea container ");

if (\_List1.Count > 0)

{

Console.WriteLine("3 : Sorteaza containerile dupa cresterea notei medii");

}

if (\_List1.Count > 0)

{

if (\_List2.Count == 0) Console.WriteLine("4 : Arunca toti studentii din ambele containere intr'unul singur ");

else Console.WriteLine("4 : Vizualizeaza continutul celui de al 3-lea container");

}

if (\_List2.Count > 0)

{

Console.WriteLine("5 : Cauta studenti promovati in containerul 3");

Console.WriteLine("6 : Cauta un student in containerul 3");

}

Console.Write("0 : Iesire");

option = Convert.ToInt32(Console.ReadLine());

switch (option)

{

case 1:

{

ShowElements(\_Dictionary);

Console.ReadKey();

}

break;

case 2:

{

if (\_List1.Count == 0)

{

\_List1 = FindElementsMove(\_Dictionary);

}

else

{

ShowElements(\_List1);

Console.ReadKey();

}

}

break;

case 3:

{

\_List1.OrderBy(obj => obj.NotaMedie);

\_Dictionary.OrderBy(obj => obj.Key.NotaMedie);

Console.WriteLine("Succes!");

Console.ReadKey();

}

break;

case 4:

{

if (\_List2.Count == 0)

{

foreach (var element in \_List1)

{

\_List2.AddFirst(element);

}

foreach (var element in \_Dictionary)

{

\_List2.AddFirst(element.Key);

}

}

else

{

ShowElements(\_List2);

Console.ReadKey();

}

}

break;

case 5:

{

if (\_List2.Count > 0)

{

\_List3 = new LinkedList<Student>(\_List2.Where(element => element.NotaMedie >= 5));

Console.WriteLine("{0} studenti sunt promovati!", \_List3.Count);

ShowElements(\_List3);

Console.ReadKey();

}

}

break;

case 6:

{

string Nume = string.Empty;

Console.WriteLine("Introduceti numele studentului : ");

Nume = Console.ReadLine();

var \_Gasit = \_List2.FirstOrDefault(element => element.Nume == Nume);

if (\_Gasit != null)

Student.WriteStudent(\_Gasit);

else Console.WriteLine("Nu a fost gasit nici un element");

Console.ReadKey();

}

break;

}

}

}

break;

case 1:

{

int option = 1;

while (option != 0)

{

Console.Clear();

Console.WriteLine("Selecteaza o optiune !");

Console.WriteLine("1 : Afiseaza continutul primului container");

if (\_Dictionary2.Count == 0)

Console.WriteLine("2 : Adauga un container cu studenti");

else

Console.WriteLine("2 : Afiseaza continutul celui de al doilea container ");

Console.WriteLine("3 : Schimba elemente dintr-o mapa");

if (\_Dictionary2.Count > 0 && \_showChangeElementsP1)

Console.WriteLine("4 : Sterge elemente din prima mapa si adauga elementele din mapa 2 in prima!");

Console.Write("0 : Iesire");

option = Convert.ToInt32(Console.ReadLine());

switch (option)

{

case 1:

{

ShowElements(\_Dictionary);

Console.ReadKey();

}

break;

case 2:

{

if (\_Dictionary2.Count == 0)

{

\_Dictionary2 = \_this.PopulateDictionary(\_Dictionary2);

}else

{

ShowElements(\_Dictionary2);

Console.ReadKey();

}

} break;

case 3:

{

var \_tempChoser = 0;

Console.Clear();

Console.WriteLine("Care container doriti sa'l modificati?");

Console.WriteLine("1 : Primul Container");

if(\_Dictionary2.Count != 0)Console.WriteLine("2 : Al doilea Container");

\_tempChoser = Convert.ToInt32(Console.ReadLine());

switch (\_tempChoser)

{

case 1: { ModificaDictionar(\_Dictionary); } break;

case 2: { if (\_Dictionary2.Count != 0) ModificaDictionar(\_Dictionary2); } break;

}

} break;

case 4:

{

if (\_Dictionary2.Count > 0 && \_showChangeElementsP1)

{

\_Dictionary = \_this.RemoveRangeAndAddSecondMap(\_Dictionary,\_Dictionary2);

\_showChangeElementsP1 = false;

}

} break;

}

}

} break;

}

}

}

}