МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение высшего образования

«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра «Информационные системы»

**Лабораторная работа №6**по дисциплине «Основы алгоритмизации и программирования»

Выполнил:  
студент(ка) гр. ПИбд-12 Дозорова Алена

Ульяновск  
2021

Задание на лабораторную работу:

использовать 5 разных структур данных/атд определённых в выбранном языке программирования

Краткое описание используемых алгоритмов и структур

Программа записывает информацию о семьях и составляет генеалогическое древо. Используются списки для записи вершин и ребер графа, граф, очередь для запоминания фамилий семей, структура как вершина графа и стек для хранения имен в семье.

Код лабораторной работы:

public class Graph

{

public struct Family

{

public string Sorname;

public int count;

public Stack<string> names;

public int index;

}

public List<Family> vertex = new List<Family>();

public List<string> eges = new List<string>();

int C = 0;

public Graph()

{

}

public void AddVertex(string name, int count, Stack<string> names)

{

Family f = new Family ();

f.Sorname = name;

f.count = count;

f.names = names;

f.index = C;

vertex.Add(f);

C++;

}

public void AddEdge(int index1, int index2)

{

eges.Add(index1 + ":"+index2);

}

public void print()

{

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

{

Console.WriteLine(vertex[i].Sorname+":");

for(int j = 0; j<eges.Count; j++)

{

string[] ind = eges[j].Split(':');

if(ind[0]==Convert.ToString(vertex[i].index))

{

Console.WriteLine(vertex[Convert.ToInt32(ind[1])].Sorname+" ");

}

}

}

}

public void print\_elem(string sornname)

{

Family family = new Family();

foreach(var f in vertex)

{

if (f.Sorname == sornname)

{

family = f;

break;

}

}

if (family.Sorname == "")

{

return;

}

Console.WriteLine("Фамилия: {0} Количество человек: {1} Имена:", family.Sorname, family.count);

while(family.names!=null)

{

Console.WriteLine(family.names.Pop());

}

}

}

static void Main(string[] args)

{

int k = 0;

Queue<string> sornames = new Queue<string>();

for (int i = 0; i<5; i++)

{

sornames.Enqueue(Console.ReadLine());

k++;

}

Graph g = new Graph();

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

{

Console.WriteLine("\nВведите количество человек: ");

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

Console.WriteLine("\nВведите имена через пробел: ");

string ch = Console.ReadLine();

string[] h = ch.Split(' ');

Stack<string> d = new Stack<string>();

foreach(string r in h)

{

d.Push(r);

}

g.AddVertex(sornames.Dequeue(), c, d);

Console.WriteLine("\nВведите номер семьи, с которой связана эта семья:");

try

{

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

g.AddEdge(y, i);

}

catch

{

}

}

g.print();

Console.WriteLine("Выведем информацию о первой семье: ");

g.print\_elem(g.vertex[0].Sorname);

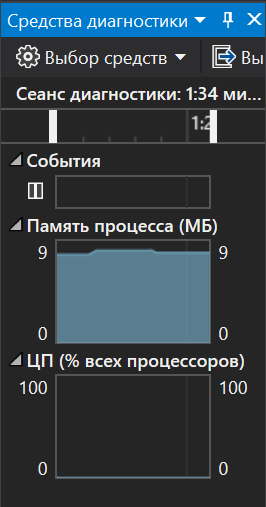
Console.ReadKey();

}

}

}

Результаты тестирования:



Выводы:

Я повторила изученный мной материал, реализовав эту программу.