Отчет по практической работе №16

Наименование работы: обобщения.

Индивидуальные задания (вариант №9):

Написать обобщённый класс SearchTree, реализующий бинарное дерево поиска с узлами, хранящими значения произвольного типа. Каждый узел должен хранить ссылку на родительский узел (для корня — нулевую ссылку). Класс должен содержать методы:

• Add — добавить элемент в дерево,

• ListFrom — найти узел с указанным значением и вернуть строку со списком значений узлов ветви от корня до найденного узла. Класс разместить в отдельной библиотеке. Написать программу, использующую его.

Ход работы:

1. Разработал программу в соответствии с заданием №1. Были реализованы методы, условия, классы, обобщения.

Листинг программы:

using System;

namespace Lab16.\_1

{

class Program

{

static void Main(string[] args)

{

SearchTree<string> t = new SearchTree<string>();

t.Insert("персик");

t.Insert("черника");

t.Insert("мандарин");

t.Insert("груша");

t.Insert("яблоко");

t.Insert("клубника");

Console.WriteLine(t.Display(t));

Console.WriteLine("Poisk^ ");

SearchTree<string> s = t.ListFrom("мандарин");

Console.WriteLine(s.Display(s));

Console.Read();

}

}

class SearchTree <T>

{

private T value;

private int count;

private SearchTree<T> left;

private SearchTree<T> right;

// вставка

public void Insert(T value)

{

if (this.value == null)

this.value = value;

else

{

if (this.value.ToString().CompareTo(value.ToString()) == 1)

{

if (left == null)

this.left = new SearchTree<T>();

left.Insert(value);

}

else if (this.value.ToString().CompareTo(value.ToString()) == -1)

{

if (right == null)

this.right = new SearchTree<T>();

right.Insert(value);

}

else

throw new Exception("Узел уже существует");

}

this.count = Recount(this);

}

// поиск

public SearchTree<T> ListFrom(T value)

{

if (this.value.ToString() == value.ToString())

return this;

else if (this.value.ToString().CompareTo(value.ToString()) == 1)

{

if (left != null)

return this.left.ListFrom(value);

else

throw new Exception("Искомого узла в дереве нет");

}

else

{

if (right != null)

return this.right.ListFrom(value);

else

throw new Exception("Искомого узла в дереве нет");

}

}

// отображение в строку

public string Display(SearchTree<T> t)

{

string result = "";

if (t.left != null)

result += Display(t.left);

result += t.value + " ";

if (t.right != null)

result += Display(t.right);

return result;

}

// подсчет

private int Recount(SearchTree<T> t)

{

int count = 0;

if (t.left != null)

count += Recount(t.left);

count++;

if (t.right != null)

count += Recount(t.right);

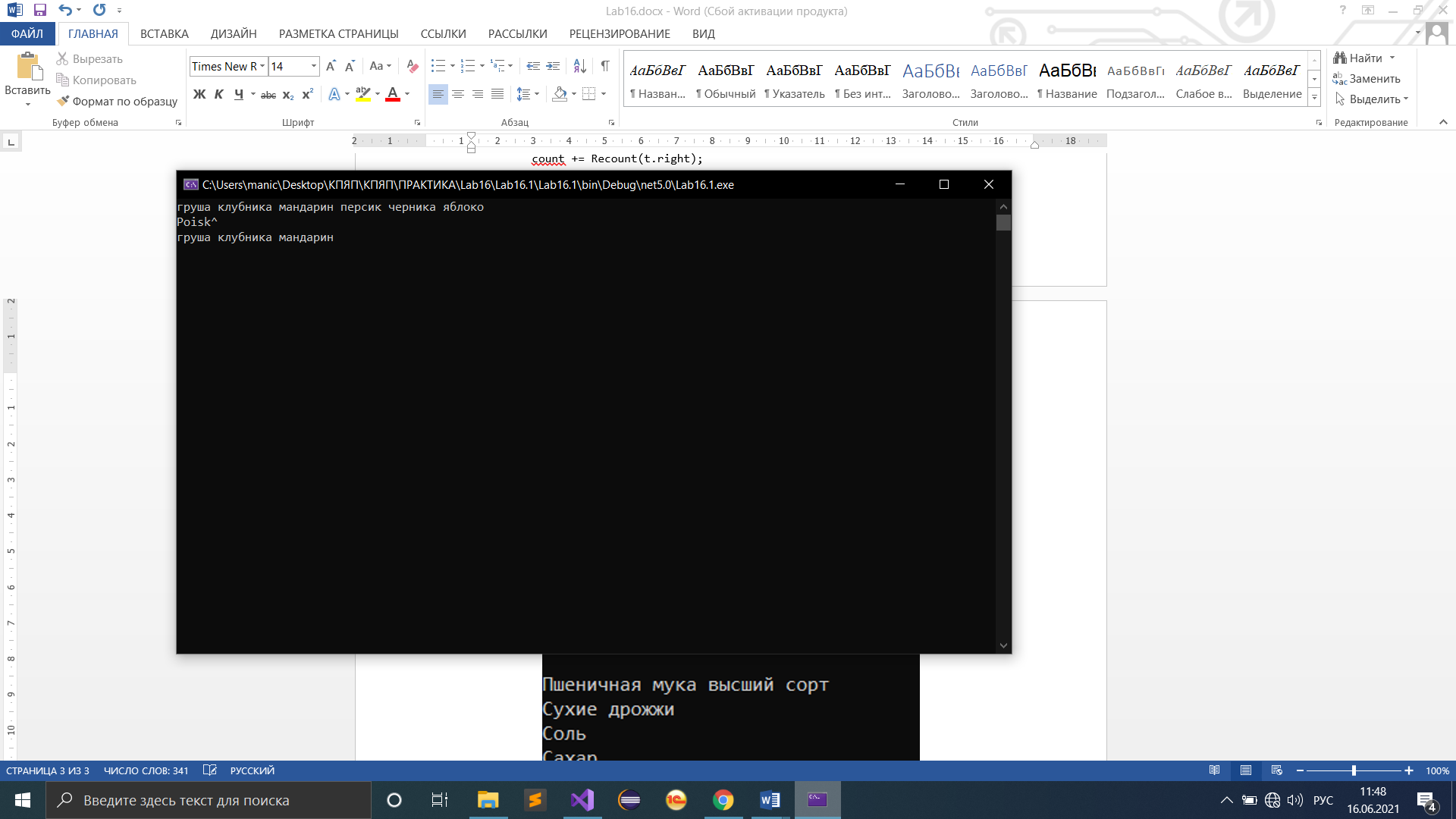
return count;

}

}

}

Результат выполнения программы Рис 1.1.



**Рисунок 1.1 – Пример работы программы c обобщениями**

Отметка\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Подпись преподавателя\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_