

Лабораторная работа №5

Тема: Тип данных – структура

Задание: разработать приложение, позволяющее вносить данные в виде, определенном в задании. Данные представляются в памяти в виде коллекции структур. Должны поддерживаться следующие операции: добавление элемента, удаление элемента, подсчет количества элементов. Также должны поддерживаться функции, определенные в задании. Результат вывести в виде текста.

Структура — это простейший способ логического объединения данных в единую сущность, предоставляемый языком программирования. С точки зрения машинного кода структура не отличается от своих полей-переменных, идущих друг за другом. Структура позволяет объединять разнородные поля в единый элемент данных. На практике, структуры часто используются в составе массивов. Массив структур — это сложная сущность, содержащая набор элементов данных с различной информацией. По сути, массив структур — это простая, но более-менее полная реализация таблицы базы данных. С отдельной структурой можно работать как с единой переменной, при этом произойдет полное копирование значения:

struct Student {

public int Shifr;

public string FIO;

public DateTime DenRozdeniya;

}

…

{

Student aStudent, bStudent;

aStudent.Shifr = 12345;

aStudent.FIO = "Иванов Иван Иванович";

aStudent.DenRozdeniya = new DateTime(1980, 1, 1);

bStudent = aStudent;

}

Однако, непосредственно сравнивать значения структур нельзя — компилятор ничего не знает о смысле данных внутри структур, и не может составить правильную операцию сравнения. Однако, операцию сравнения структур можно доопределить самостоятельно, например, так:

struct Student

{

public int Shifr;

public string FIO;

public DateTime DenRozdeniya;

public static bool operator==(Student a, Student b)

{

return a.Shifr == b.Shifr && a.FIO == b.FIO && a.DenRozdeniya == b.DenRozdeniya;

}

public static bool operator!=(Student a, Student b)

{

return a.Shifr != b.Shifr || a.FIO != b.FIO || a.DenRozdeniya != b.DenRozdeniya;

}

}

В случае работы с набором одинаковых структур данных, можно использовать массив записей:

struct Person

{

public String F;

public String I;

public String O;

}

…

{

Person[] persons = new Person[100500];

}

Однако, более предпочтительным является использование списка:

struct Person

{

public String F;

public String I;

public String O;

}

…

{

List<Person> persons = new List<Person>();

}

Этот список «инкапсулирует» внутри себя динамический массив и все операции с ним, предоставляя нам удобный интерфейс для добавления, удаления и поиска элементов, по сравнению с просто массивом. В большинстве случаев именно применение «общих» коллекций из пространства имен System.Collections.Generic является предпочтительным.

Для ввода данных следует использовать набор полей ввода (это делает возможным внесение всей структуры данных за один прием). Для отображения же целесообразно представить данные в виде таблицы, где строки — единичные записи, столбцы — поля структуры данных, а ячейки — конкретные значения полей. Существует два стандартных компонента, пригодных для отображения таблиц — это DataGridView (таблица строк) и ListView с расширенным отображением элементов и произвольной отрисовкой. Ознакомимся с ним подробнее.

ListView — это надстройка над компонентом операционной системы. В операционной системе данный компонент используется для отображения списка файлов в окнах проводника и для рабочего стола. Он имеет 4 режима отображения — список, маленькие иконки, большие иконки и отчет. Нас интересует последнее — для этого установим свойство View в Details. Колонки в таком представлении редактируются с использованием интерактивного редактора свойства Columns.

Каждый элемент списка в ListView — это структура (объект) типа ListViewItem. У него есть множество свойств:

Index — номер элемента в списке

Tag — произвольный объект (компонент сам никак не обрабатывает это поле)

Selected — логическое свойство, выделен элемент или нет. Для выделения множества элементов следует установить свойство MultiSelect компонента в true (установлено по-умолчанию).

Text — текст элемента. Именно это текст будет показан в ПЕРВОЙ колонке.

SubItems — набор дочерних элементов — ListViewItem.ListViewSubItem, хранящий текст и другие параметры ячеек, начиная со ВТОРОЙ колонки.

Пример использования:

private void button1\_Click(object sender, EventArgs e)

{

Person p;

p.F = textBox1.Text;

p.I = textBox2.Text;

p.O = textBox3.Text;

persons.Add(p);

ListViewItem a = new ListViewItem();

a.Text = p.F;

ListViewItem.ListViewSubItem b = new ListViewItem.ListViewSubItem();

b.Text = p.I;

ListViewItem.ListViewSubItem c = new ListViewItem.ListViewSubItem();

c.Text = p.O;

a.SubItems.Add(b);

a.SubItems.Add(c);

listView1.Items.Add(a);

}

Вариант №1

Данные — сведения о студенте, включают: шифр, фамилию, имя, отчество, группу, факультет, направление подготовки.

1. Найти студентов с введенной фамилией.
2. Найти студентов на введенном факультете
3. Найти студентов в группе
4. Найти студентов по направлению подготовки, обучающихся НЕ на введенном факультете.

Вариант №2

Данные — сведения о сотруднике, включают: личный номер, фамилию, имя, отчество, год рождения, специальность, стаж

1. Найти сотрудников со стажем, больше введенного.
2. Найти сотрудников с определенной специальностью
3. Найти сотрудников с одинаковым стажем и возрастом
4. Найти сотрудников с одинаковой специальностью и стажем

Вариант №3

Данные — сведения о налогоплательщике, включают: ИНН, название (компании или фио), тип (физ. лицо, юр. лицо), форма налогообложения (ООО, ЗАО, ОАО, физ.лицо, ПБОЮЛ, некоммерческая организация), дата подачи последней декларации

1. Найти налогоплательщиков, с даты подачи декларации которых прошло более года.
2. Найти налогоплательщиков с определенной формой налогообложения
3. Найти налогоплательщиков с некорректной формой налогооблажения (ОАО и физ. лицо, юр. лицо и ПБОЮЛ)
4. Найти налогоплательщиков с датой подачи в определенный месяц

Вариант №4

Данные — бюро пропусков, включают: номер пропуска, фамилию, имя, отчество, дата выдачи, срок действия, уровень допуска, дата последней аттестации.

1. Найти все просроченные пропуска
2. Найти все пропуска, у которых с последней аттестации прошло более полугода
3. Найти все пропуска, которые следует перевыпустить (с последней аттестации менее полугода, срок заканчивается в ближайшие 63 дня)
4. Найти все пропуска, выданные на конкретное лицо