**ЛЕКЦИЯ № 07**

**СТРУКТУРЫ. ИНДЕКСНАЯ СОРТИРОВКА**

**СТРУКТУРЫ**

Структура – универсальная форма данных, которая в отличие от массива может содержать элементы, относящиеся к различным типам данных. Это определяемый пользователем тип данных. Сначала создается описание типа структуры, затем создаются экземпляры этого типа.

Структура создается с помощью ключевого слова "struct". В отличие от массива и языка С в структуре в С++ могут находиться не только данные любого типа, но и функции (методы). Все данные и функции в структуре по умолчанию являются общедоступными (public). В конце определения структуры обязательно ставится точка с запятой (;). **В отличие от массива структуры можно присваивать целиком.**

Для обращения к отдельным элементам структуры используется оператор принадлежности – (.) точка (оператор прямого доступа ) или оператор косвенного доступа – (->) стрелка, если используется указатель.

**Пример 4.11** Описание структуры, массив структур, указатель на структуру

#include "stdafx.h"

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* К У Р С Ы С + + \*

\*-------------------------------------------------------------------\*

\* Project Type : Win32 Console Application \*

\* Project Name : Ex04\_Массивы\_Структуры\Ex04\_11\_Структуры.cpp \*

\* File Name : Ex04\_11\_Структуры Индексная сортировка.CPP \*

\* Programmer(s) : Чечиков Ю.Б. & Безродных И.П. \*

\* Modifyed By : \*

\* Created : 06/06/08 \*

\* Last Revision : 18/04/20 \*

\* Comment(s) : СТРУКТУРЫ, МАССИВ СТРУКТУР \*

\* УКЗАТЕЛЬ НА СТРУКТУРУ \*

\* ПЕРЕДАЧА МАССИВА СТРУКТУР В ФУНКЦИЮ \*

\* ЧТЕНИЕ МАССИВА СТРУКТУР ИЗ ФАЙЛА \*

\* ИЗМЕНЕНИЕ МАССИВА СТРУКТУР В ФУНКЦИИ \*

\* ЗАПИСЬ МАССИВА СТРУКТУР В ФАЙЛ \*

\* ВСЕ ФАЙЛЫ ТЕКСТОВЫЕ \*

\* ФРАГМЕНТ ИНДЕКСНОЙ СОТРИРОВКИ \*

\* \*

\*-------------------------------------------------------------------\*

\* РЕЗУЛЬТАТ РАБОТЫ ПРОГРАММЫ \*

\* ИНДЕКСНАЯ СОРТИРОВКА \*

\* Исходный массив AGroup - ТАБЛИЦА \*

\* /PrintStud \*

\* 0 IVANOV 9 2005 \*

\* 1 PETROV 6 1988 \*

\* 2 ANDREEV 12 2001 \*

\* 3 NUKOLAEV 6 2019 \*

\* 4 ZIMIN 15 2018 \*

\* 5 BOBROV 12 2015 \*

\* 6 ABDULIN 3 2017 \*

\* \*

\* . . . Заполнение индексного массива IndexVec \*

\* из исходного массива AGroup . . . \*

\* Исходный индексный массив IndexVec \*

\* 0 IVANOV \*

\* 1 PETROV \*

\* 2 ANDREEV \*

\* 3 NUKOLAEV \*

\* 4 ZIMIN \*

\* 5 BOBROV \*

\* 6 ABDULIN \*

\* \*

\* . . . СОРТИРОВКА ИНДЕКСНОГО МАССИВА IndexVec \*

\* ПО ПОЛЮ key ЛЮБЫМ СПОСОБОМ . . . \*

\* \*

\* Отсортированный IndexVec ИСХОДНЫЙ \*

\* 6 ABDULIN 0 IVANOV......9 2005 \*

\* 2 ANDREEV 1 PETROV......6 1988 \*

\* 5 BOBROV 2 ANDREEV....12 2001 \*

\* 0 IVANOV 3 NUKOLAEV....6 2019 \*

\* 3 NUKOLAEV 4 ZIMIN......15 2018 \*

\* 1 PETROV 5 BOBROV.....12 2015 \*

\* 4 ZIMIN 6 ABDULIN.....3 2017 \*

\* \*

\* Массив AGroup, упорядоченный по полю Name \*

\* 6 ABDULIN..... 3 2017 \*

\* 2 ANDREEV..... 12 2001 \*

\* 5 BOBROV..... 12 2015 \*

\* 0 IVANOV..... 9 2005 \*

\* 3 NUKOLAEV..... 6 2019 \*

\* 1 PETROV..... 6 1988 \*

\* 4 ZIMIN..... 15 2018 \*

\* END 2 \*

\* Для продолжения нажмите любую клавишу . . . \*

\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <iomanip>

#include <fstream> // читаем данные из файла

#include <string>

using namespace std;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Г Л О Б А Л Ь Н Ы Е К О Н С Т А Н Т Ы \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//const char fNAME[] = "SRC1.TXT"; //файл исходных данных

const char fNAME[] = "STUD2.TXT"; //файл исходных данных

const char fNAME\_Out[] = "RES1.txt"; //файл результатов

const int INDFLAG\_ON = true; //печатать индекс

const int INDFLAG\_OFF = false; //НЕ печатать индекс

// Объявим структуру Student

struct Student

{

string Name; //фамилия - удобней при сортировке

//char Name[40]; //фамилия

int Curs; //курс

int God; //год поступления в институт

};

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* П Р О Т О Т И П Ы Ф У Н К Ц И Й \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void InputStud(Student \*pSt, int &QStud); //ввод структуры из файла

void PrintStud(Student \*pSt, int QStud, bool IndFlag); //печать структуры

void FilePrintStud(Student \*pSt, int QStud); //печать структуры в файл

void ModifyStud(Student \*pSt, int QStud); //изменения структуры

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* О С Н О В Н А Я П Р О Г Р А М М А \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int main()

{

setlocale(LC\_ALL, "RUSSIAN");

// Создадим экземпляры структуры "a", "b", "c", "d"

//и инициализируем их

Student a = { "Ivanov" , 3, 1978 };

Student b = { "Petrov" , 1, 1979 };

Student c = { "Sidorov", 5, 1977 };

Student d = { "Lenin" , 3, 1888 };

Student Temp;

int i;

//1 структура и ее поля

// Выводим на экран поля структур,

// используя имя экземпляра структуры

// и оператор прямого доступа - точку

cout << " " << a.Name << " " << a.Curs

<< " " << a.God << endl;

cout << " " << b.Name << " " << b.Curs

<< " " << b.God << endl;

/\*

cout << "ENTER new student\n";

cout <<"\t Name = ";

cin >> Temp.Name;

cout << "\t Curs = ";

cin >> Temp.Curs;

cout << "\t God = ";

cin >> Temp.God;

cout << Temp.Name << " " << Temp.Curs

<< " " << Temp.God << endl;

\*/

//2 массив структур

Student group[5];

group[0] = a;

group[1] = { "STALIN", 5, 1894 };

group[2] = b;

group[3] = c;

group[4] = d;

cout << "\tGROUP LIST\n";

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

{

cout << " " << group[i].Name

<< " " << group[i].Curs

<< " " << group[i].God << endl;

}//for i

//3 работа со структурой через указатель

Student \*pS = &b;

cout << "\t DIRECT\n";

cout << " " << b.Name << " " << b.Curs

<< " " << b.God << endl;

cout << "\t POINTER\n";

cout << " " << pS->Name << " " << pS->Curs

<< " " << pS->God << endl;

//===============================================================

//передача массива (указателя) в функцию и изменение массива

PrintStud(group, 5, INDFLAG\_OFF);

cout << "\n\tПосле функции PrintStud()\n";

//отобразим изменения, сделанные в функции

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

{

cout << "/After PrintStud: " << group[i].Name

<< " " << group[i].Curs << " " << group[i].God << endl;

}//for i

cout << endl;

//===============================================================

//чтение массива структур из файла

//образец файла исходных данных STUD1.TXT

//3

//SELEDKIN-VODKIN 3 2005

//IVANIN 2 1988

//KVAKIN 4 2001

//данные идут с 1-й позиции

const int M\_MAX = 10;

int NRec = 0; //число записей

Student AGroup[M\_MAX], ASt;

//запись, печать и модификация массива структур

InputStud(AGroup, NRec); //ввод из файла

PrintStud(AGroup, NRec, INDFLAG\_OFF); //печать массива

ModifyStud(AGroup, NRec); //изменение массива

FilePrintStud(AGroup, NRec); //печать массива в файл

cout << "END 1\n";

system("PAUSE");

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* ИНДЕКСНАЯ СОРТИРОВКА \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

struct IndexItem //строка индексного массива

{

int ind;

string key; //для символьного ключа

//int key; //для числового ключа

};

IndexItem IndexVec[M\_MAX]; //индексный массив

// int i;

system("cls"); //очистка экрана

cout << "\tИНДЕКСНАЯ СОРТИРОВКА\n";

cout << " Исходный массив AGroup - ТАБЛИЦА\n";

PrintStud(AGroup, NRec, INDFLAG\_ON); //печать массива

cout << " . . . Заполнение индексного массива IndexVec \n";

cout << " из исходного массива AGroup . . .\n";

//ЯКОБЫ заполнение индексного массива IndexVec

// из исходного массива AGroup

IndexVec[0] = { 0, "IVANOV" };

IndexVec[1] = { 1, "PETROV" };

IndexVec[2] = { 2, "ANDREEV" };

IndexVec[3] = { 3, "NUKOLAEV" };

IndexVec[4] = { 4, "ZIMIN" };

IndexVec[5] = { 5, "BOBROV" };

IndexVec[6] = { 6, "ABDULIN" };

cout << "Исходный индексный массив IndexVec\n";

for (i = 0; i < NRec; i++)

{

cout << " " << IndexVec[i].ind << " "

<< IndexVec[i].key << endl;

/\*

0 IVANOV

1 PETROV

2 ANDREEV

3 NUKOLAEV

4 ZIMIN

5 BOBROV

6 ABDULIN

\*/

}//for i

cout << endl;

cout << " . . . СОРТИРОВКА ИНДЕКСНОГО МАССИВА IndexVec\n";

cout << " ПО ПОЛЮ key ЛЮБЫМ СПОСОБОМ . . .\n\n";

/\*

//индексная сортировка

СОРТИРОВКА ИНДЕКСНОГО МАССИВА IndexVec

ПО ПОЛЮ key ЛЮБЫМ СПОСОБОМ

В результате содержимое индексного массива IndexVec будет:

6 ABDULIN

2 ANDREEV

5 BOBROV

0 IVANOV

3 NUKOLAEV

1 PETROV

4 ZIMIN

\*/

//якобы после сортировки индексного массива IndexVec ЛЮБЫМ СПОСОБОМ

IndexVec[0] = { 6, "ABDULIN" };

IndexVec[1] = { 2, "ANDREEV" };

IndexVec[2] = { 5, "BOBROV" };

IndexVec[3] = { 0, "IVANOV" };

IndexVec[4] = { 3, "NUKOLAEV" };

IndexVec[5] = { 1, "PETROV" };

IndexVec[6] = { 4, "ZIMIN" };

//печать результата

cout << "Отсортированный IndexVec \t ИСХОДНЫЙ\n";

for (i = 0; i < NRec; i++)

{

cout << " " << IndexVec[i].ind << " " //индексный массив

<< IndexVec[i].key << "\t\t\t" << i << " "

//для удобства

<< AGroup[i].Name << setw(7) << setfill('.')//исходный массив

<< AGroup[i].Curs << " " << AGroup[i].God

<< endl;

/\*

Отсортированный IndexVec ИСХОДНЫЙ

6 ABDULIN 0 IVANOV 9 2005

2 ANDREEV 1 PETROV 6 1988

5 BOBROV 2 ANDREEV 12 2001

0 IVANOV 3 NUKOLAEV 6 2019

3 NUKOLAEV 4 ZIMIN 15 2018

1 PETROV 5 BOBROV 12 2015

4 ZIMIN 6 ABDULIN 3 2017

\*/

}//for i

cout << endl;

//выводим записи исходного несортированного массива AGroup

//упорядоченными по полю key (в нашем случае Name)

cout << " Массив AGroup, упорядоченный по полю Name\n";

for (i = 0; i < NRec; i++)

{

cout << " " << IndexVec[i].ind //индекс из исходного массива

//строка из исходного массива

<< " " << AGroup[IndexVec[i].ind].Name

<< setw(7) << setfill('.')

<< " " << AGroup[IndexVec[i].ind].Curs

<< "\t" << AGroup[IndexVec[i].ind].God << endl;

/\*

6 ABDULIN..... 3 2017

2 ANDREEV..... 12 2001

5 BOBROV..... 12 2015

0 IVANOV..... 9 2005

3 NUKOLAEV..... 6 2019

1 PETROV..... 6 1988

4 ZIMIN..... 15 2018

\*/

}//for i

cout << "END 2\n";

system("pause");

return 0;// конец блока и main() соответственно

}//end main()

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Р Е А Л И З А Ц И Я Ф У Н К Ц И Й \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//ввод из файла

void InputStud(Student \* AGroup, int &NRec)

{

int i;

cout << "\n\t/InputStud \n";

ifstream fin(fNAME); //открытие входного файла

if (!fin) //не удалось открыть файл

{

cout << "\a\n\tCannot open file " << fNAME << "! \n\n";

system("pause");

}//if

fin >> NRec;

cout << "Nrec = " << NRec << endl;

i = 0; //число реально считанных записей

while (!fin.eof())

{

//fin >> ASt.Name >> ASt.Curs >> ASt.God;

//cout << ASt.Name << " " << ASt.Curs << " " << ASt.God << endl;

fin >> AGroup[i].Name >> AGroup[i].Curs

>> AGroup[i].God;

cout << AGroup[i].Name << " " << AGroup[i].Curs << " "

<< AGroup[i].God << endl;

i++;

}//while

fin.close(); //закрыть файл

}//end InputStud()

//----------------------------------------------------------

//изменение массива

void ModifyStud(Student group[], int QStud)

{

int i;

cout << "\n\t/ModifyStud \n";

//изменим и напечатаем

for (i = 0; i < QStud; i++) //цикл по количеству студентов

{

group[i].Curs \*= 3; //изменение

cout << " " << group[i].Name

<< " " << group[i].Curs

<< " " << group[i].God << endl;

}//for i

}//end ModifyStud()

//----------------------------------------------------------

//печать массива

void PrintStud(Student \*pSt, int QStud, bool IndFlag)

{

int i;

cout << "\t/PrintStud \n";

//напечатаем и изменим

for (i = 0; i < QStud; i++)

{

cout << " ";

if (IndFlag) //печатать индекс

cout << i << " ";

cout << pSt->Name

<< " " << pSt->Curs << " " << pSt->God << endl;

// pSt->Curs \*= 2; //изменение

pSt++; //следующий индекс

}//for i

}//end PrintStud()

//----------------------------------------------------------

//печать структуры в файл

void FilePrintStud(Student group[], int QStud)

{

int i;

cout << "\n\t/FilePrintStud \n";

ofstream fout(fNAME\_Out); //открытие выходного файла

if (!fout)

{

cout << "\a\n\tCannot open file " << fNAME\_Out << "! \n\n";

system("pause");

}

cout << "Nrec = " << QStud << endl;

fout << "Nrec = " << QStud << endl;

for (i = 0; i < QStud; i++) //цикл по количеству студентов

{

cout << " " << group[i].Name //на экран

<< " " << group[i].Curs

<< " " << group[i].God << endl;

fout << " " << group[i].Name //в файл

<< " " << group[i].Curs

<< " " << group[i].God << endl;

}//for i

fout.close();

}//end FilePrintStud()

/\*\*\* End Of Ex04\_11\_Структуры Индексная сортировка.cpp File \*\*\*/

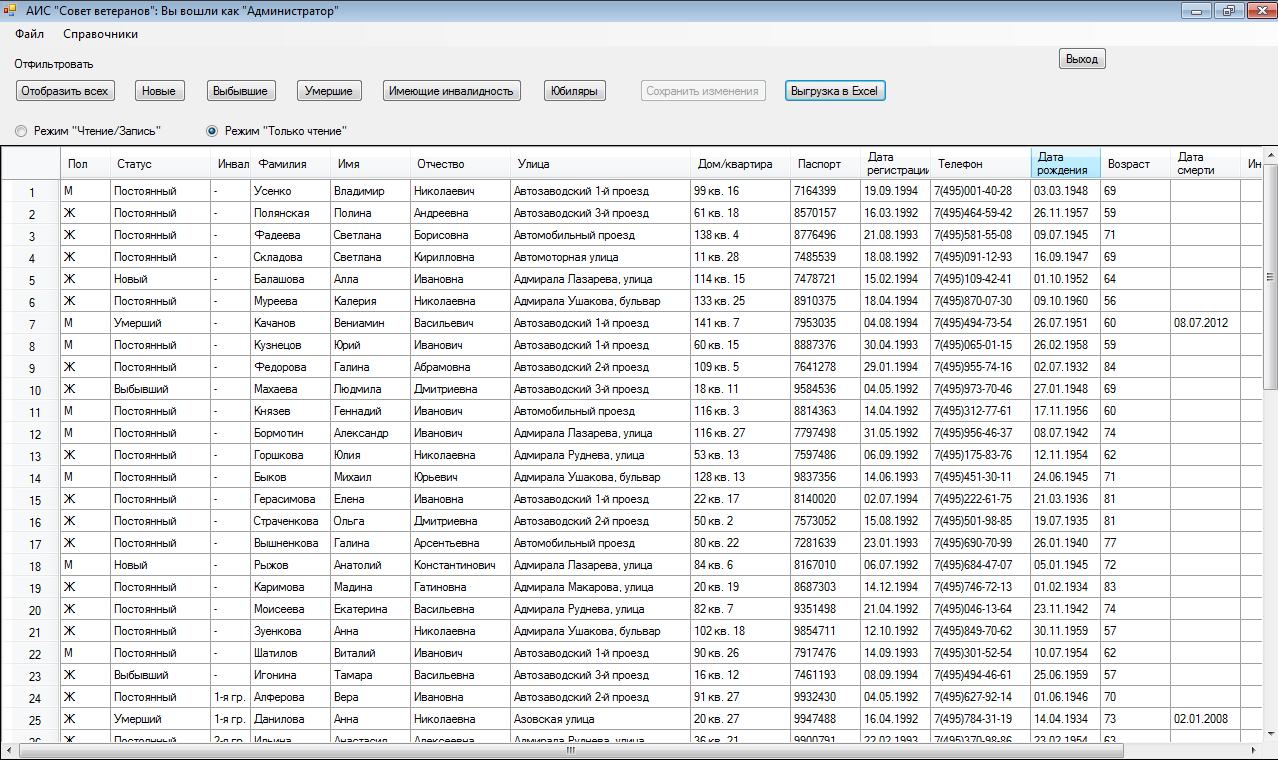
**<К О Н Е Ц Л Е К Ц И И № >**

**Индексная сортировка**

Сортировка по индексному полю предполагает сортировку исходных данных с использованием вспомогательного массива. Элементом массива является запись, содержащая ключевое поле( информация, которая подлежит сортировке) и номер соответствующей строки массива (индекс) откуда взята эта информация. Изначально индексный массив получается упорядоченным по номерам строк, а по ключевым полям - неизвестно. Затем сортируется вспомогательный массив по ключевому полю и переставляются записи целиком вместе с номерами строк. В результате индексный массив становится упорядоченным по ключевому полю, а по номерам строк – как получится. **Исходный массив записей при этом остается без изменения**.

Для просмотра данных исходного массива в отсортированном порядке используется упорядоченный индексный массив. Идет цикл по индексному массиву, берется значение очередного индекса и печатается элемент основного массива с индексом, взятым из этого индексного массива. Таким образом, осуществляется вывод исходного массива в порядке, заданном номерами строк в отсортированном индексном массиве. Собственно сортировка индексного массива выполняется любым способом

Индексная сортировка находит широкое применение в базах данных, содержащих таблицы с большим числом столбцов и огромным количеством строк. Сортировать такую таблицу по разным столбцам неэффективно, так как после каждой сортировки надо хранить копию массива, упорядоченного по одному полю. Получается число копий таблицы равно числу сортировок по разным столбцам (ключевым полям). После изменения таблицы все сортировки приходится проводить заново. Недостатки очевидны – большие затраты дисковой памяти и времени на сортировки. Индексные массивы занимают мало места, во время работы хранятся в оперативной памяти и после изменения основного массива легко переупорядочиваются.



/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* К У Р С Ы С + + \*

\*-------------------------------------------------------------------\*

\* Project Type : Win32 Console Application \*

\* Project Name : Ex04\_Массивы\_Структуры\Ex04\_11\_Структуры.cpp \*

\* File Name : Ex04\_09.CPP \*

\* Programmer(s) : Чечиков Ю.Б. & Безродных И.П. \*

\* Modifyed By : \*

\* Created : 06/06/08 \*

\* Last Revision : 21/04/20 \*

\* Comment(s) : СТРУКТУРЫ, МАССИВ СТРУКТУР \*

\* УКЗАТЕЛЬ НА СТРУКТУРУ \*

\* ПЕРЕДАЧА МАССИВА СТРУКТУР В ФУНКЦИЮ \*

\* ЧТЕНИЕ МАССИВА СТРУКТУР ИЗ ФАЙЛА \*

\* ИЗМЕНЕНИЕ МАССИВА СТРУКТУР В ФУНКЦИИ \*

\* ЗАПИСЬ МАССИВА СТРУКТУР В ФАЙЛ \*

\* ВСЕ ФАЙЛЫ ТЕКСТОВЫЕ \*

\* ФРАГМЕНТ ИНДЕКСНОЙ СОТРИРОВКИ \*

\* \*

\*-------------------------------------------------------------------\*

\* РЕЗУЛЬТАТ РАБОТЫ ПРОГРАММЫ \*

\* ИНДЕКСНАЯ СОРТИРОВКА \*

\* Исходный массив AGroup - ТАБЛИЦА \*

\* /PrintStud \*

\* 0 IVANOV 9 2005 \*

\* 1 PETROV 6 1988 \*

\* 2 ANDREEV 12 2001 \*

\* 3 NUKOLAEV 6 2019 \*

\* 4 ZIMIN 15 2018 \*

\* 5 BOBROV 12 2015 \*

\* 6 ABDULIN 3 2017 \*

\* \*

\* . . . Заполнение индексного массива IndexVec \*

\* из исходного массива AGroup . . . \*

\* Исходный индексный массив IndexVec \*

\* 0 IVANOV \*

\* 1 PETROV \*

\* 2 ANDREEV \*

\* 3 NUKOLAEV \*

\* 4 ZIMIN \*

\* 5 BOBROV \*

\* 6 ABDULIN \*

\* \*

\* . . . СОРТИРОВКА ИНДЕКСНОГО МАССИВА IndexVec \*

\* ПО ПОЛЮ key ЛЮБЫМ СПОСОБОМ . . . \*

\* \*

\* Отсортированный IndexVec ИСХОДНЫЙ \*

\* 6 ABDULIN 0 IVANOV......9 2005 \*

\* 2 ANDREEV 1 PETROV......6 1988 \*

\* 5 BOBROV 2 ANDREEV....12 2001 \*

\* 0 IVANOV 3 NUKOLAEV....6 2019 \*

\* 3 NUKOLAEV 4 ZIMIN......15 2018 \*

\* 1 PETROV 5 BOBROV.....12 2015 \*

\* 4 ZIMIN 6 ABDULIN.....3 2017 \*

\* \*

\* Массив AGroup, упорядоченный по полю Name \*

\* 6 ABDULIN..... 3 2017 \*

\* 2 ANDREEV..... 12 2001 \*

\* 5 BOBROV..... 12 2015 \*

\* 0 IVANOV..... 9 2005 \*

\* 3 NUKOLAEV..... 6 2019 \*

\* 1 PETROV..... 6 1988 \*

\* 4 ZIMIN..... 15 2018 \*

\* END 2 \*

\* Для продолжения нажмите любую клавишу . . . \*

\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <iomanip>

#include <fstream> // читаем данные из файла

#include <string>

using namespace std;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Г Л О Б А Л Ь Н Ы Е К О Н С Т А Н Т Ы \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//const char fNAME[] = "SRC1.TXT"; //файл исходных данных

const char fNAME[] = "STUD1.TXT"; //файл исходных данных

const char fNAME\_Out[] = "RES1.txt"; //файл результатов

const int INDFLAG\_ON = true; //печатать индекс

const int INDFLAG\_OFF = false; //НЕ печатать индекс

// Объявим структуру Student

struct Student

{

string Name; //фамилия - удобней при сортировке

//char Name[40]; //фамилия

int Curs; //курс

int God; //год поступления в институт

};

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* П Р О Т О Т И П Ы Ф У Н К Ц И Й \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void InputStud(Student \*pSt, int & QStud); //ввод структуры из файла

void PrintStud(Student \*pSt, int QStud, bool IndFlag); //печать структуры

void FilePrintStud(Student \*pSt, int QStud); //печать структуры в файл

void ModifyStud(Student \*pSt, int QStud); //изменения структуры

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* О С Н О В Н А Я П Р О Г Р А М М А \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int main()

{

setlocale(LC\_ALL, "RUSSIAN");

// Создадим экземпляры структуры "a", "b", "c", "d"

//и инициализируем их

Student a = { "Ivanov", 3, 1978 };

Student b = { "Petrov", 1, 1979 };

Student c = { "Sidorov", 5, 1977 };

Student d = { "Lenin", 3, 1888 };

Student Temp;

int i;

//1 структура и ее поля

// Выводим на экран поля структур,

// используя имя экземпляра структуры

// и оператор прямого доступа - точку

cout << " " << a.Name << " " << a.Curs

<< " " << a.God << endl;

cout << " " << b.Name << " " << b.Curs

<< " " << b.God << endl;

/\*

cout << "ENTER new student\n";

cout <<"\t Name = ";

cin >> Temp.Name;

cout << "\t Curs = ";

cin >> Temp.Curs;

cout << "\t God = ";

cin >> Temp.God;

cout << Temp.Name << " " << Temp.Curs

<< " " << Temp.God << endl;

\*/

//2 массив структур

Student group[5];

group[0] = a;

group[1] = { "STALIN", 5, 1894 };

group[2] = b;

group[3] = c;

group[4] = d;

cout << "\tGROUP LIST\n";

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

{

cout << " " << group[i].Name

<< " " << group[i].Curs

<< " " << group[i].God << endl;

}//for i

//3 работа со структурой через указатель

Student \* pS = &b;

cout << "\n\t DIRECT\n";

cout << " " << b.Name << " " << b.Curs

<< " " << b.God << endl;

cout << "\t POINTER\n";

cout << " " << pS->Name << " " << pS->Curs

<< " " << pS->God << endl;

//===============================================================

//передача массива (указателя) в функцию и изменение массива

PrintStud(group, 5, INDFLAG\_OFF);

//cout << "\n\tПосле функции PrintStud()\n";

////отобразим изменения, сделанные в функции

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

//{

// cout << "/After PrintStud: " << group[i].Name

// << " " << group[i].Curs << " " << group[i].God << endl;

//}//for i

cout << endl;

//===============================================================

//чтение массива структур из файла

//образец файла исходных данных STUD1.TXT

//3

//SELEDKIN-VODKIN 3 2005

//IVANIN 2 1988

//KVAKIN 4 2001

//данные идут с 0-й позиции

const int M\_MAX = 10;

int NRec = 0; //число записей

Student AGroup[M\_MAX], ASt;

//запись, печать и модификация массива структур

InputStud(AGroup, NRec); //ввод из файла

PrintStud(AGroup, NRec, INDFLAG\_OFF); //печать массива

ModifyStud(AGroup, NRec); //изменение массива

FilePrintStud(AGroup, NRec); //печать массива в файл

cout << "END 1\n";

system("PAUSE");

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* ИНДЕКСНАЯ СОРТИРОВКА \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

struct IndexItem //строка индексного массива

{

int ind;

string key; //для символьного ключа

//int key; //для числового ключа

};

IndexItem IndexVec[M\_MAX]; //индексный массив

// int i;

//исходный массив, якобы считанный из файла - НЕ МЕНЯЕТСЯ

AGroup[0] = { " IVANOV", 9, 2005 };

AGroup[1] = { " PETROV", 6, 1988 };

AGroup[2] = { " ANDREEV", 12, 2001 };

AGroup[3] = { " NUKOLAEV", 6, 2019 };

AGroup[4] = { " ZIMIN", 15, 2018 };

AGroup[5] = { " BOBROV", 12, 2015 };

AGroup[6] = { " ABDULIN", 3, 2017 };

NRec = 7;

system("cls"); //очистка экрана

cout << "\tИНДЕКСНАЯ СОРТИРОВКА\n";

cout << " Исходный массив AGroup - ТАБЛИЦА\n";

PrintStud(AGroup, NRec, INDFLAG\_ON); //печать массива

cout << " . . . Заполнение индексного массива IndexVec \n";

cout << " из исходного массива AGroup . . .\n";

//ЯКОБЫ заполнение индексного массива IndexVec

// из исходного массива AGroup

//IndexVec[i].ind = i;

//IndexVec[i].key = AGroup[i].Name;

IndexVec[0] = { 0, "IVANOV" };

IndexVec[1] = { 1, "PETROV" };

IndexVec[2] = { 2, "ANDREEV" };

IndexVec[3] = { 3, "NUKOLAEV" };

IndexVec[4] = { 4, "ZIMIN" };

IndexVec[5] = { 5, "BOBROV" };

IndexVec[6] = { 6, "ABDULIN" };

cout << "Исходный индексный массив IndexVec\n";

for (i = 0; i < NRec; i++)

{

cout << " " << IndexVec[i].ind << " "

<< IndexVec[i].key << endl;

/\*

0 IVANOV

1 PETROV

2 ANDREEV

3 NUKOLAEV

4 ZIMIN

5 BOBROV

6 ABDULIN

\*/

}//for i

cout << endl;

cout << " . . . СОРТИРОВКА ИНДЕКСНОГО МАССИВА IndexVec\n";

cout << " ПО ПОЛЮ key ЛЮБЫМ СПОСОБОМ . . .\n\n";

/\*

//индексная сортировка

СОРТИРОВКА ИНДЕКСНОГО МАССИВА IndexVec

ПО ПОЛЮ key ЛЮБЫМ СПОСОБОМ

В результате содержимое индексного массива IndexVec будет:

6 ABDULIN

2 ANDREEV

5 BOBROV

0 IVANOV

3 NUKOLAEV

1 PETROV

4 ZIMIN

\*/

//якобы после сортировки индексного массива IndexVec ЛЮБЫМ СПОСОБОМ

IndexVec[0] = { 6, "ABDULIN" };

IndexVec[1] = { 2, "ANDREEV" };

IndexVec[2] = { 5, "BOBROV" };

IndexVec[3] = { 0, "IVANOV" };

IndexVec[4] = { 3, "NUKOLAEV" };

IndexVec[5] = { 1, "PETROV" };

IndexVec[6] = { 4, "ZIMIN" };

//печать результата

cout << "Отсортированный IndexVec \t ИСХОДНЫЙ\n";

for (i = 0; i < NRec; i++)

{

cout << " " << IndexVec[i].ind << " " //индексный массив

<< IndexVec[i].key << "\t\t\t" << i << " "

//для удобства

<< AGroup[i].Name << setw(7) << setfill('.')//исходный массив

<< AGroup[i].Curs << " " << AGroup[i].God

<< endl;

/\*

Отсортированный IndexVec ИСХОДНЫЙ

6 ABDULIN 0 IVANOV 9 2005

2 ANDREEV 1 PETROV 6 1988

5 BOBROV 2 ANDREEV 12 2001

0 IVANOV 3 NUKOLAEV 6 2019

3 NUKOLAEV 4 ZIMIN 15 2018

1 PETROV 5 BOBROV 12 2015

4 ZIMIN 6 ABDULIN 3 2017

\*/

}//for i

cout << endl;

//выводим записи исходного несортированного массива AGroup

//упорядоченными по полю key (в нашем случае Name)

cout << " Массив AGroup, упорядоченный по полю Name\n";

for (i = 0; i < NRec; i++)

{

cout << " " << IndexVec[i].ind //индекс из исходного массива

//строка из исходного массива

<< " " << AGroup[IndexVec[i].ind].Name

<< setw(7) << setfill('.')

<< " " << AGroup[IndexVec[i].ind].Curs

<< "\t" << AGroup[IndexVec[i].ind].God << endl;

/\*

6 ABDULIN..... 3 2017

2 ANDREEV..... 12 2001

5 BOBROV..... 12 2015

0 IVANOV..... 9 2005

3 NUKOLAEV..... 6 2019

1 PETROV..... 6 1988

4 ZIMIN..... 15 2018

\*/

}//for i

cout << "END 2\n";

system("pause");

return 0;// конец блока и main() соответственно

}//end main()

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Р Е А Л И З А Ц И Я Ф У Н К Ц И Й \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//ввод из файла

void InputStud(Student \* AGroup, int &NRec)

{

int i;

cout << "\n\t/InputStud \n";

ifstream fin(fNAME); //открытие входного файла

if (!fin) //не удалось открыть файл

{

cout << "\a\n\tCannot open file " << fNAME << "! \n\n";

system("pause");

}//if

fin >> NRec;

cout << "Nrec = " << NRec << endl;

i = 0; //число реально считанных записей

while (!fin.eof())

{

//fin >> ASt.Name >> ASt.Curs >> ASt.God;

//cout << ASt.Name << " " << ASt.Curs << " " << ASt.God << endl;

fin >> AGroup[i].Name >> AGroup[i].Curs

>> AGroup[i].God;

cout << AGroup[i].Name << " " << AGroup[i].Curs << " "

<< AGroup[i].God << endl;

i++;

}//while

fin.close(); //закрыть файл

}//end InputStud()

//----------------------------------------------------------

//изменение массива

void ModifyStud(Student group[], int QStud)

{

int i;

cout << "\n\t/ModifyStud \n";

//изменим и напечатаем

for (i = 0; i < QStud; i++) //цикл по количеству студентов

{

group[i].Curs \*= 3; //изменение

cout << " " << group[i].Name

<< " " << group[i].Curs

<< " " << group[i].God << endl;

}//for i

}//end ModifyStud()

//----------------------------------------------------------

//печать массива

void PrintStud(Student \*pSt, int QStud, bool IndFlag)

{

int i;

cout << "\n\t/PrintStud \n";

//напечатаем и изменим

for (i = 0; i < QStud; i++)

{

cout << " ";

if (IndFlag) //печатать индекс

cout << i << " ";

cout << pSt->Name

<< " " << pSt->Curs << " " << pSt->God << "\n";

//// pSt->Curs \*= 2; //изменение

//cout << pSt[i].Name

// << " " << pSt[i].Curs << " " << pSt->God << endl;

pSt++; //следующий индекс

}//for i

}//end PrintStud()

//----------------------------------------------------------

//печать структуры в файл

void FilePrintStud(Student group[], int QStud)

{

int i;

cout << "\n\t/FilePrintStud \n";

ofstream fout(fNAME\_Out); //открытие выходного файла

if (!fout)

{

cout << "\a\n\tCannot open file " << fNAME\_Out << "! \n\n";

system("pause");

}

cout << "Nrec = " << QStud << endl;

fout << "Nrec = " << QStud << endl;

for (i = 0; i < QStud; i++) //цикл по количеству студентов

{

cout << " " << group[i].Name //на экран

<< " " << group[i].Curs

<< " " << group[i].God << endl;

fout << " " << group[i].Name //в файл

<< " " << group[i].Curs

<< " " << group[i].God << endl;

}//for i

fout.close();

}//end FilePrintStud()

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* End Of Ex04\_11\_Структуры.cpp File \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**КОНЕЦ**