Министерство науки и высшего образования Российской Федерации

Пензенской государственный университет

Кафедра "Вычислительная техника"

**Отчёт**

по лабораторной работе №1

по курсу «Логика и Основы Алгоритмизации в Инженерных Задачах»

на тему «Простые структуры данных»

Выполнил:

Студент группы 23ВВ1

Юров Д.М.

Принял:

Митрохин М.А.

Юрова О.В.

Пенза 2024

**Цель работы:** изучить простые структуры данных в языке Си.

**Лабораторное задание:**

***Задание 1:*** написать программу, вычисляющую разницу между максимальным и минимальным элементами массива.

***Задание 2****:* написать программу, реализующую инициализацию массива случайными числами.

***Задание 3****:* написать программу, реализующую создание массива произвольного размера, вводимого с клавиатуры.

***Задание 4****:* написать программу, вычисляющую сумму значений в каждом столбце (или строке) двумерного массива.

***Задание 5:*** написать программу, осуществляющую поиск среди структур student структуру с заданными параметрами (фамилией, именем и т.д.).

**Теоретическая часть:**

К простым структурам данных языка Си относятся массивы, строки (массивы символов). ***Структуры*** (struct) – простейшая составная структура данных.

***Массив*** – это конечная совокупность данных одного типа.

Массивы могут состоять из целых чисел, чисел с плавающей запятой, символов и других типов данных. Существуют даже массивы массивов, – многомерные массивы.

***Динамический массив.*** Для использования функций динамического распределения памяти необходимо подключение библиотеки <malloc.h> или <stdlib.h> (в зависимости от компилятора и используемого стандарта языка).

*Функции динамического выделения памяти*

void\* malloc(*РазмерМассиваВБайтах*);  
 void\* calloc(*ЧислоЭлементов*, *РазмерЭлементаВБайтах*);

выделяют блок памяти, размером *РазмерМассиваВБайтах* или *ЧислоЭлементов*\**РазмерЭлементаВБайтах* байт, и возвращает указатель на начало блока.

Элементами массива могут быть как данные простых типов, так и составных типов (например, структуры), поэтому для точного определения размера элемента массива в общем случае используется функция

int sizeof(*ТипЭлемента*);  
Эта функция возвращает количество байт, занимаемое элементом указанного типа.

Память, динамически выделенная с использованием функций calloc(), malloc(), должна быть освобождена после окончания использования функцией

 free(*указатель*);

Для ***генерации случайных чисел*** в языке Си используется функция rand(), которая содержится в библиотеке <stdlib.h>.

При повторных запусках программы функция rand() будет возвращать одинаковые последовательности чисел. Для получения уникальных последовательностей необходимо инициализировать генератор случайных чисел различными значениями. Для этого используется функция srand(), аргументом которой является инициализирующее целое число.

***Структура*** - это совокупность переменных одного или нескольких типов, сгруппированных в один элемент.

**Практическая часть:**

**Листинг:**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdlib.h>

#include <stdio.h>

#include <iostream>

#include <time.h>

#include <iomanip>

using namespace std;

void main()

{

srand(time(NULL));

//\* task 2: initialization randomized array \*

cout << "\* task 2: initialization randomized array \*" << endl;

int A[20] = {};

cout << endl << "your array:" << endl << "[ ";

for (int i = 0; i < 20; i++)

{

A[i] = rand() % 202;

cout << A[i] << " ";

}

cout << "]" << endl << endl << endl;

//\* task 3: initialization randomized dynamic array \*

cout << "\* task 3: initialization randomized dynamic array \*" << endl << endl;

int z = 0;

cout << "input array size" << endl;

cin >> z;

int\* m = (int\*)malloc(sizeof(int) \* z);

if (z == 0)

{

cout << endl << "array size cannot be zero" << endl;

return;

}

else

{

cout << endl << "your array:" << endl << "[ ";

for (int i = 0; i < z; i++, m++)

{

\*m = rand() % 202;

cout << \*m << " ";

}

m--;

cout << "]" << endl << endl << endl;

}

//\* task 1: difference between max and min element \*

cout << "\* task 1: difference between max and min element \*" << endl << endl;

int max\_m = 0, min\_m = 1000;

int max\_p = 0, min\_p = 0;

for (; z != 0; z--, m--)

{

if (\*m >= max\_m)

{

max\_m = \*m;

max\_p = z;

}

if (\*m <= min\_m)

{

min\_m = \*m;

min\_p = z;

}

}

cout << "greatest element is " << max\_p << " and it = " << max\_m << endl;

cout << "smallest element is " << min\_p << " and it = " << min\_m << endl;

cout << "difference = " << max\_m - min\_m << endl;

cout << endl << endl;

//\* task 4: row and column sums \*

cout << "\* task 4: row and column sums \*" << endl << endl;

int N[5][5];

int row\_sum = 0, column\_sum = 0;

cout << "generated matrix: " << endl;

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

{

if (i == 0) cout << "/";

if (i != 0 && i != 4) cout << "|";

if (i == 4) cout << "\\";

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

{

N[i][j] = rand() % 202;

cout << setw(7) << N[i][j] << " ";

row\_sum += N[i][j];

}

if (i == 0) cout << "\\";

if (i != 0 && i != 4) cout << "|";

if (i == 4) cout << "/";

cout << setw(7) << row\_sum << endl;

row\_sum = 0;

}

cout << endl;

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

{

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

{

column\_sum += N[i][j];

}

cout << setw(8) << column\_sum;

column\_sum = 0;

}

cout << setw(9) << "sum" << endl << endl;

//\* task 5: search in struct \*

cout << "\* task 5: search in struct \*" << endl << endl;

string l\_names[] = { "Makarov", "Vidyaev", "Usov", "Kazakov", "Kuzmin", "Jiganov", "Yurov", "Pilnov" };

string names[] = { "Daniil", "Alexander", "Alexey", "Vladimir", "Denis", "Anton", "Maksim", "Nikita" };

string faculty[] = { "FVT", "FITE", "PTET", "Ist-Fil", "PED", "YUR-Fak", "Econom", "MED" };

string grade\_book\_num[] = { "23VVV1312", "21VI32", "24IPO231", "19LL13234", "22NS13", "23ZMOS412", "23ZFPB11", "17KE242" };

struct student

{

string famil, name, facult, Nomzach;

} stud[10], tmp\_stud[10];

int x = 0;

int temp\_type = 0;

string temp\_value;

int c = 0;

cout << "All students:" << endl << setw(20) << "last name" << setw(20) << "name" << setw(20) << "faculty" << setw(20) << "gr book num" << endl << endl;

for (int i = 0; i < 10; i++)

{

x = rand() % 8;

stud[i].famil = l\_names[x];

x = rand() % 8;

stud[i].name = names[x];

x = rand() % 8;

stud[i].facult = faculty[x];

x = rand() % 8;

stud[i].Nomzach = grade\_book\_num[x];

cout << setw(20) << stud[i].famil << setw(20) << stud[i].name << setw(20) << stud[i].facult << setw(20) << stud[i].Nomzach << endl;

}

cout << endl << "enter search parameter" << endl << "(last name = 1 | name = 2 | faculty = 3 | grade book number = 4):" << endl;

cin >> temp\_type;

cout << endl << "enter search value:" << endl;

cin >> temp\_value;

switch (temp\_type)

{

case 1:

{

for (int i = 0; i < 10; i++)

{

if (stud[i].famil == temp\_value)

{

tmp\_stud[c] = stud[i];

c++;

}

}

break;

}

case 2:

{

for (int i = 0; i < 10; i++)

{

if (stud[i].name == temp\_value)

{

tmp\_stud[c] = stud[i];

c++;

}

}

break;

}

case 3:

{

for (int i = 0; i < 10; i++)

{

if (stud[i].facult == temp\_value)

{

tmp\_stud[c] = stud[i];

c++;

}

}

break;

}

case 4:

{

for (int i = 0; i < 10; i++)

{

if (stud[i].Nomzach == temp\_value)

{

tmp\_stud[c] = stud[i];

c++;

}

}

break;

}

default:

{

cout << "error. must type 1 to 4." << endl << endl;

return;

}

}

if (c <= 0)

{

cout << endl << "error. check entered value, it's wrong" << endl << endl;

}

else

{

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

{

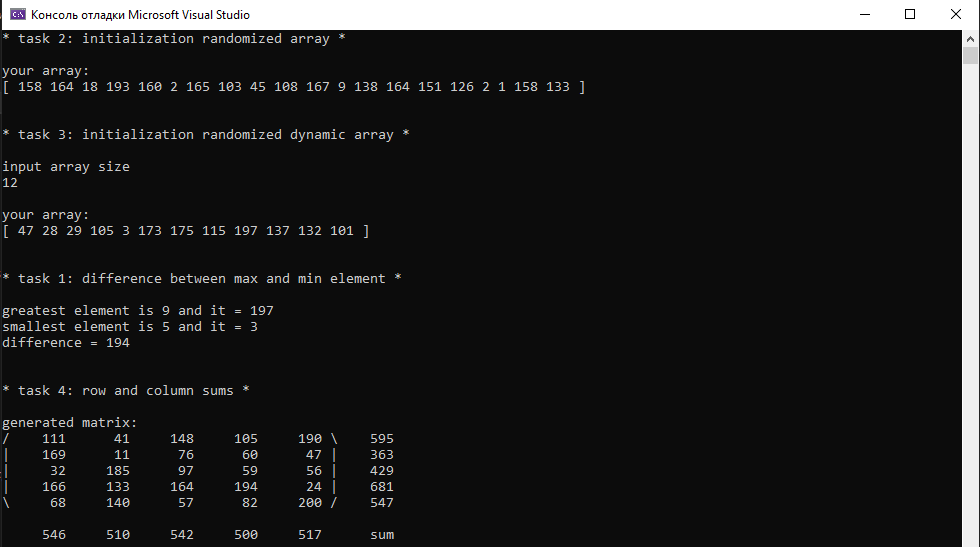
cout << endl << "Information about student:" << endl << "last name: " << tmp\_stud[i].famil << endl << "name: " << tmp\_stud[i].name << endl << "faculty: " << tmp\_stud[i].facult << endl << "grade book number: " << tmp\_stud[i].Nomzach << endl << endl;

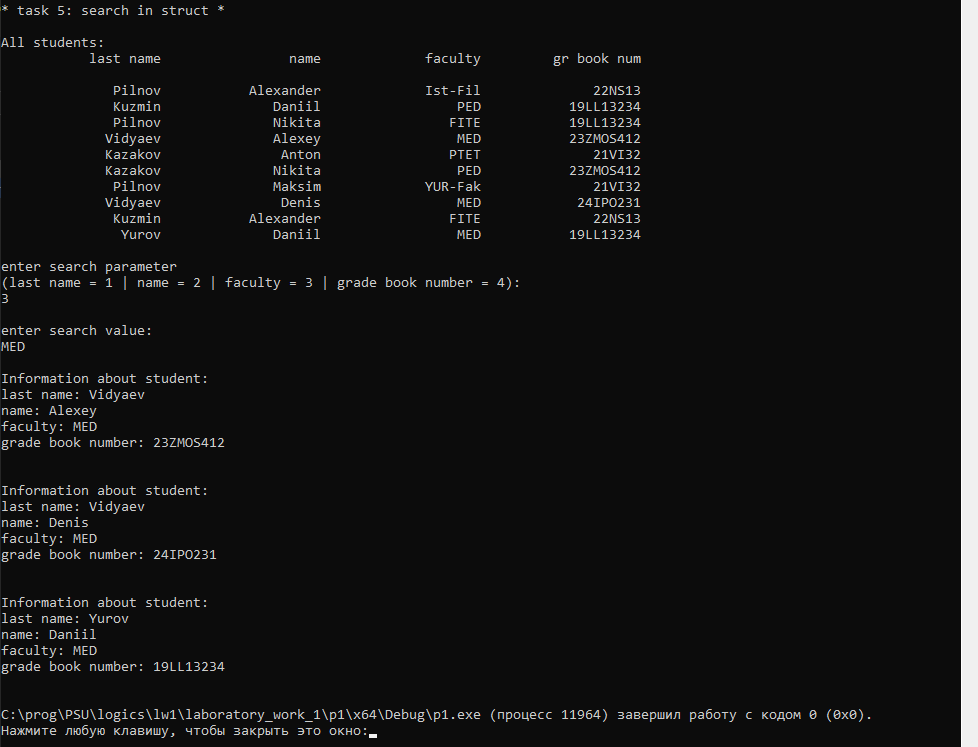
}

}

}

**Результат работы программы:**

****

****

**Вывод:** изучили простые структуры данных на практике.