Пензенский государственный университет

Кафедра "Вычислительной техники"

**Отчёт**

По лабораторной №1

по дисциплине "Л и ОА в ИТ"

на тему "Простые структуры данных"

***Выполнили студенты группы 20вв4:***

*Волков А.Ю.*

*Исаев С.Д.*

***Приняли:***

*Акифьев И.В.*

*Юрова О. В.*

Пенза 2021

# 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 <iostream>

#include <stdio.h>

#include <time.h>

#include <locale.h>

#include <conio.h>

#include <string.h>

int main()

{

setlocale(LC\_ALL, "Rus");

int m[10];

int max = 0;

int min = 200;

int r, i;

srand(time(NULL));

printf("num 1\n");

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

if (m[i] < min)

min = m[i];

}

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

if (m[i] > max)

max = m[i];

}

r = max - min;

printf("\n%d\n", r);

printf("num 2\n");

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

m[i] = rand() % 20 + 1;

printf("%3d", m[i]);

}

printf("\n");

printf("num 3\n");

int\* z;

int n;

printf("obiem\n");

scanf("%d", &n);

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

for (i = 0; i < n; i++) {

z[i] = rand() % 21 - 10;

printf("%3d", z[i]);

}

printf("\n");

free(z);

printf("num 3.1\n");

int\*\* M;

int n, h, j;

printf("obiem\n");

scanf("%d", &n);

scanf("%d", &h);

M = (int\*\*) malloc (n \* sizeof(int\*));

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

M[i] = (int\*)malloc(h \* sizeof(int));

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

{

for (j = 0; j < h; j++)

{

M[i][j] = rand() % 21 - 10;

printf("%3d", \*(M + i \* h + j));

}

}

printf("\n");

free(M);

printf("num 4\n");

int a[5][5];

int sc[5];

int s, j;

for (i = 0; i < 5; i++) sc[i] = 0;

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

s = 0;

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

a[i][j] = rand() % 10;

printf("%5d", a[i][j]);

s += a[i][j];

sc[j] += a[i][j];

}

printf(" |%d\n", s);

}

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

printf("%5s", "--");

}

printf("\n");

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

printf("%5d", sc[i]);

}

printf("\n");

printf("num 5\n");

struct student {

char famil[20], name[20], number[20];

};

char poisk[20];

int flag = 0;

setlocale(LC\_ALL, "Russian");

student arr[3];

printf("familiya imya gruppa\n");

for (int i = 0; i < 2; i++) {

scanf("%s %s %s", arr[i].famil, arr[i].name, arr[i].number);

}

for (int i = 0; i < 2; i++) {

printf(" %s %s %s \n", arr[i].famil, arr[i].name, arr[i].number);

}

printf("familiya\n");

scanf("%s", poisk);

for (int i = 0; i < 2; i++) {

if (strcmp(poisk, arr[i].famil) == 0) {

flag = 1;

printf(" %s %s %s \n", arr[i].famil, arr[i].name, arr[i].number);

}

}

if (flag == 0) {

printf("nenaxod");

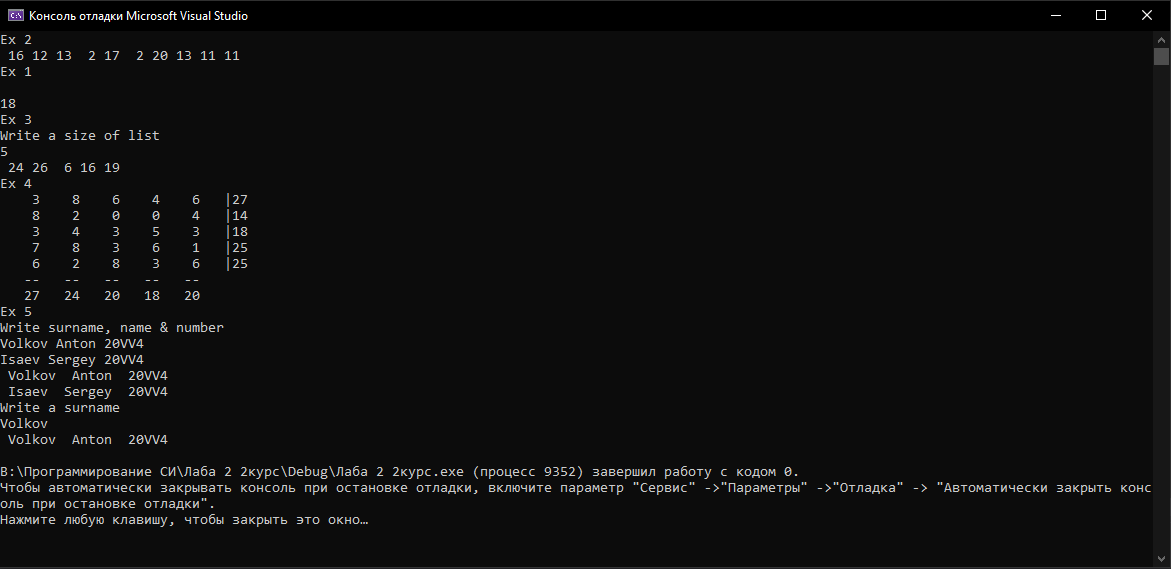
}

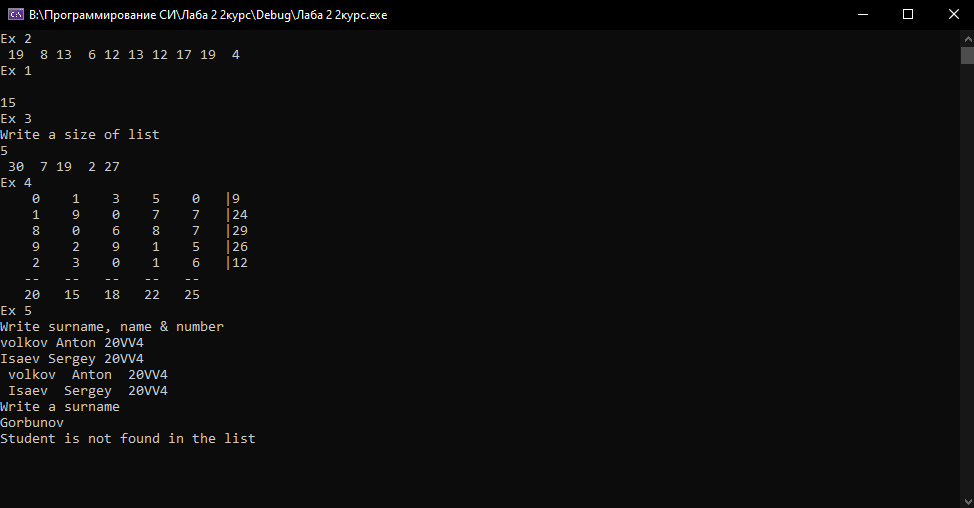
\_getch();

return 0;

}

**Результат работы программы:**

****

****

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