**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра информационных систем**

отчет

**по практической работе №1**

**по дисциплине «Программирование»**

Тема: **Структуры**

|  |  |  |
| --- | --- | --- |
| Студент(ка) гр. |  | Власов И. Д |
| Преподаватель |  | Глущенко А. Г. |

Санкт-Петербург

2025

**Основные теоретические положения.**

## **Одномерные динамические массивы**

Для того чтобы создать в динамической области некоторый объект, необходима одна обычная переменная-указатель (не динамическая переменная). Сколько таких объектов понадобится для одновременной обработки, столько необходимо иметь обычных переменных-указателей. Таким образом, проблема задач неопределенной размерности созданием одиночных динамических объектов решена быть не может.

Решить эту проблему поможет возможность создавать в динамической области памяти массивы объектов с таким количеством элементов, которое необходимо в данный момент работы программы, т. е. создание динамических массивов. Действительно, для представления массива требуется всего одна переменная-указатель, а в самом массиве, на который ссылается этот указатель, может быть столько элементов, сколько требуется в данный момент времени.

## **Изменение размера динамического массива**

Очень часто в процессе работы программы требуется изменять размеры уже созданных и заполненных данными массивов. Общий алгоритм решения этой задачи таков:

1.      создать исходный массив размерности n1 и заполнить его данными;  
2.      создать промежуточный массив размерности n2 (пусть n2 > n1);  
3.      скопировать данные из исходного массива в промежуточный массив;  
4.      освободить память от исходного массива;  
5.      переменной-указателю исходного массива присвоить значение переменной-указателя промежуточного массива;  
6.      заполнить новые элементы массива данными.

## Двумерные динамические массивы

Создание многомерных массивов несколько сложнее, чем одномерных динамических массивов.

**Определение структур**

Структуры представляют собой группы связанных между собой, как правило, разнотипных переменных, объединенных в единый объект, в отличие от массива, все элементы которого однотипны. В языке C++ структура является видом класса и обладает всеми его свойствами.  
Примечание  
Класс - в ООП представляет собой шаблон для создания объектов, обеспечивающий начальные значения состояний: инициализация полей-переменных и реализация поведения функций или методов. ООП в данном курсе не рассматривается, но иногда будут упоминаться понятия, связанные с ним.

Для определения структуры применяется ключевое слово **struct**, а сам формат определения выглядит следующим образом:

struct [имя\_типа] {

тип\_1 элемент\_1;

тип \_2 элемент\_2;

…

тип\_k элемент\_k;

} [ список\_описателей ];

**Инициализация структур**

Инициализация структур аналогична инициализации массивов: в фигурных скобках передаются значения для элементов структуры по порядку. Так как в структуре student первым определено свойство, которое представляет тип **int**- число, то в фигурных скобках вначале идет число. И так далее для всех элементов структуры по порядку.

При определении переменной структуры ее можно сразу инициализировать, присвоив какое-нибудь значение:

struct Student ivan = {23, "Ivan", 0393};

**Структуры как элементы структур**

Кроме базовых примитивных типов данных как **int**или **char**, массивов и указателей в качестве элементов структуры можно использовать другие структуры.

struct Pizza {

char name[20];

int price;

};

struct Client {

char name[20];

char phone[20];

};

struct Order {

struct Client client;

int count;

struct Pizza pizza;

};

**Выполнение работы.**

**1)Меню**

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

2)Создание студентов

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.

3)Вывод студентов

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.