Основы программирования на языке c++ ч.3

Задание 1

1. Создайте переменную, выделив память из “общей кучи”
2. Выведите на экран адрес переменной и её значение
3. Добавьте ввод значения этой переменной с клавиатуры
4. Освободите память
5. Создайте динамический массив (с вводом размерности с клавиатуры)
6. Выведите на экран адрес массива и его значение (по имени)
7. Заполните массив по вашему усмотрению
8. Выведите на экран адрес массива и его значение (по имени)
9. Выведите на экран значение первого элемента массива
10. Измените значение первого эл-та массива, и повторите действия 8 и 9

Задание 1.1

1. Создайте переменную типа int с любым значением
2. Создайте указатель под тип int
3. Создайте ссылку на переменную из п.1
4. Выделите память из кучи под переменную типа int и запишите адрес в указатель из п.2
5. Реализуйте функцию, выводящую значение переданной переменной и её адрес (при возможности). Можно использовать перегрузку

Задание 1.2

1. В программе создать 2 массива: статический на 5 элементов и динамический на 7 элементов.
2. Оба массива не инициализировать
3. Создать функцию, которая выводит массив на экран
4. Функция должна работать с этими обоими массивами (перегрузку не использовать)
5. Вывести значения этих массивов на экран и сравнить их
6. Создайте переменную типа int, задайте ей значение -100000000000(11 нулей)
7. Постарайтесь изменить значение любого элемента любого массива на -1000. Посмотрите на результат
8. Проинициализируйте один элемент массива и измените его значение на -100000000000(11 нулей)

Задание 1.2.1

1. Создайте рекурсивную функцию, вызывающую саму себя с увеличением передаваемого параметра на 1

Задание 1.2.2

1. Через рекурсивную функцию опять запустите подсчет на +1
2. Добавьте возможность пользователю самому выбирать от какого числа вести подсчет
3. Добавьте возможность пользователю самому выбирать до сколько вести подсчёт
4. Добавьте возможность пользователю выбрать число, на которое будет увеличиваться

Перегрузка функций

Функции, имеющие одно имя, но другой список передаваемых параметров - называются перегруженными

int my_func1 (int a)

int my_func1 (double b)

int my_func (int a)

double my_func (int a)

Задание 2

Создайте перегрузку функции по следующим правилам:

1. При передаче в функцию одного натурального числа на экран выводится соответствующее кол-во строк “Hello, World!”
2. При передаче в функцию двух натуральных чисел на экран выводится диапазон натуральных чисел, лежащих между ними
3. При передаче в функцию вещественного числа она должна возвращать только часть после точки (1.6 => 6)
4. При передаче в функцию двух вещественных чисел вернуть наибольшее из них
5. При передаче в функцию статического одномерного массива целых чисел вернуть наименьший элемент

Шаблоны функций

В случае, когда в функцию надо передать одно и то же количество параметров, но тип передаваемых параметров может различаться от вызова к вызову, не затрагивая логику тела функции - можно использовать шаблон функции.

Шаблон функции - некое описание поведения (алгоритм) компилятора при встрече вызова конкретного имени

Шаблоны функций. Пример

```
template <typename T>
T my_func (T mf_param)
{
	if(mf_param !=0)
		return 1;
	else
		return 0;
}
```

Задание ?. Back From The Past

Программа получает на вход 2 вещественных числа

Программа должна вывести:

1. Модули этих чисел
2. Сумму модулей этих чисел
3. Модуль суммы

~~Тут можно реализовать функции~~

Тут можно реализовать шаблоны функций

Задание 3.

Создайте программу, с двумя меню:

В первом попросите пользователя выбрать какой тип двух переменных он хочет ввести (int , float , double)

Во втором меню предложите несколько вариантов действий, реализованных через шаблоны:

1. Поиск максимального
2. Среднее арифметическое
3. Модуль суммы
4. Вывод целой части суммы чисел

Случайные числа

За генерацию случайных чисел отвечает функция rand(), в которую можно передать зерно(seed) для получения тех самых рандомных чисел

Но как постоянно задавать разное зерно для генерации?

Задание 4. Программа “Dynamic Array 1x”

Пользователь вводит размер одномерного массива, который заполняется случайными числами

Реализуйте меню для комфортной работы в вашей программе

Реализуйте функцию замены двух элементов, выбранных пользователем, местами

Добавьте возможность отсортировать массив

Реализуйте возможность удалить элемент массива (для пользователя)

Можно ли добавить функционал на расширение массива?

Задание 5

Пользователь вводит с клавиатуры размер двумерного массива

Далее пользователь выбирает заполнить массив процедурно или вводом с клавиатуры

1. Вывести на экран массив
2. Вывести на экран общее количество элементов
3. Вывести на экран наибольший и наименьший элементы массива
4. Вывести на экран реверсивный массив

Массивы с ништяками

Стоит упомянуть, что существуют несколько “модификаций” массивов:

1. std::array
2. std::vector
3. list из библиотеки <list> (который является не совсем массивом)

Задание 6.

1. В программе нужно создать vector
2. Создать функцию, выводящую адрес каждого элемента vector на экран и его значение
3. Сделайте связку “изменение vector - вызов функции” несколько раз (измените размер vector и значения в нем, после чего вызовите функцию)
4. В каких случаях меняется адрес первого элемента vector?
5. Можно ли вывести значение в ячейках, которые больше не принадлежат vector?

Строки

Строка - массив из char - элементов.

Для работы со строками необходимо подключить библиотеку <string>

Строка довольно гибка:

Можно не указывать размерность, достаточно просто проинициализировать

Строки можно складывать, сравнивать, редактировать и пр.

Чтение/запись файлов

Для работы с файлами необходимо подключить библиотеку <fstream>

Есть 2 класса файлов : ifstream (in) и ofstream (out)

Файл для чтения должен существовать

Файл для записи либо создаётся автоматически, либо перезаписывается

Задание 7.1

Пользователь вводит своё имя, дату рождения и откуда он

В ответ программа выдаёт приветствие в формате

Hello <имя> from <место> <дата рождения> dob. Your age is <текущий возраст>

Не запаривайтесь!!!!1111111

Задание 7.2

Пользователь вводит строку

Программа должна запросить символ и вывести позицию первого вхождения этого символа

Расширьте выводом данных о всех вхождениях

Задание 7. Использование erase

1. Пользователь вводит строку
2. Удалите нулевой элемент строки и выведите её на экран
3. Удалите первые 5 элементов строки и выведите их на кран
4. Позвольте пользователю самому выбрать какой элемент удалить
5. Добавьте пользователю выбор выбор сколько элементов удалить, начиная с нулевого (один или несколько)
6. Расширьте функционал п.5 добавив выбор начала и конца диапазона для удаления

Структуры

В случае необходимости собрать несколько переменных в одну “кучу” можно использовать структуры

```
struct human
{
    int age = 0;
    int weight = 0;
    int id = 0;
};
```

Задание 8

Программа запрашивает данные пользователя:

Дату рождения (день, месяц, год), имя и фамилию.

Вычислить текущий возраст пользователя

Знак зодиака

Вывести всю полученную и вычисленную информацию на экран

Классы

Класс с точки зрения топорного программирования - это структуры с методами

Свойства в классе - это переменные

Методы в классе - это функции

Объект класса - элемент, созданный на основе данного класса

Пример: Класс - мебель (материал, цвет). Объект - табурет (табурет - это мебель со своим цветом и из своего материала)

Синтаксис классов

```
Class Mebel
{
        string name = “ “;
        string material = “ “;
        int id = 0;
        void print()
                {
                        cout << name << material << id;
                }
};
```

Классное задание

Создать класс Cars, содержащий в себе следующие поля:

Марка, Пробег , Год выпуска, Страна - Производитель, Мощность

Программа должна запросить у пользователя количество вводимых автомобилей

После чего пользователь вводит данные о них

По окончании заполнения вывести данные о всех автомобилях на экран, отсортировав их по году выпуска.

Классное задание 2

Создать класс Student, содержащий следующие поля:
имя, возраст, идентификатор, класс обучения

реализовать методы редактирования студента и вывода информации о нем

запросить у пользователя кол-во студентов и создать массив студентов необходимого размера

при создании объекта класса Student выводить сообщение о его создании (объекта)

при удалении объекта класса Student выводить сообщение о его удалении, при этом указывая имя удаляемого студента

Что в каких файлах писать

Исполняемые файлы (.cpp)

- Весь код, в котором происходят вычисления
- Также можно подключать заголовочные файлы и библиотеки

Заголовочные файлы (.h)

- Объявления классов и функций
- Подключение библиотек для корректной работы заголовочного файла

#pragma once и #define

Для того, чтобы код не дублировался дважды в одном документе рекомендуется использовать директиву #pragma once, которая проверяет на единовременное подключение для текущего obj - файла

Еще можно использовать #define, который требует уникального идентификатора. Также его нужно использовать в связке #ifndef #endif

Пример:

```
#ifndef MyFile

#define MyFile

//тут код для исполнения

#endif
```

Мультиклассное задание

Задача: посчитать сумму двух чисел

Создать программу, в main - функции которой есть только вызовы собственных функций

Должно быть 3 файла: main, заголовочный и исполняемый

Функция для примера: getanum получает переменную и текст для вывода сообщения, getanum должна вывести сообщение и изменить значение переданной переменной на то, которое пользователь введет с клавиатуры