**Введение**

Выбор заданий по 2 посл цифрам если цифры например 57 а вариантов 14 ваш вариант 57-14=43-14 ….

Инкапсуляция - сведение кода и данных воедино в одном объекте, получившим название класс.

Наследование - наличие в языке ООП механизма, позволяющего объектам класса наследовать характеристики более простых и общих типов. Наследование обеспечивает как требуемый уровень общности, так и необходимую специализацию.

Полиморфизм - дословный перевод с греческого "много форм". В С++ полиморфизм реализуется с помощью виртуальных функций, которые позволяют в рамках всей иерархии классов иметь несколько версий одной и той же функции. Решение о том, какая именно версия должна выполняться в данный момент, определяется на этапе выполнения программы и носит название позднего связывания.

Существует несколько реализаций системы, поддерживающих стандарт С++, из которых можно выделить реализации Visual C++ (Microsoft) и Builder C++ (Inprise). Отличия относятся в основном к используемым библиотекам классов и средам разработки. В действительности в С++ программах можно использовать библиотеки языка С, библиотеки классов С++, библиотеки визуальных классов VCL (Builder C++), библиотеку MFC (Visual C++ и Builder C++).

Язык С++ является родоначальником множества объектно-ориентированных языков, таких как Java, C#, PHP и др.

Данное пособие предназначено для начинающих изучение технологии ООП для проектирования систем управления на основе С++.

Работа № 1. Программирование алгоритмов с использованием динамических массивов

Теоретические сведения

Задание к работе

Варианты заданий

Контрольные вопросы

Цель работы – научиться использовать операции динамического выделения и освобождения памяти на примере работы с одномерными и двумерными массивами, а также косвенное обращение к элементам массива.

**Теоретические сведения**

Объявление динамического массива

Массивы, создаваемые в динамической памяти, будем называть динамическими (размерность становится известна в процессе выполнения программы). При описании массива после имени в квадратных скобках задается количество его элементов (размерность), например int a[10]. Размерность массива может быть задана только константой или константным выражением.

При описании массив можно инициализировать, то есть присвоить его элементам начальные значения, например:

int а[10] = {1, 1, 2, 2, 5, 100};

Если инициализирующих значений меньше, чем элементов в массиве, остаток массива обнуляется, если больше — лишние значения не используются. Элементы массивов нумеруются с нуля, поэтому максимальный номер элемента всегда на единицу меньше размерности. Номер элемента указывается после его имени в квадратных скобках, например, а[0], а[3].

Если до начала работы программы неизвестно, сколько в массиве элементов, в программе следует использовать динамические массивы. Память под них выделяется с помощью операции new или функции malloc в динамической области памяти во время выполнения программы. Адрес начала массива хранится в переменной, называемой указателем. Например:

int n = 10; int \*mass1 = new int[n];

Во второй строке описан указатель на целую величину, которому присваивается адрес начала непрерывной области динамической памяти, выделенной с помощью операции new. Выделяется столько памяти, сколько необходимо для хранения n величин типа int. Величина n может быть переменной. Инициализировать динамический массив нельзя.

Обращение к элементу динамического массива осуществляется так же, как и к элементу обычного. Если динамический массив в какой-то момент работы программы перестает быть нужным и мы собираемся впоследствии использовать эту память повторно, необходимо освободить ее с помощью операции delete[], например: delete [] a; (размерность массива при этом не указывается).

delete[] mass1;

При необходимости создания многомерных динамических массивов сначала необходимо с помощью операции new выделить память под n указателей (вектор, элемент которого - указатель), при этом все указатели располагаются в памяти последовательно друг за другом. После этого необходимо в цикле каждому указателю присвоить адрес выделенной области памяти размером, равным второй границе массива

mass2=new int\*[row]; // mass2 - указатель на массив указателей на одномерные массивы for(i=0;i<row;i++) mass2[i]=new int[col]; // каждый элемент массива указывает на одномерный for (i=0; i<row;i++) for (j=0;j<col;j++)

Освобождение памяти от двумерного динамического массива:

for(i=0;i<row;i++) //удаление всех одномерных delete[] mass2[i]; // массивов delete[] mass2; // удаление массива указателей на одномерные массивы

**Задание к работе**

Общая постановка. Составить программы - одномерные массивы: задания 1-25, двухмерные массивы: задания 26-50. Массивы создаются в динамической области памяти с использованием операций NEW и DELETE. Ввод исходных данных: реальный размер массивов и их значения. Обращение к элементам массива – через косвенную адресацию.

**Варианты заданий**

16. Заданы два массива А(5) и В(5). В каждом из массивов найти наименьшее значение и разделить на него все элементы массивов. На печать вывести исходные и преобразованные массивы.

41. Задан массив действительных чисел А(n). Необходимо каждый элемент массива разделить на среднее арифметическое этих элементов. На печать вывести исх. и преобразов. массивы.