**Лабораторная работа № 20**

**Работа с динамическим массивом объектов**

**Теоретическое введение.** Классы представляют абстрактные типы данных с открытым интерфейсом и скрытой внутренней реализацией. В классах реализованы базовые принципы *объектно-ориентированного программирования* (ООП):

1. абстракция данных;
2. инкапсуляция – в классах объединяются данные и методы (функции) для работы с ними, так как лишь через методы возможен доступ к сокрытым данным класса;
3. наследование – в производных классах наследуются члены базового класса;
4. полиморфизм – возможность использования одних и тех же методов для работы с различными объектами базового и порожденных им классов.

Определение простейшего класса без наследования имеет вид:

**class имя\_класса {**

**// по умолчанию раздел private – частные члены класса**

**public: // открытые функции и переменные класса**

**};**

Имя класса является новым типом данных. Понятию переменной данного типа соответствует понятие объекта класса. Список членов класса включает описание данных и функций. Функции, описания которых находятся в определении класса, называются функциями-членами класса.

Переменные и функции, объявленные в разделе класса по умолчанию или явно как ***private,*** имеют область видимости в пределах класса. Их можно сделать видимыми вне класса, если объявить в разделе ***public:***.Обычно переменные объявляются в разделе ***private***, а функции в разделе ***public***.

Классами в С++ являются также структуры ***(struct*)** и объединения ***(union*).** В отличие от класса члены структуры по умолчанию являются открытыми, а не закрытыми. Кроме того, объединения не могут наследоваться и наследовать.

При реализации функциональной части класса могут быть использованы функции-члены класса, конструкторы, деструкторы, функции-операторы. Функции класса всегда объявляются внутри класса. Определение функции может находиться и внутри класса. Такие функции называются ***inline***-функциями. Обычно определения функций-членов класса помещаются вне класса. При этом перед именем функции указывается ***имя\_класса::*** .

**тип имя\_класса:: имя\_функции (описание аргументов)**

**{ /\*тело функции\*/ }**

Вызов функций осуществляется одним из двух способов:

**имя\_объекта.имя\_функции(аргументы);**

**указатель\_на\_объект -> имя\_функции(аргументы);**

Обращение к данным объекта класса осуществляется с помощью функций, вызываемых из объектов. При этом функции-члену класса передается скрытый указатель ***this*** на объект, вызывающий функцию.

Функции-«друзья» класса, объявляемые в классе со спецификатором ***friend***, указатель ***this*** не содержат. Объекты, с которыми работают такие функции, должны передаваться в качестве их аргументов. Это обычные функции языка С++, которым разрешен доступ к закрытым членам класса.

**Пример.**

**/\*** Создается класс Student. Формируется динамический массив объектов. При тестировании выводится: сформированный список студентов, список студентов заданного факультета, список студентов для заданных факультета и курса. \*/

#include <iostream>

#include <conio.h>

#include <string.h>

using namespace std;

struct date // дата рождения

{

char daymon[6];

int year;

};

//======= class Student =================

class Student {

char name[30]; //private

date t;

char adr[30], fac[20];

int kurs;

public:

Student();

char \*getfac();

int getkurs();

void show();

};

Student::Student()

{

cout << "Input name:"; cin >> name;

cout << "Input date of born\n";

cout << "Day.mon:";

cin >> t.daymon;

cout << "Year:";

cin >> t.year;

cout << "Input adr:"; cin >> adr;

cout << "Input fac:"; cin >> fac;

cout << "Input kurs:"; cin >> kurs;

}

void Student::show()

{

cout << "Name :" << name << endl;

cout << "Was born :" << t.daymon << '.' << t.year << endl;

cout << "Address :" << adr << endl;

cout << "Fac :" << fac << endl;

cout << "Kurs :" << kurs << endl;

}

char \*Student::getfac() { return fac; }

int Student::getkurs() { return kurs; }

void spisfac(Student spis[], int n)//список студентов заданного факультетата

{

char fac[20];

cout << "Input faculty:";

cin >> fac;

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

if (strcmp(spis[i].getfac(), fac) == 0)spis[i].show();

}

void spisfackurs(Student spis[], int n) //список студентов заданных факультета и курса

{

int i, k;

char fac[20];

cout << "Input faculty:";

cin >> fac;

cout << "Input the course:";

cin >> k;

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

if ((strcmp(spis[i].getfac(), fac) == 0) && (spis[i].getkurs() == k))

spis[i].show();

}

//========= main ================

int main() {

Student \*spis;

int n;

cout << "Input a number of students: ";

cin >> n;

spis = new Student[n];

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

cout << "==============================" << endl;

spis[i].show();

}

spisfac(spis, n);

spisfackurs(spis, n);

delete[] spis;

cout << "press any key!";

while (!\_kbhit());

return 0;

}

# Задания для самостоятельного решения

Разработать классы для описанных ниже объектов. Включить в класс методы set (…), get (…), show (…). Определить другие методы.

1. **Abiturient**: Фамилия, Имя, Отчество, Адрес, Оценки. Создать массив объектов. Вывести:

а) список абитуриентов, имеющих неудовлетворительные оценки;

б) список абитуриентов, сумма баллов у которых не меньше заданной;

в) выбрать N абитуриентов, имеющих самую высокую сумму баллов, и список абитуриентов, имеющих полупроходной балл.

1. **Aeroflot**: Пункт назначения, Номер рейса, Тип самолета, Время вылета, Дни недели. Создать массив объектов. Вывести:

а) список рейсов для заданного пункта назначения;

б) список рейсов для заданного дня недели;

в) список рейсов для заданного дня недели, время вылета для которых больше заданного.

1. **Book**: Автор, Название, Издательство, Год, Количество страниц.

Создать массив объектов. Вывести:

а) список книг заданного автора;

б) список книг, выпущенных заданным издательством;

в) список книг, выпущенных после заданного года.

1. **Worker**: Фамилия и инициалы, Должность, Год поступления на работу, Зарплата. Создать массив объектов. Вывести:

а) список работников, стаж работы которых на данном предприятии превышает заданное число лет;

б) список работников, зарплата которых больше заданной;

в) список работников, занимающих заданную должность.

1. **Train**: Пункт назначения, Номер поезда, Время отправления, Число общих мест, Купейных, Плацкартных. Создать массив объектов. Вывести:

а) список поездов, следующих до заданного пункта назначения;

б) список поездов, следующих до заданного пункта назначения и отправляющихся после заданного часа;

в) список поездов, отправляющихся до заданного пункта назначения и имеющих общие места.

1. **Product**: Наименование, Производитель, Цена, Срок хранения, Количество. Создать массив объектов. Вывести:

а) список товаров для заданного наименования;

б) список товаров для заданного наименования, цена которых не превышает указанной;

в) список товаров, срок хранения которых больше заданного.

1. **Patient**: Фамилия, Имя, Отчество, Адрес, Номер медицинской карты, Диагноз. Создать массив объектов. Вывести:

а) список пациентов, имеющих данный диагноз;

б) список пациентов, номер медицинской карты которых находится в заданном интервале.

1. **Bus**: Фамилия и инициалы водителя, Номер автобуса, Номер маршрута, Марка, Год начала эксплуатации, Пробег. Создать массив объектов. Вывести:

а) список автобусов для заданного номера маршрута;

б) список автобусов, которые эксплуатируются больше 10 лет;

в) список автобусов, пробег у которых больше 10 000 км.

1. **Customer**: Фамилия, Имя, Отчество, Адрес, Телефон, Номер кредитной карточки, Номер банковского счета. Создать массив объектов. Вывести:

а) список покупателей в алфавитном порядке;

б) список покупателей, номер кредитной карточки которых находится в заданном интервале.

1. **File**: Имя файла, Размер, Дата создания, Количество обращений. Создать массив объектов. Вывести:

а) список файлов, упорядоченный в алфавитном порядке;

б) список файлов, размер которых превышает заданный;

в) список файлов, число обращений к которым превышает заданное.

1. **Word**: Слово, Номера страниц, на которых слово встречается (от 1 до 10), Число страниц. Создать массив объектов. Вывести:

а) слова, которые встречаются более чем на N страницах;

б) слова в алфавитном порядке;

в) для заданного слова номера страниц, на которых оно встречается.

1. **House**: Адрес, Этаж, Количество комнат, Площадь. Создать массив объектов. Вывести:

а) список квартир, имеющих заданное число комнат;

б) список квартир, имеющих заданное число комнат и расположенных на этаже, который находится в определенном промежутке;

в) список квартир, имеющих площадь, превосходящую заданную.

1. **Phone**: Фамилия, Имя, Отчество, Адрес, Номер, Время внутригородских разговоров, Время междугородних разговоров. Создать массив объектов. Вывести:

а) сведения об абонентах, время внутригородских разговоров которых превышает заданное;

б) сведения об абонентах, воспользовавшихся междугородней связью;

в) сведения об абонентах, выведенные в алфавитном порядке.

1. **Person**: Фамилия, Имя, Отчество, Адрес, Пол, Образование, Год рождения. Создать массив объектов. Вывести:

а) список граждан, возраст которых превышает заданный;

б) список граждан с высшим образованием;

в) список граждан мужского пола.

1. **Student**: Фамилия, Имя, Отчество, Дата рождения, Адрес, Телефон, Факультет, Курс. Создать массив объектов. Вывести:

а) список студентов заданного факультета;

б) списки студентов для каждого факультета и курса;

в) список студентов, родившихся после заданного года.