Практическое занятие № 14 «Доступ к элементам класса в С++»

Учебные цели:

- получение умений и навыков доступа к элементам класса в С+.

Воспитательные цели:

- формировать диалектико-материалистическое мировоззрение;
- формировать навыки самостоятельности и дисциплинированности;
- стимулировать активную познавательную деятельность обучаемых, способствовать формированию у них творческого мышления.

Категория слушателей: 2,3 курс.
Время: 90 мин.
Место проведения: компьютерный класс.
Материально-техническое обеспечение:
1) персональный компьютер *IBM PC* с операционной системой Windows XP; 2) среда разработки приложений *Visual C++.NET.*

ПЛАН ПРАКТИЧЕСКОГО ЗАНЯТИЯ

Учебные вопросы	Время, мин
Вступительная часть	5
1. Доступ к элементам класса в С++	15
2. Выполнение индивидуального задания.	65
Заключительная часть	5

Элементы теории

Классы служат для определения *типов объектов*. Объект отличается от значения тем, что он обладает *состоянием*, которое может меняться в течение жизни объекта. В то время как тип данных определяет множество значений посредством множества операций, нельзя сказать то же самое о типе объектов. Никакой тип объектов не определяет множества объектов. Скорее тип объектов определяет *множество состояний* объекта посредством *множества методов*. Среди методов объекта мы можем выделить *конструкторы*, служащие для инициализации объектов, *мутаторы*, служащие для изменения состояния объектов, и *анализаторы*, служащие для анализа состояния объекта.

После описания класса можно объявить одну или несколько переменных и констант, содержащих объекты данного класса, например:

```
Student Ivanov, Petrov;
```

После обработки такого объявления в С++ каждой переменной выделяется память для хранения объекта (т.е. всех полей) данного класса. Для дальнейшего доступа к открытым полям и методам объекта используется

операция «.» (точка), аналогичная операции доступа к полям структуры, например:

```
Ivanov.Set_ocenka_Inf(4);
```

В C++ также можно описать переменную-указатель на объект:

```
Student* pk;
```

Такая переменная сначала инициализируется указателем на конкретный объект после чего с этим объектом можно оперировать посредством операции «->», например:

```
pk = new Student;
pk-> Set_ocenka_Inf(4);
int oc_inf = pk-> Get_ocenka_Inf();
```

Каждый метод класса имеет скрытый параметр – указатель `this`. Этот указатель содержит адрес текущего объекта. Рассмотренные ранее функции `Set_ocenka_Inf()` и `Get_ocenka_Inf()` также содержат этот параметр.

В листинге 1 приведен пример использования указателя `this` в явном виде.

Листинг 1.

```
class Student
{
public:
void Set_ocenka_Inf (int ocenka)
{this->oc_inf=ocenka;}
int Get_ocenka_Inf( )
{return this->oc_inf;}
void Set_ocenka_Math (int ocenka)
{oc_math=ocenka;}
int Get_ocenka_Math( )
{return oc_math;}
private:
int oc_inf, oc_math;
};
```

В функциях `Set_ocenka_Inf()` и `Get_ocenka_Inf()` при обращении к переменным класса `Student` указатель `this` используется в явном виде. В функциях `Set_ocenka_Math()` и `Get_ocenka_Math()` такое обращение осуществляется неявно. Несмотря на различие в синтаксисе, оба варианта идентичны.

В дальнейшем при обсуждении проблемы перегрузки операторов будет приведено несколько реальных примеров использования указателя `this`. В данный момент необходимо понимать, что `this` – это указатель, хранящий адрес объекта, в котором он используется.

Память для указателя `this` не выделятся и не освобождается программно. Эту задачу берет на себя компилятор.

Функции-мутаторы состояния `Set_ocenka_Inf()` и `Set_ocenka_Math()` были определены выше как не возвращающие

значения. При использовании подобных связанных функций иногда возникает желание выстроить операции в цепочку. Для этого требуется, чтобы функции возвращали ссылку на измененный объект. Например, мы могли бы написать на C++:

```
void sessia (Student& Fam, int exm_inf, int exm_math)
{
Fam.Set_ocenka_Inf(exm_inf).Set_ocenka_Math(exm_math);
 }
```

чтобы выставить `Fam` экзаменационные оценки по дисциплинам «Информатика» и «Математика». Для этого надо, чтобы функции возвращали ссылку на `Student`. Опишем теперь измененную структуру класса `Student`:

```
class Student
{
public:
void sessia (Student& Fam, int exm_inf, int exm_math)
{
Fam.Set_ocenka_Inf(exm_inf).Set_ocenka_Math(exm_math);
}
Student& Set_ocenka_Inf (int ocenka)
{this->oc_inf=ocenka;
return *this;} // функция возвращает ссылку на объект
int Get_ocenka_Inf( )
{return this->oc_inf;}
Student& Set_ocenka_Math (int ocenka)
{oc_math=ocenka;
return *this;} // функция возвращает ссылку на объект
int Get_ocenka_Math( )
{return oc_math;}
private:
int oc_inf,oc_math;
};
```

Тело функции `main()` будет иметь вид:

```
int main()
{
Student Ivanov,Petrov;
Ivanov.sessia(Ivanov,4,5);
Ivanov.sessia(Petrov,3,3);
cout <<"Inf="<<Ivanov.Get_ocenka_Inf()<<endl;
cout<<"Math="<<Ivanov.Get_ocenka_Math()<<endl;
cout <<"Inf="<<Petrov.Get_ocenka_Inf()<<endl;
cout<<"Math="<<Petrov.Get_ocenka_Math()<<endl;
}
```

Результат работы программы представлен на рис. 1.

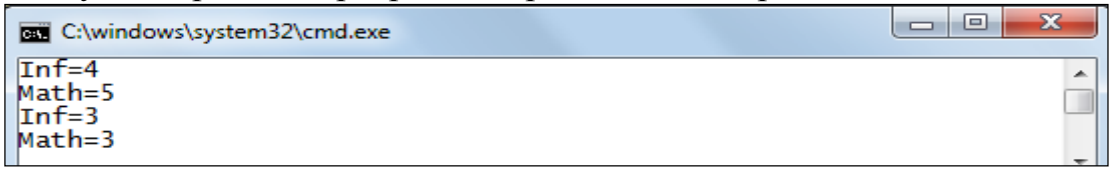

Рис. 1. Результат работы программы с использованием указателя this

Вопросы для самопроверки

1. Дать определение понятию объект класса.
2. Поясните назначение методов объекта: конструктор, мутатор и анализатор?
3. Как объявить объект класса, массив объектов?
4. Как можно обратиться к открытым полям и методам объекта?
5. Как можно обратиться к закрытым полям объекта?
6. Какое значение содержит указатель this?
7. Приведите пример использования в программах указателя this.

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ОТРАБОТКЕ УЧЕБНЫХ ВОПРОСОВ

1. Доступ к элементам класса в C++

Задание 1. Для класса `C_pacient`, созданного в предыдущей лабораторной работе создать **методы** класса для выполнения заданных функций по обработке информации: вывести на экран список пациентов, лежащих в палате номер N, где N – вводится с клавиатуры. Кроме указания фамилии и инициалов пациентов, вывести их пол, данные наблюдения за их температурой и среднее значение температуры за время наблюдения.

Структура класса `C_pacient`: фамилия и инициалы пациента, пол, возраст, наименование подразделения, специальное звание, диагноз, номер палаты, наименование отделения больницы, массив температур (с датами измерения) пациента, количество дней наблюдения за пациентом

Пример выполнения задания 1.

Создадим методы класса `C_pacient`, позволяющие вывести на экран список пациентов (фамилии и инициалы), лежащих в палате номер N, где N – вводится с клавиатуры, а также их пол, данные наблюдения за их температурой и среднее значение температуры за время наблюдения. Описание класса `C_pacient` с новыми методами приведено в листинге 2.

Листинг 2.

```
struct d_tem
{
     char data[11];
     float temperatura;
};

class C_pacient
{
     public:

     char* GetFam  (void)
     {    return cFam_in;
```

```
        };

        void SetFam (int i)
        {char newline;
        cout << "Vvedite FIO "<<i<<"-ogo pacienta: ";
        cin.get(cFam_in, 29, '\n');
        cin.get(newline);
        };

        char* GetPol  (void)
        {    return cpol;
        };

        void SetPol(int i)
        {char newline;
        cout << "Vvedite pol "<<i<<"-ogo pacienta: ";
        cin.get(cpol, 5, '\n');
        cin.get(newline);};


        void SetVozrast (int vozr)
        {
        cvozrast=vozr;
        }
        int GetVozrast () const
        {
        return cvozrast;
        }

        char* Get_Naim_Podr (void)
        {    return cnaim_podr;
        };

        void Set_Naim_Podr(int i)
        {char newline;
        cout << "Vvedite nainmenovanie podrazdelenia "<<i<<"-
ogo pacienta: ";
        cin.get(cnaim_podr, 50, '\n');
        cin.get(newline);};

        char* Get_Spec_zv (void)
        {    return cspec_zv;
        };

        void Set_Spec_zv(int i)
        {char newline;
        cout << "Vvedite specialnoe zvanie "<<i<<"-ogo
pacienta: ";
        cin.get(cspec_zv, 30, '\n');
        cin.get(newline);};

        void SetdTem(int m)
        {char newline;
```

```
for (int i=0; i<m; i++)
{ cout << "Vvedite "<<i+1<<"-yu datu: ";
cin.get(dTem[i].data, 11, '\n');
cin.get(newline);
cout << "Vvedite temperaturu: ";
cin>>dTem[i].temperatura;
cin.get(newline);   }
};
void GetdTem(int m) const
{ for (int i=0; i<m; i++)
cout << dTem[i].data<<'\t'<<dTem[i].temperatura<<endl;
cout << endl;
};

void SetkDen(int m)
{
ckDen=m;
};

int GetkDen(void) const
{return ckDen;};

void den(int m) const
{ int k;
k=m%10;
switch (k)
{
case 1: cout <<" den' ";
     break;
case 2: cout<<" dnay ";
     break;
case 3: cout<<" dnay ";
     break;
case 4: cout<<" dnay ";
     break;
default : cout <<" dney "; };
};

void Srtem (int m) const
{    float sum=0;
     for (int i=0; i<m; i++)
     sum+=dTem[i].temperatura;
     cout << "Sr. tem-ra ="<< sum/m <<" (za "<<m;
     den(m);
     cout<<" nabludeniay)"<<endl;
};

void Set_n_pal(int n)
{
cnom_pal=n;
};

int Get_n_pal(void) const
```

```
        {return cnom_pal;};

        int Poisk (C_pacient& bolnoy, int N_pal) const
        {
        if (bolnoy.Get_n_pal()==N_pal)
        {cout<<bolnoy.GetFam()<<"\t"<<bolnoy.GetPol()<<"\t"
             <<" nabludenie: "
              <<bolnoy.GetkDen()<<" ";
        bolnoy.den(bolnoy.GetkDen());
        cout<<endl;
        bolnoy.GetdTem(bolnoy.GetkDen());
        bolnoy.Srtem(bolnoy.GetkDen());
        return 1;
        }
        else return 0;
        }

    private:

        char cFam_in[30];
        char cpol[5];
        int cvozrast;
        char cnaim_podr[50];
        char cspec_zv[30];
        int cnom_pal;
        d_tem dTem[10];
        int ckDen;
        };

    int main()
    {

        C_pacient bolnye[30];
        int n,m,nom_pal,vozrast;
        char newline;
        cout<<"Vvedite kol-vo pacientov n=";
        cin>>n;
        cin.get(newline);
        for (int i=0; i<n; i++)
             {cout<<i+1<<". ";
             bolnye[i].SetFam(i+1);
             bolnye[i].SetPol(i+1);
             cout<<"Vvedite vozrast bolnogo: ";
             cin>>vozrast;
             cin.get(newline);
             bolnye[i].SetVozrast(vozrast);
             bolnye[i].Set_Naim_Podr(i+1);
             bolnye[i].Set_Spec_zv(i+1);
             cout<<"Vvedite nomer palaty dlay razmechenia
bol'nogo N=";
             cin>>nom_pal;
             bolnye[i].Set_n_pal(nom_pal);
             cout<<"Vvedite kol-vo dney nabludeniay m=";
```

```
            cin>>m;
            cin.get(newline);
            bolnye[i].SetkDen(m);
            bolnye[i].SetdTem(m);
            }
        cout << endl;
        int s=1, sum;
        while (s)
        {
         cout<<"Vvedite 1 - dlay poiska pacientov"<<endl;
         cout<<"Vvedite 0 - dlay vyhoda iz programmy."<<endl;
         cin>>s;
         cin.get(newline);
        if (s==0) return 0;
        cout<<"Vvedite nomer palaty N=";
        cin>>nom_pal;
        sum=0;
        for (int i=0; i<n; i++)
        sum+=bolnye[i].Poisk(bolnye[i],nom_pal);
        if (sum==0) cout<< "V palate N="<<nom_pal
                        <<" pacientov net!";
        cout<<endl;
        }
}
```

2. Выполнение индивидуального задания

Задание 1. В соответствии с Вашим вариантом опишите **методы** класса для выполнения заданных функций по обработке информации:

1. Вывести на экран фамилии всех студентов N группы, где N – вводится с клавиатуры.

Структура класса с именем STUDENT:

- Фамилия;
- Имя;
- Отчество;
- Номер группы;
- Успеваемость (массив оценок по любой учебной дисциплине размерностью не менее 5 элементов).

2. Вывести на экран фамилии всех студентов, имеющих хотя бы одну неудовлетворительную оценку.

Структура класса с именем STUDENT:

- Фамилия и инициалы;
- Название факультета;
- Год рождения;
- Номер группы;
- Успеваемость (массив оценок по любой учебной дисциплине размерностью не менее 5 элементов).

3. Вывести на экран фамилии всех студентов, средний балл которых ниже 3.0.

Структура класса с именем STUDENT:

- Фамилия и инициалы;
- порядковый номер;
- Номер группы;
- Успеваемость (массив оценок по любой учебной дисциплине размерностью не менее 5 элементов).

4. Вывести на экран номера рейсов, на которые есть еще свободные места.

Структура класса с именем AEROFLOT:

- Название пункта назначения рейса;
- Номер рейса;
- Тип самолёта;
- Время вылета;
- Количество свободных мест.

5. Вывести на экран номера рейсов, стоимость билетов на которые не превышает N рублей. N – вводится с клавиатуры.

Структура класса с именем AEROFLOT:

- Название пункта назначения рейса;
- Номер рейса;
- Тип самолёта;
- Время полёта;
- Стоимость билета.

6. Вывести на экран сотрудников (фамилии), родившихся в N-ом месяце. N - вводится с клавиатуры.

Структура класса с именем SPISOK:

- Фамилия;
- Имя;
- Отчество;
- Номер телефона;
- Стаж работы;
- День рождения;
- Месяц рождения (номер месяца);
- Год рождения.

7. Вывести на экран сотрудников (фамилии и год рождения), старше N лет. N – вводится с клавиатуры.

Структура класса с именем SPISOK:

- Фамилия и инициалы;
- Номер телефона;
- Должность;
- День рождения;
- Месяц рождения;
- Год рождения.

8. Вывести на экран сотрудников (фамилии), чей стаж работы более N лет. N – вводится с клавиатуры.

Структура класса с именем SPISOK:

- Фамилия;
- Имя;
- Отчество;
- Номер телефона;
- Должность;
- Стаж в работы.

9. Вывести на экран информацию о книгах (фамилии автора и название книги), имеющихся в библиотеке, год издания которых равен N. N − вводится с клавиатуры.

Структура класса с именем KNIGA:

- Фамилия автора;
- Название книги;
- Издательство;
- Год издания;
- Количество страниц.

Контрольные вопросы

1. Дать определение понятиям класс, объект.
2. Основные концепции объектно-ориентированного программирования.
3. Инкапсуляция, наследование, полиморфизм.
4. Объектно-ориентированный подход к разработке программ.
5. История и основы языка C++.
6. Примеры простых программ на языке C++.
7. Объектно-ориентированные средства языка C++.
8. Объекты, классы.
9. Инкапсуляция данных и методы доступа?