**Открытые и закрытые члены класса**

В отличии от полей структуры, доступных всегда, в классах могут быть члены и методы различного уровня доступа:

• открытые public (публичные), вызов открытых членов и методов класса осуществляется с помощью оператора . ("точка");

• закрытые private (приватные), доступ к которым возможен только с помощью открытых методов.

• защищенные методы (protected).

После описания класса необходимо описать переменную типа class.

Например, name\_class name;

здесь name\_class – имя класса, name – имя переменной.

Переменная типа class называется «объектом» или «экземпляром класса». Объявление переменной типа class (в нашем примере переменная name типа name\_class) называется созданием (инициализацией) объекта (экземпляра класса).

После описания переменной можно обращаться к членам и методам класса. Обращение к членам и методам класса осуществляется аналогично обращению к полям структуры с помощью оператора «.» (точка).

1name.p1; //Обращение к полю p1 экземпляра класса name.

name.f1(par1,par2,…parn); //Обращение к методу f1 экземпляра класса name,

//par1, par2, …, parn – список формальных параметров функции f1.

Члены класса доступны из любого метода класса и их не надо передавать в качестве параметров функций-методов.

ЗАДАЧА 2.1. Рассмотрим класс complex для работы с комплексными числами. В классе complex будут члены класса:

• double x – действительная часть комплексного числа;

• double y – мнимая часть комплексного сила.

методы класса:

• double modul() – функция вычисления модуля комплексного числа;

• void show\_complex() – функция выводит комплексное число на экран.

Ниже приведен текст класса и функция main, демонстрирующая работу с классом.

#include <iostream>

#include <string>

#include <math.h>

const double PI=3.14159;

using namespace std;

class complex //Определяем класс complex

{

public:

double x; //Действительная часть комплексного числа.

double y; //Мнимая часть комплексного числа.

//Метод класса complex – функция modul, для вычисления модуля комплексного //числа.

double modul()

{

return pow(x\*x+y\*y,0.5);

}

//Метод класса complex – функция show\_complex, для вывода комплексного //числа.

void show\_complex()

{

if (y>=0) //Вывод комплексного числа с положительной мнимой частью.

cout<<x<<"+"<<y<<"i"<<endl;

else //Вывод комплексного числа с отрицательной мнимой частью.

cout<<x<<y<<"i"<<endl;

}

};

int main()

{

complex chislo; //Определяем переменную chislo типа complex.

chislo.x=3.5; //Определяем действительную часть комплексного числа.

chislo.y=-1.432; //Определяем мнимую часть комплексного числа.

chislo.show\_complex(); //Вывод комплексного числа, chislo.show\_complex() –обращение //к методу класса.

cout<<"Modul' chisla="<<chislo.modul(); //Вывод модуля комплексного числа, //chislo.modul() – обращение к методу класса.

return 0;

}

Результат работы программы:

3.5-1.432i

Modul chisla=3.78162

Использование открытых членов и методов позволяет получить полный доступ к элементам класса, однако это не всегда хорошо. Если все члены класса объявить открытыми, то при непосредственном обращении к ним появится потенциальная возможность внести ошибку в функционирование взаимосвязанных между собой методов класса. Поэтому, общим принципом является следующее: «Чем меньше открытых данных о классе используется в программе, тем лучше». Уменьшение количества публичных членов и методов позволит минимизировать количество ошибок. Желательно, чтобы все члены класса были закрытыми и тогда невозможно будет обращаться к членам класса непосредственно с помощью оператора «.» Количество открытых методов также следует минимизировать.

Если в описании элементов класса отсутствует указание метода доступа, то члены и методы считаются закрытыми (private). Принято описывать методы за пределами класса.

ЗАДАЧА 2.2. Изменим рассмотренный ранее пример класса complex. Добавим метод vvod, предназначенный для ввода действительной и мнимой части числа, члены класса и метод show\_complex сделаем закрытыми, а остальные методы открытыми. Текст программы будет иметь вид:

class complex //Определяем класс complex

{

//Открытые методы.

public:

void vvod();

double modul();

//Закрытые члены и методы.

private:

double x;

double y;

void show\_complex();

};

//Описание открытого метода vvod класса complex.

void complex::vvod()

{

cout<<"Vvedite x\t";

cin>>x;

cout<<"Vvedite y\t";

cin>>y;

// Вызов закрытого метода show\_complex из открытого метода vvod.

show\_complex();

}

//Описание открытого метода modul класса complex.

double complex::modul()

{

return pow(x\*x+y\*y,0.5);

}

//Описание закрытого метода show\_complex класса complex.

void complex::show\_complex()

{

if (y>=0)

cout<<x<<"+"<<y<<"i"<<endl;

else cout<<x<<y<<"i"<<endl;

}

int main()

{

complex chislo;

chislo. vvod();

cout<<"Modul kompleksnogo chisla="<<chislo.modul();

return 0;

}

Результат работы программы:

Vvedite x 3

Vvedite y -1

3-1i

Modul kompleksnogo chisla=3.16228

В рассмотренном примере показано совместное использование открытых и закрытых элементов класса. Разделение на открытые и закрытые в этом примере несколько искусственное, оно проведено только для иллюстрации механизма совместного использования закрытых или открытых элементов класса. Если попробовать обратиться к методу show\_complex() или к членам класса x , y из функции main, то компилятор выдаст сообщение об ошибке (доступ к элементам класса запрещен).