Этапы развития программирования

- «Блюдо спагетти»
- Процедурное программирование
- Структурное программирование
- Объектное программирование
- Объектно-ориентированное программирование

Чего нам не хватает?

- Предметная область задачи, как правило, выражается в терминах объектов, а не функций и структур.
- Заказчик и разработчик говорят на разных языках, что приводит к неправильной постановке задачи.

Объект

- Состояние (state)
- Поведение (behavior)
- Уникальность (identity)

Поведение

- Способность реагировать на внешние события
- Способность воздействовать на другие объекты

Состояние

- Влияет на поведение объекта
- Может меняться в зависимости от внешних воздействий
- Объект имеет набор (пространство) разрешенных состояний

Уникальность

- Каждый объект можно отличить от других при любых условиях
- Объекты дискретны (измеряются в «штуках»)

Класс

Класс – совокупность объектов, обладающих подобными свойствами.

Класс характеризуется в первую очередь поведением, характерным для всех его представителей.

Сообщения

Сообщения – способ взаимодействия объектов.

Поведение объекта определяет набор сообщений, на которые он может реагировать, а также которые он может посылать другим объектам.

Инкапсуляция

Инкапсуляция – сокрытие состояния объекта

Первичным является поведение объекта. О состоянии объекта можно узнать через его поведение.

Абстракция

Абстракция – выделение свойств объектов, которые представляют интерес с точки зрения конкретной задачи.

Любой класс является абстракцией.

Класс комплексных чисел

```
Complex a (1, 0), b (0,1), c;

c = a+b;

double re = c.re ();

if (a == b){
        …
}
```

Прототип класса Complex

```
class Complex{
private:
        //Атрибуты
        double real, image;
public:
        //Конструктор
        Complex (double _re = 0, double _im = 0);
        //Деструктор (виртуальный)
        virtual ~Complex ();
        //Методы
        double re () const;
        double im () const;
        double abs () const;
        double arg () const;
        //...
};
```

Прототип класса Complex

```
class Complex{
        //...
public:
        //Конструктор копии
        Complex (const Complex & src);
        //Оператор присваивания
        Complex & operator = (const Complex & arg);
        //Операторы сравнения
        bool operator == (const Complex & arg) const;
        bool operator != (const Complex & arg) const;
        //Арифметические операторы
        Complex operator + (const Complex arg) const;
        Complex operator * (const Complex arg) const;
        //...
};
```

Атрибуты и методы классов

Атрибуты – в C++ аналог полей структур. Отображают состояние объектов данного класса.

Методы – отображают поведение объектов класса (сообщения, которые объекты могут воспринимать).

В C++ методы – это функции, являющиеся полями класса. Метод может быть применен к объекту (экземпляру) класса. При этом объект выступает в роли одного из аргументов класса.

Модификаторы доступа

public: поле доступно всем.

private: поле доступно методам класса и друзьям класса.

protected: поле доступно методам класса, подклассов и друзьям класса.

Реализация методов класса

```
class Complex{
        //...
};

double Complex::abs () const{
        return sqrt (real*real + image*image);
}
```

```
class Complex{
        //...
        double re () const{
                return this -> real; //return real
        }
        //...
};
```

Конструкторы

```
class Complex{
        //...
        Complex (double _re = 0, double _im = 0);
        Complex (const Complex & src);
        //...
};

Complex::Complex (double _re, double _im){
        real = _re;
        image = _im;
}

Complex::Complex (const Complex & src){
        real = src.real;
        image = src.image;
}
```

Вызовы конструкторов

```
Complex a (1,0);
Complex b;                          //Complex b (0,0)
Complex c = Complex (0,1);          //Complex c (0,1)
Complex d (a), e = b;               //конструктор копии
Complex * cptr = 0;
cptr = new Complex (1,1);

Complex x;
x = a;                              //присваивание

x = Complex (1, 1);                 //?
```

Предопределенные конструкторы

- ClassName (); - конструктор по умолчанию. Существует, если не определен ни один конструктор. Вызывает конструкоры по умолчанию для всех атрибутов.

- ClassName (const & ClassName src); - конструктор копии. Существует, если не определен явный конструктор копии. Вызывает конструкторы копии для всех атрибутов.

Деструкторы

~ClassName ();

Деструктор вызывается при уничтожении стековой переменной, либо при вызове оператора delete.