Основы объектно-ориентированного программирования

Введение

Лекция 1

2008 г

Содержание

- Этапы развития программирования
- Программная инженерия
- Фундаментальные понятия ООП
- Этапы разработки программ с использованием ООП
- Основополагающие концепции ООП
- Инкапсуляция
- Наследование
- Полиморфизм
- Объекты
- Анатомия класса
- Управление доступом к элементам класса
- Объявление класса в программе. Пример1. «Класс TPerson»
- Контрольные вопросы
- Мини-тест
- Список литературы

Этапы развития программирования

Технология программирования – это система методов, способов и приемов обработки и выдачи информации.

- Написание программ в машинных кодах
- Появление языков низкого уровня
- Языки высокого уровня. Технология нисходящего структурного программирования
- Появление ООП

Предпосылки и история

- Первый кризис программирования
- Повторное использование кода
  - Модульное программирование
- Рост сложности программ
  - Структурное программирование
- Модификация программ
  - Объектно-ориентированное программирование
- Продолжение кризиса программирования

Повторное использование кода

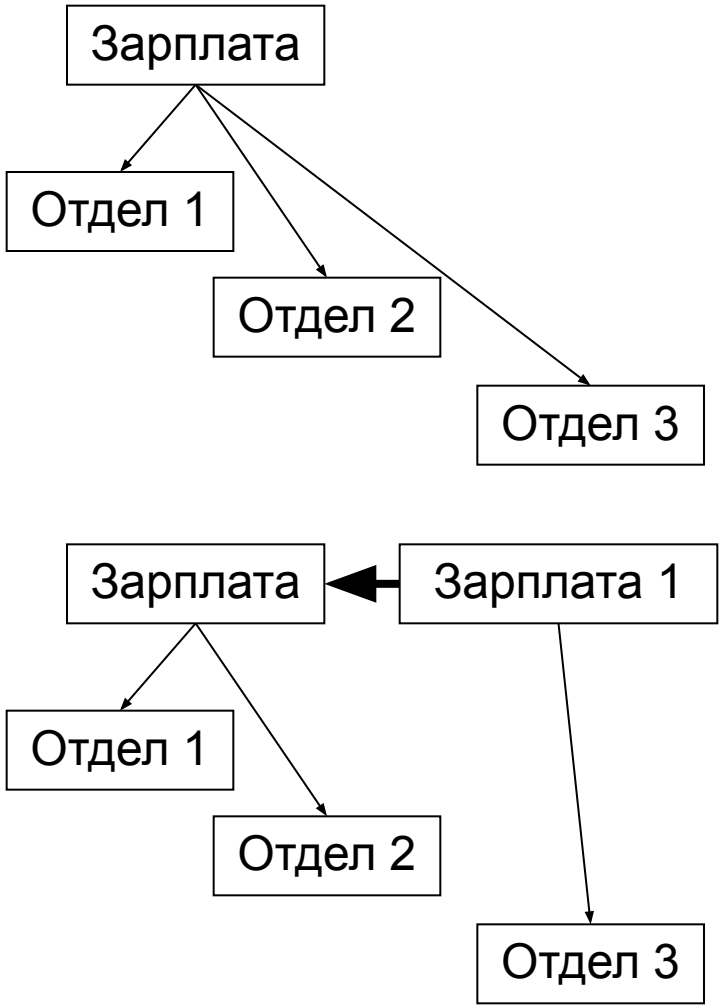
- Проблема
  - Дублирование фрагментов кода
- Модульное программирование
  - Выделение фрагментов в модули
  - Повторное использование модулей
  - Создание библиотек модулей

Рост сложности программ

- Проблема
  - Сложные программные комплексы
    - Объем кода, к-во связей, к-во разработчиков, к-во пользователей
  - Жизненный цикл: стадии внедрения и сопровождения
- Структурное программирование
  - «Правильное» проектирование и кодирование
  - Основные принципы:
    - Нисходящее проектирование
    - Применение специальных языков проектирования
    - Дисциплина проектирования и разработки:
      - планирование и документирование проекта
      - поддержка соответствие кода проектной документации
    - Структурное кодирование (линейный блок, If-then-else, цикл)

Модификация программ

- Проблема
  - изменения в проекте и программе без изменения ранее написанного кода
- Объектно-ориентированное программирование
  - Класс – модуль со свойствами, поведением, обязанностями
  - Парадигмы ООП:
    - Инкапсуляция и сокрытие деталей
    - Наследование
    - Полиморфизм

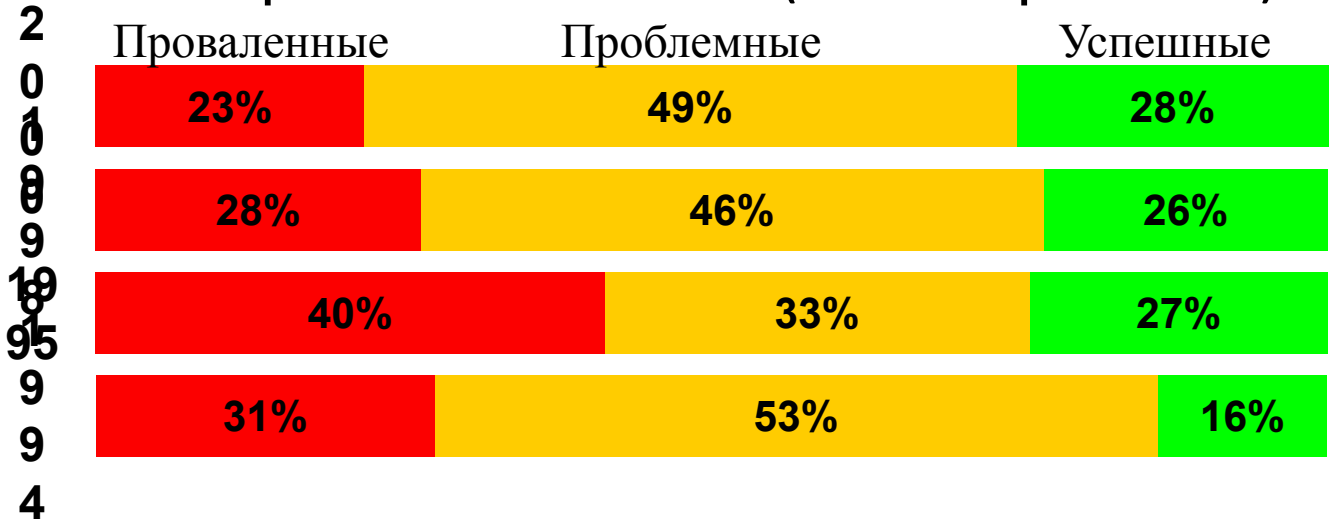


Некоторые итоги

- Главная цель программной инженерии - сокращение стоимости ПО
- Сформировались основные принципы и методы проектирования ПО:
  - Жизненный цикл ПО
  - Модульное программирование
  - Структурное проектирование и программирование
  - Объектно-ориентированное проектирование и программирование

Продолжение кризиса

- Кризис программирования принимает хронические формы:
  - США тратит более $200 млрд. на более чем 170 тыс. проектов
  - потери от недополученного эффекта измеряются триллионами.
- Успешные проекты не часты (30000 проектов)



Источник: The Standish Group International, Inc., Extreme Chaos, 2000
http://www1.standishgroup.com//sample_research/PDFpages/extreme_chaos.pdf

Определения

Программная инженерия – это

- установление и использование обоснованных инженерных принципов (методов) для экономного получения ПО, которое надежно и работает на реальных машинах. [Bauer 1972].
- та форма инженерии, которая применяет принципы информатики (computer science) и математики для рентабельного решения проблем ПО. [CMU/SEI-90-TR-003]
- применение систематического, дисциплинированного, измеряемого подхода к разработке, использованию и сопровождению ПО [IEEE 1990].
- дисциплина, целью которой является создание качественного ПО, которое завершается вовремя, не превышает выделенных бюджетных средств и удовлетворяет выдвигаемым требованиям [Schach, 99]

Абстрактные и реальные объекты

- Абстрактный объект это описание реального объекта минус подробности
- Пример 1:

- Абстрактные человек-это описание человека, которое содержит атрибуты и варианты поведения:
  - Имя
  - Фамилия
  - Рост
  - Вес
- Реальный человек определяет значения атрибутов:
  - Майкл
  - Джексон
  - 180 см
  - 80 кг

Пример 2: формочки для печенья

Почему объекты?

- Фокусирование на объектах упрощает для нас понимание сложных вещей.
- Уделяем внимание лишь важным аспектам
- Пример : преподаватель - студент

Объекты в деловом мире

- Бизнес процесс: Заказ товара.
- Объекты:
  - Форма заказа
  - Список товаров
  - Товарная накладная
  - Ордер на отгрузку
  - Метка отгрузки

Поведение:

- Ввод информации
- Изменение информации
- Удаление информации
- Отображение формы
- Обработка заказа
- Отмена заказа

Концепция

Класс и *Объект*

- **Класс** - принципиально новый тип данных.
- **Класс** представляет собой множество **объектов**
  - имеющих общую структуру
  - обладающих одинаковым поведением.
- **Класс** является дальнейшим развитием типа структура (запись)

Концепция

- **Объект** является представителем (**экземпляром**) какого-либо класса.
- Объект **обладает**
  - состоянием
  - поведением
  - идентичностью.
- **Состояние объекта** характеризуется
  - набором его свойств (атрибутов)
  - текущими значениями каждого из этих свойств.
- **Поведение объекта** - выполнения определенной последовательности характерных для него действий.
- **Идентичность объекта** – это свойство (или набор свойств) объекта, которое позволяет отличить его от всех прочих объектов того же типа (класса).

Класс

простое объяснение

- Класс – это шаблон который определяет атрибуты и методы реального мира.
- Пример: форма для печенья в виде буквы «А» - это не буква А, она лишь определяет , как буква А выглядит.
- Если вам необходим объект, представляемый классом, создаете экземпляр.
- Каждый экземпляр содержит те же самые атрибуты и методы, которые определены в классе. У каждого экземпляра своя копия.
- Метод – поведение объекта. (опять про печенье, но в форме собаки)

Этапы разработки программ с использованием ООП

ООП представляет собой ***систематизированный подход*** к алгоритмической формализации ***сложных предметных областей***.

ООП предполагает этапы разработки программ:

- Первый этап: **абстрагирование**
- Второй этап: **инкапсуляция**
- Третий этап: **модульность**
- Четвертый этап: **иерархия**

Этапы разработки программ с использованием ООП

Абстрагирование

- Выделение абстракций. Это - **анализ предметной** области, для которой составляется программа, с целью определения
  - основных объектов этой предметной области,
  - их свойств,
  - отношений между объектами,
  - возможных операций над объектами или их составляющими.

Различие:

Процедурное *программирование нацелено на* ***моделирование действий****, выполняемых* ***компьютером****.*

ООП *нацелено на* ***моделирование предметной области*** *решаемой задачи.*

Этапы разработки программ с использованием ООП

Инкапсуляция

- Типизация объектов и синтез абстрактных типов данных. Определение новых типов данных и наборов специфических функций и операций, применяемых к этим типам данных.

Модульность

- Объектная декомпозиция. Выделение подтипов (подобъектов) для каждого из типов и их составляющих.

Иерархия

- Композиционная иерархизация объектов. Т.е. выделение родовитых и композиционных отношений над объектами.

Парадигмы ООП:

- ***инкапсуляция***
- ***наследование***
- ***полиморфизм***

Инкапсуляция (encapsulation)

- это ***объединение*** производного ***типа данных с набором функций***, используемых при работе с этим типом данных, в единый класс.

- Функции, включенные в класс, называют **методами класса**
- Данные – элементами или **полями класса**,
- Конкретные представители класса – **объекты, экземпляры**.

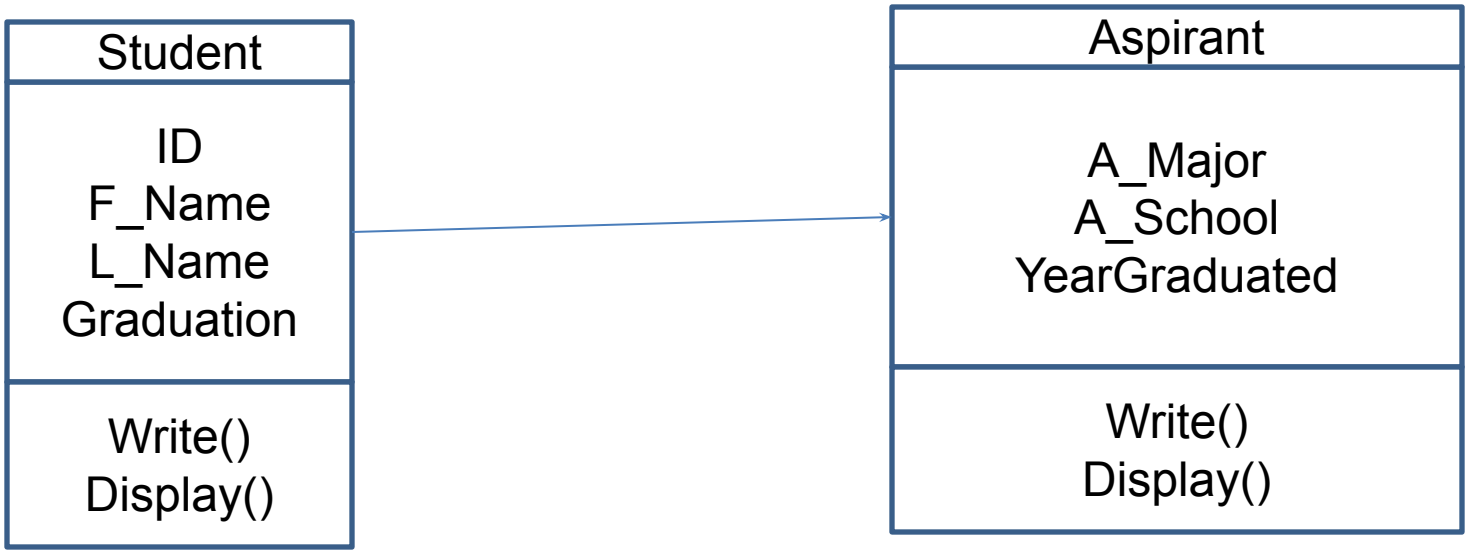
Класс (объект) - это то, что поддерживает инкапсуляцию

- Инкапсуляция позволяет сделать класс «самодостаточным» для решения конкретной задачи.
- Класс всегда несет в себе некоторую функциональность.
- Это мощное средство обмена готовыми к работе программными заготовками

Инкапсуляция и ограничение доступа к данным

- Инкапсуляция предполагает возможность ограничения доступа к данным (полям) класса.
- Это позволяет
  - **упростить интерфейс** класса, показав наиболее существенные для внешнего пользователя данные и методы.
  - обеспечить **возможность внесения изменений** в реализацию класса без изменения других классов (важно для дальнейшего сопровождения и модернизации программного кода).
- При сокрытии полей объекта **доступ** к ним осуществляется **только** посредством **методов класса**. Это защищает данные от внешнего вмешательства или неправильного использования

Инкапсуляция



Управление доступом

Ключи доступа

- ***private*** - элементы данных могут использоваться только функциями-методами класса, к которому принадлежат эти элементы данных
- ***public*** - элементы данных могут использоваться любыми функциями программы
- ***protected*** - элементы данных могут использоваться функциями-методами того же класса, к которому принадлежат эти элементы данных, а также функциями- методами производных классов (классов-потомков)

По умолчанию *ключ доступа* ***private****. Т.е. если ключи доступа не указаны, то все элементы класса являются скрытыми (недоступными).*

Попытка обратиться в программе к скрытым данным или методам вызывает сообщение:

<имя элемента класса> is not accessible

Наследование (inheritance)

- это **возможность определять новые классы** посредством добавления полей, свойств и методов к **уже существующим классам**.

- Такой механизм получения новых классов называется ***порождением***.
- При этом новый, порожденный, класс (**потомок**) **наследует все** поля, методы и свойства своего базового, родительского класса.
- Наследование поддерживает концепцию **иерархии классов** (hierarchical classification).
- Применение иерархии классов делает управляемыми большие потоки информации.
- Наследование **обеспечивает поэтапное создание сложных классов** и разработку собственных библиотек классов

Полиморфизм (polymorphism)

Полиморфизм (polymorphism)

- Это возможность **заменить** в классе потомке **метод** класса родителя, **сохранив** при этом **имя** метода.
- Это свойство классов решать схожие по смыслу проблемы разными способами.

Цель полиморфизма - **использование одного имени** для задания **общих** для класса **действий**.

- Для изменения метода необходимо **перекрыть** его в потомке, т.е. **объявить** в потомке **одноименный** метод и реализовать в нем нужные действия.
- В результате объекте-родителе и объекте-потомке будут действовать **два одноименных метода**, имеющие **разную алгоритмическую основу**.
- Концепция полиморфизма - идея «один интерфейс - множество методов».
- Полиморфизм позволяет манипулировать объектами различной степени сложности путем создания общего для них стандартного интерфейса для реализации похожих действий.

Описание класса в Delphi

- type
- < имя класса > = class(< имя класса-родителя >)
- public
- < описание общедоступных элементов >
- published
- < описание элементов, доступных в Инспекторе Объектов >
- protected
- < описание элементов, доступных в классах-потомках >
- private
- < описание элементов, доступных только в модуле >
- end;

Пример описания класса

```
TControl = class(TComponent)
      private
            FOnDblClick: TNotifyEvent;
            FOnHouseDown: TMouseEvent;
            FOnHouseHove:  TMouseMoveEvent;
      protected
            property OnDblClick: TNotifyEvent read FOnDblClick write FOnDblClick;
            property OnMouseDown: TMouseEvent read FOnMouseDown write
    FOnMouseDown;
            property OnMouseMove: TMouseMoveEvent read FOnMouseMove write
    FOnMouseMove;
end;
```

Пример описания класса

```
type
    TStudent = class
        FAge  :  integer;
        function GetAge  :  integer;
        procedure SetAge(Value  :  integer);
        property Age  :  integer read GetAge write SetAge;
end;
```

Пример 1. Объявление класса в программе. Постановка задачи

Пусть необходимо создать класс, описывающий некоторые характеристики человека.

- Будем хранить о человеке следующие данные:

  фио, рост в см, вес в кг.

- Для работы с этими данными предусмотрим следующие методы:
  - методы **инициализации полей** (названия таких методов имеют префикс Set); назначение этих методов - присваивать новые значения полям класса; новые значения полей будут предаваться через параметры функций-методов;
  - метод, позволяющий **получить информацию о текущем состоянии объекта** (значении всех его полей).

Пример 1. Объявление класса в программе. Программная реализация

```
#include <stdio.h>
#include <string.h>
//описание структуры класса
class TPerson
{
 private:
     char fname[15];
     int frost;
     float fwes;
 public:
  char *    Show() ;
  void SetName (char* value);
  void SetRost  (int  value);
  void SetWes  (float  value);
} ;
```

Пример 1. Объявление класса в программе. Реализация методов класса

```
//реализация методов класса
 char* TPerson::Show()
   {static char S[100];
      sprintf (S,"Это %s; его рост - %d см, вес - %3.2f кг",  fname,  frost,
      fwes);
    return S;  }
void TPerson::SetName(char* value)
    { strcpy(fname,value);
     return;    }
void TPerson::SetRost(int value)
    { frost=value;
     return;    }
 void TPerson::SetWes(float value)
    { fwes=value;
     return;    }
```

Итоги
Контрольные вопросы

- Перечислите основные этапы развития технологии программирования.
- Определите понятие и назначение класса
- Определите понятие и характеристики объекта
- Перечислите и охарактеризуйте этапы разработки программ с использованием ООП
- Назовите и охарактеризуйте основные концепции ООП
- Из каких элементов состоит класс?
- Как ограничить доступ к составляющим класса?

вопрос 1

1. Что такое **класс**?

– Это модуль, сохраненный в файле
– Это функция
– Это библиотека
– Это тип данных
– Это специальная программа

вопрос 2

2. Что такое методы класса?

– Это инкапсулированные в классе данные
– Это функции для работы с полями класса
– Это специальные функции в составе ОС Windows

вопрос 3

2. Что такое полиморфизм?

– Это переключатель
– Это спец.функция для работы с полями класса
– Это одно имя – разный функционал