CS Teaching Academy

Chapter 6, Module 2: Programming Paradigms / OOP

Objectives

- Identify and differentiate major programming paradigms
- Understand and explain principles of Object Oriented Programming

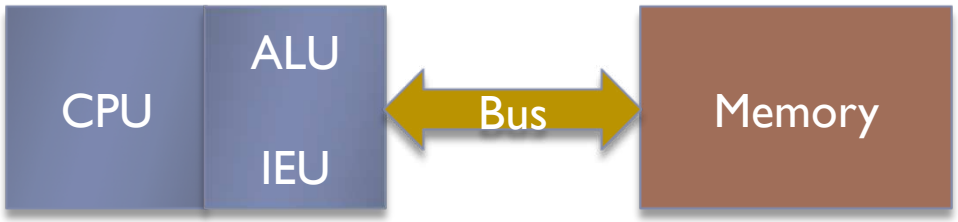
Programming Paradigms

- Functional
  - Based on Lambda Calculus (theory of functions)
  - Ex: Lisp, Scheme, Racket
- Declarative
  - Based on First Order Predicate Logic or Production System
  - Ex: Prolog (FOPL), OPS5, CLIPS, Jess (PS)
- Imperative
  - Based on Von Neumann architecture
  - Procedural or Object Oriented

Programming Today

Imperative Programming

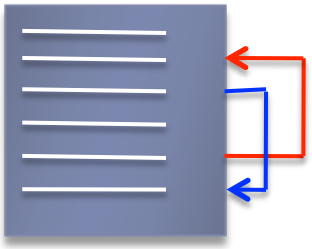
- Von Neumann Machine (John von Neumann, 1903-1957)



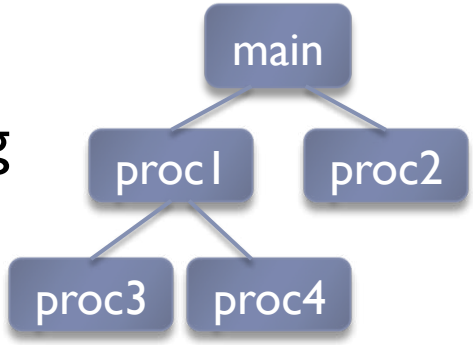
1. Instruction execution is sequential
2. Program variables represent memory
3. Primary goal is efficient use of hardware (CPU, memory)

Imperative Programming Generations

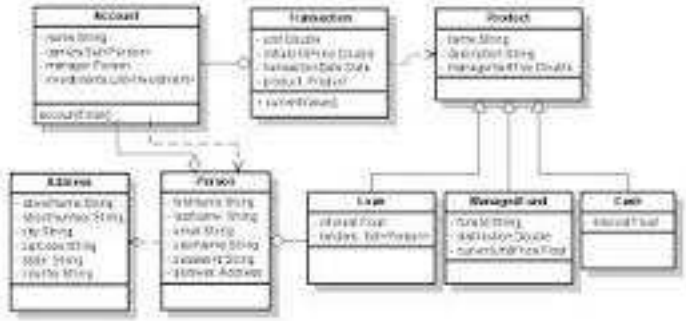
- Gen 1: Linear Programming (go-to's)
  - "spaghetti code"



- Gen 2: Structured programming
  - procedural decomposition



- Gen 3: Object-Oriented Programming
  - abstraction, encapsulation, inheritance



Principles of OOP

1. Abstraction
2. Inheritance
3. Encapsulation
4. Data Hiding
5. Polymorphism

Abstraction

- Abstraction:
  - (n): a model of an application domain entity
  - (v): to create models in software
- Invented for complex system modeling & simulation
  - Simula modeling language
- Expanded for general programming in SmallTalk
- Ex: Banking System abstractions
  - account, customer, transaction, money, statement, employee, ...

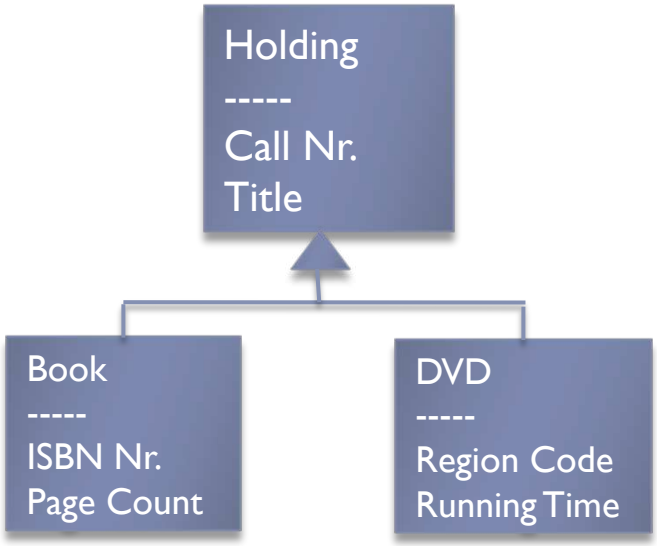
Abstraction

- Each abstraction produces a set of objects (a class)
- Each object is unique
- Components of an Abstraction
  - Attributes (instance variables, unique to objects)
  - Methods (procedures, shared by objects)

Inheritance

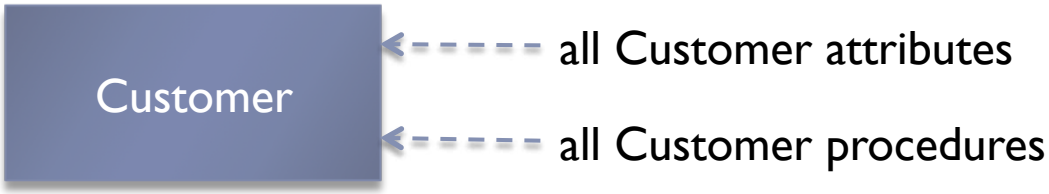
- Software reuse through expansion
  - Inheritance creates sub-classes
  - Sub-classes add attributes and methods to parents

- Ex: Library



Encapsulation

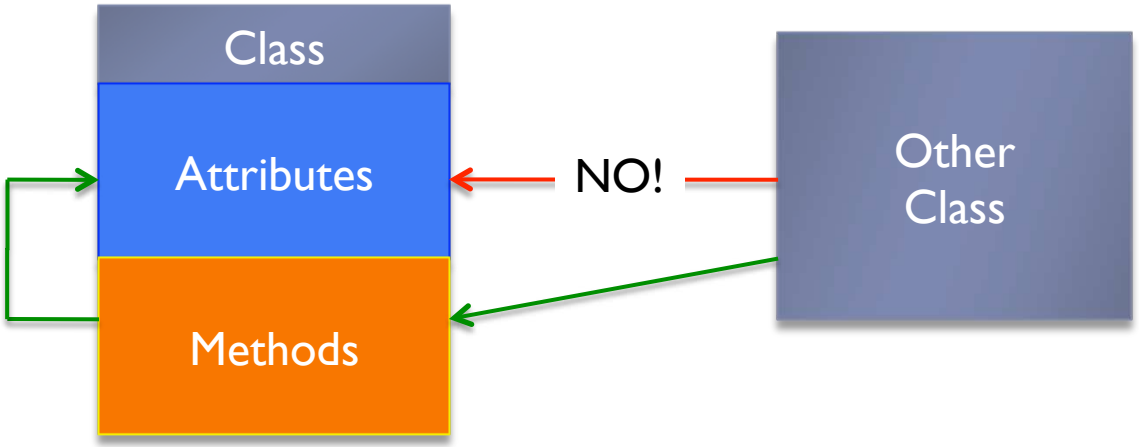
- Each entity is enclosed in a single abstraction



- A significant change from structured programming
  - attributes are scattered among procedures

Data Hiding

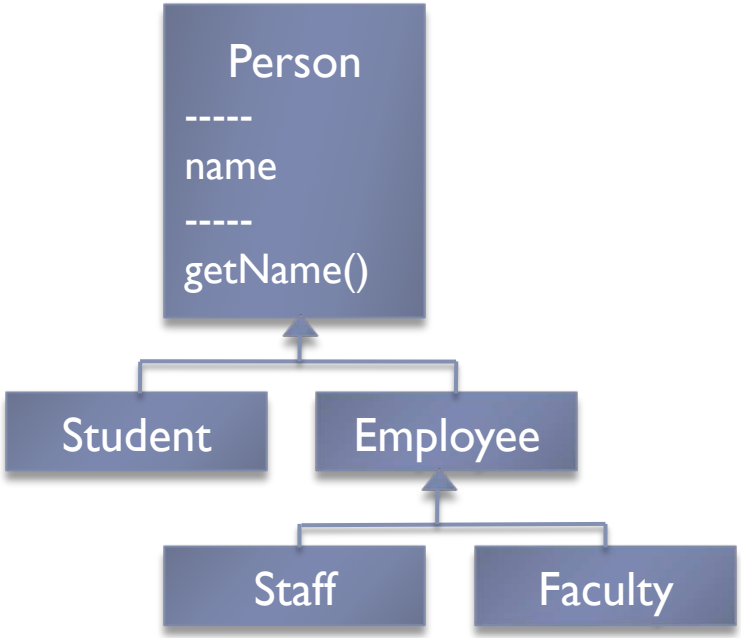
- Attributes are hidden from the outside world
- Access is controlled through methods



- Advantages:
  - Promotes correctness (access to attributes is validated)
  - Allows attributes to be modified independently of other classes

Polymorphism

- "Multiple forms"
- Access can be generic, via inheritance structures




```
Faculty fac = new Faculty();
String name = fac.getName();

displayPerson(fac);
```

```
procedure displayPerson(Person p)
{
  ...
}
```

Summary

- Programming Paradigms
  - Linear (spaghetti code)
  - Structured (procedural decomposition)
  - Object-Oriented (abstraction)

- Principles of OOP
  - Abstraction
  - Inheritance
  - Encapsulation (is NOT data hiding)
  - Data Hiding
  - Polymorphism