Benazir Bhutto Shaheed University Lyari, Karachi

Department of Computing Science & Information Technology

Objected Oriented Programming (OOP) CS and IT 2nd Semester (CS-IT-321)

Topic: OOP Concepts & Principles

Object

Definition: a thing that has identity, state, and behavior

- identity: a distinguished instance of a **class**
- state: collection of values for its **variables**
- behavior: capability to execute **methods**

* variables and methods are defined in a class

Class

Definition: a collection of data (fields/ variables) and methods that operate on that data

- data/methods define the contents/capabilities of the instances (objects) of the class
- a class can be viewed as a *factory* for objects
- a class defines a *recipe* for its objects

Package

- Collection of Classes
  - Preventing naming conflicts.
    - For example there can be two classes with name Employee in two packages, college.staff.cse.Employee and college.staff.ee.Employee

Software Evolution

- To built today's complex software it is just not enough to put together a sequence of programming statements and sets of procedures and modules; we need to incorporate sound construction techniques and program structures that are easy to comprehend, implement and modify.
- Sincethe invention of the computer, many programming approaches have been tried.

Software Evolution

- Such as Modular programming, top-down programming, bottom-up programming and structured programming.
  - The primary motivation in each case has been the concern to handle the increasing complexity of programs that are reliable and maintainable.

OOP Principles

- There are four main building blocks of OOP
  - Abstraction
  - Encapsulation
  - Inheritance
  - Polymorphism

Abstraction

- Abstraction is one of the key concepts of object-oriented programming (OOP) languages. Its main goal is to handle complexity by hiding unnecessary details from the user. That enables the user to implement more complex logic on top of the provided abstraction without understanding or even thinking about all the hidden complexity.

Encapsulation

- A key OOP concept: “Information Hiding”
- Key points
  - The user of an object should have access only to those methods (or data) that are essential
  - Unnecessary implementation details should be hidden from the user
  - In Java/C++, use classes and access modifiers (public, private, protected)

Inheritance

- Inheritance:
  - programming language feature that allows for the implicit definition of variables/methods for a class through an existing class
- Subclass relationship
  - B is a subclass of A
  - B inherits all definitions (variables/methods) in A
- Superclass variables, subclass objects
  - Polymorphism and dynamic binding

Polymorphism

- “Many forms”
  - allow several definitions under a single method name
- Example:
  - “move” means something for a person object but means something else for a car object
- Dynamic binding:
  - capability of an implementation to distinguish between the different forms during run-time

Static & Dynamic Binding

- Static:
  - Static binding happens at compile-time
- Dynamic:
  - Dynamic binding happens at runtime.

Access Modifiers

- Public:
  - **public** means everyone is allowed to access
- Private:
  - **private** means that only members **of the** same class are allowed to access
- Protected:
  - **protected** means that members of subclasses are also allowed

Dry Run

- Dry running a program involves the programmer working through a program on paper, usually using a table called a ‘trace table’.
- The programmer adds columns for any variables or expressions that are important.
- When this has been done, the programmer works through the program, line by line, filling into the trace table the values of variables and expressions as they change.
- By doing this, the programmer can spot the exact position when things start going wrong with the program - when variables suddenly contain unexpected values or expressions don't hold the expected state.