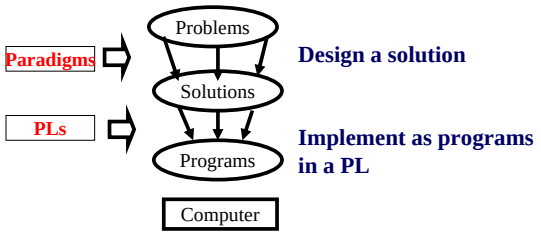
Programming Paradigms (Styles)

Problem Solving via Programs

- A computer is a tool that solves problems by means of programs written in PLs.
- A typical program development process:
  - Requirement analysis (specification)
  - Design a solution
  - Implement as programs in a PL
  - Verification/validation

Paradigms and Programs



What Is a Programming Paradigm?

- A collection of conceptual patterns that together mold the program design process and ultimately determine a program's structure.
  - How we think about and formulate solutions
  - The form of programs

Programming Paradigms and PLs

- Relationship between PPs and PLs:
  - Once we visualize a solution via a paradigm's conceptual pattern, we express it in a PL
  - A PL that reflects a particular paradigm well, this PL is said to support that paradigm
- Does a PL support only one paradigm ?
  - No, typically it supports one paradigm very well.

Programming Paradigms

- Operational Approach
  - Describes step-by-step how to construct a solution.
- Definitional (Declarative) Approach
  - States properties about the solution.
- Demonstrational Approach
  - Demonstrates solutions (examples) to specific instances of simple problems.

Programming Paradigms

- Operational Approach
  - Imperative (Procedural) Programming
  - Object-Oriented Programming
- Definitional (Declarative) Approach (Paradigms)
  - Functional Programming
  - Logic Programming
- Demonstrational Approach (Paradigms)
  - Program by example
  - Visual Programming

Imperative Programming Paradigms

- Characterized by an abstract model of a computer that consists of a single large global store.
- The computer stores an encoded representation of a computation and executes a sequence of command or expressions that modifies the store.
- Best represented by Von-Neumann style computer.

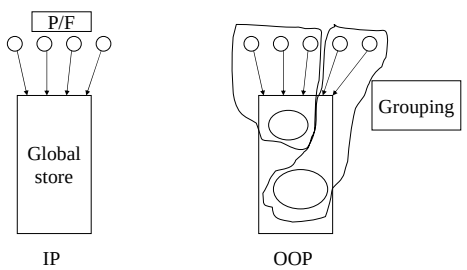
Imperative (Procedure-Oriented) Thinking

- How to do IP?
  - Determine what data values will be required.
  - Represent these data values by associating them with storage locations.
  - Derive a step-by-step sequence of transformation to the store.
  - The final state represents the result values (solution).

Object-Oriented Programming Paradigms

- In IP, each procedure/function deals with the concrete representation of data values in a single global store.
- In OOP, each procedure/function operates on abstract values, rather than on stored representation.
  - Objects

Imperative vs. Object-Oriented Programming



Object-Oriented Thinking

- How to do OOP ?
  - Define suitable objects for the problem.
  - Use these objects to describe step-by-step sequence of operations.

Functional Programming Paradigms

- Based on the mathematical model of functions and function composition.
  - No concepts of a storage location that can be assigned or modified. (no variables, no commands)
- Referentially transparent
  - Easier to understand program
- Functions treated as first class values
  - No restriction of use

Function-Oriented Thinking

- How to do FP ?
  - Define functions (using function composition)
  - Use a function application

Logic Programming Paradigms

- Based on the mathematical model of facts and rules that allow deduction of other facts.
  - A logic program = facts + rules

Logic-Oriented Thinking

- How to do LP ?
  - Define facts
  - Define rules
  - Use a query

Parallel/Concurrent Programming Paradigms

- Cooperating parallel process
  - asynchronous
- Single process applied to many data
  - synchronous

Parallel/Concurrent Programming

- How to use P/C P?
  - Automatically detect parallelism in a sequential pgm.
  - Add mechanisms that describe parallelism..

Programming Paradigms