What are Programming Paradigms?

- “The concepts and abstractions used to represent the elements of a program (e.g., objects, functions, variables, constraints, etc.) and the steps that compose a computation (assignment, evaluation, continuations, data flows, etc.).”
- Or, a way to classify the style of programming.

Most Languages Are Hybrids!

- This makes it hard to teach to students, because most languages have facets of several paradigms!
  - Called ‘Multi-paradigm’ languages
  - Scratch too!
- It’s like giving someone a juice drink (with many fruit in it) and asking to taste just one fruit!

Functional Programming (review)

- Computation is the evaluation of functions
  - Plugging pipes together
  - Each pipe, or function, has exactly 1 output
  - Functions can be input!
- Features
  - No state
  - No mutation
  - No side effects
- Examples (the not pure)
  - Scheme, Scratch BYOB

Imperative Programming

- “Sequential” Programming
  - Computation a series of steps
    - Assignment allowed
    - Mutation allowed
    - Changing variables
  - Like following a recipe. E.g.,
    - Procedure f(x)
      - ans = x
      - ans = ans + ans
      - ans = (x+3) * ans
      - return ans
  - Examples: (the not pure)
    - Pascal, C

Object-Oriented Programming (OOP)

- Objects as data structures
  - With methods you ask of them
  - These are the behaviors
  - With local state, to remember
  - These are the attributes
- Classes & Instances
  - Instance an example of class
    - E.g., fluffy is instance of Dog
- Inheritance saves code
  - Hierarchical classes
    - E.g., pianist special case of musician, a special case of performer
- Examples (the not pure)
  - Java, C++

OOP Example: SketchPad

- Dr. Ivan Sutherland
  - “Father of Computer Graphics”
  - 1988 Turing Award (“Nobel prize” for CS)
  - Wrote Sketchpad for his foundational 1963 thesis
- The most impressive software ever written
- First…
  - Object-oriented system
  - Graphical user interface
  - Non-procedural language

http://www3.ntu.edu.sg/home/ehchua/programming/java/images/OOP-Objects.gif
http://en.wikipedia.org/wiki/Sketchpad

Visualization of Programming Concepts

- “The concept will look like this”
- Programming Paradigms
  - Paradigms...
  - Paradigms...

Visual representation of concepts:

1. Programming Paradigms
2. Paradigms...
3. Paradigms...
4. Paradigms...
5. Paradigms...
6. Paradigms...
7. Paradigms...
8. Paradigms...
9. Paradigms...
10. Paradigms...

Example:

- E.g., x = (x+3) * x
  - The value of x is computed by evaluating the expression (x+3) * x
  - The result of the evaluation is then assigned back to the variable x

http://en.wikipedia.org/wiki/Imperative_programming
http://en.wikipedia.org/wiki/Imperative_programming

References:

- en.wikipedia.org/wiki/Programming_paradigm
- en.wikipedia.org/wiki/Functional_programming
- en.wikipedia.org/wiki/Imperative_programming
- en.wikipedia.org/wiki/Object-oriented_programming
- en.wikipedia.org/wiki/Sketchpad

Visual explanations of concepts:

1. Paradigms...
2. Paradigms...
3. Paradigms...
4. Paradigms...
5. Paradigms...
6. Paradigms...
7. Paradigms...
8. Paradigms...
9. Paradigms...
10. Paradigms...

Example explanations:

- E.g., x = (x+3) * x
  - The value of x is computed by evaluating the expression (x+3) * x
  - The result of the evaluation is then assigned back to the variable x

Visual representations of key concepts:

- Paradigms...
- Paradigms...
- Paradigms...
- Paradigms...
- Paradigms...
- Paradigms...
- Paradigms...
- Paradigms...
- Paradigms...
- Paradigms...

Example representations:

- E.g., x = (x+3) * x
  - The value of x is computed by evaluating the expression (x+3) * x
  - The result of the evaluation is then assigned back to the variable x

Visual aids for learning concepts:

- Paradigms...
- Paradigms...
- Paradigms...
- Paradigms...
- Paradigms...
- Paradigms...
- Paradigms...
- Paradigms...
- Paradigms...
- Paradigms...
**Declarative Programming Example**

- Betty
  - Kitty was 2nd
  - I was 3rd
- Ethel
  - I was on top
  - Joan was 2nd
- Joan
  - I was 3rd
- Kitty
  - I came out 2nd
- Mary
  - I was 4th
- Betty
  - I was 1st

**Turing Completeness**

- A Turing Machine has an infinite tape of 0s and 1s and instructions that say whether to move the tape left, right, read, or write it.
- A Universal Turing Machine is one that can simulate a Turing machine on any input.
- A language is considered Turing Complete if it can simulate a Universal Turing Machine.

**Ways to Remember the Paradigms**

- Functional
  - Evaluate an expression and use the resulting value for something
- Object-oriented
  - Send messages between objects to simulate the temporal evolution of a set of real world phenomena
- Imperative
  - First do this and next do that
- Declarative
  - Answer a question via search for a solution

**Summary**

- Each paradigm has its unique benefits
  - If a language is Turing complete, it is equally powerful
  - Paradigms vary in efficiency, scalability, overhead, fun, "how" vs "what" to specify, etc.
- Modern languages usually take the best from all
  - E.g., Scratch
    - Can be functional
    - Can be imperative
    - Can be object-oriented
    - Can be declarative