Jonathan McKinsey

The Beauty and Joy of Computing

Lecture #5
Programming Paradigms

Programming Language Trends

- Python usage rises with growing popularity of Data Science.
- Java and C still dominate the job market.
- The demand for functional paradigm support increases with Cloud Computing.

Programming Paradigms

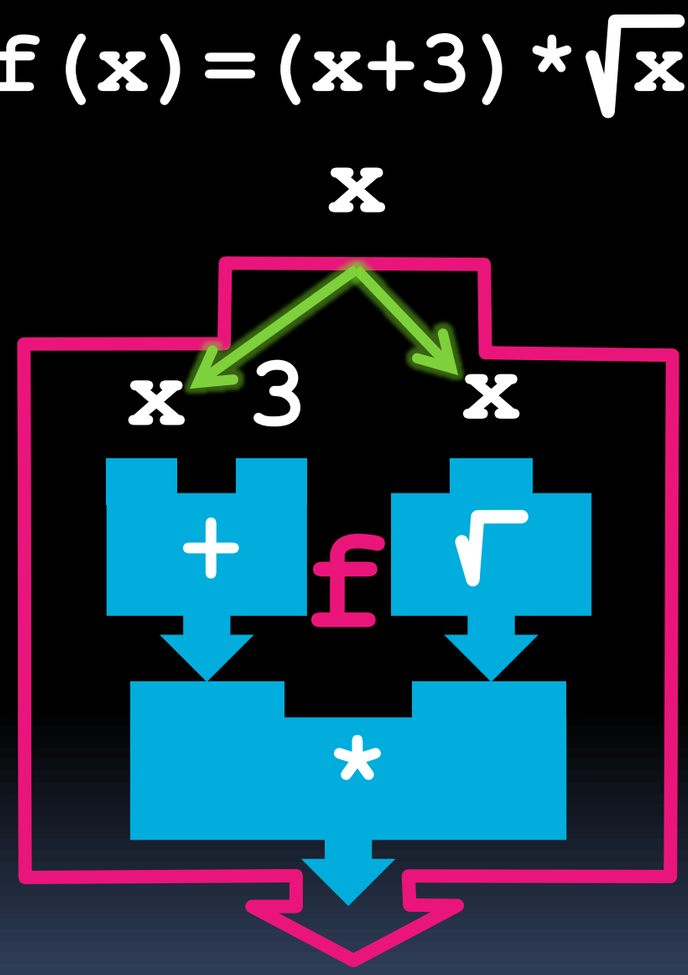
What are Programming Paradigms?

- “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 (assignation, evaluation, continuations, data flows, etc.).”
- Or, a way to classify the style of programming.

McKinsey

Functions Review

- Computation is the evaluation of functions
  - Plugging pipes together
  - Function: ≥ 0 inputs, 1 output
  - Functions can be input!
- Features
  - No state
    - E.g., variable assignments
  - No mutation
    - E.g., changing variable values
  - No side effects
    - E.g., nothing else happens
- Examples (though not pure)
  - Scheme, Scratch, BYOB, Snap!



McKinsey

Imperative Programming

- "Sequential" Programming
- Computation a series of steps
  - Assignment allowed
    - Setting variables
  - Mutation allowed
    - Changing variables
- Like following a recipe. E.g.,
- Procedure f(x)
  - ans = x
  - ans = $\sqrt{\text{ans}}$
  - ans = (x+3) * ans
  - return ans
- Examples: (though not pure)
  - Pascal, C

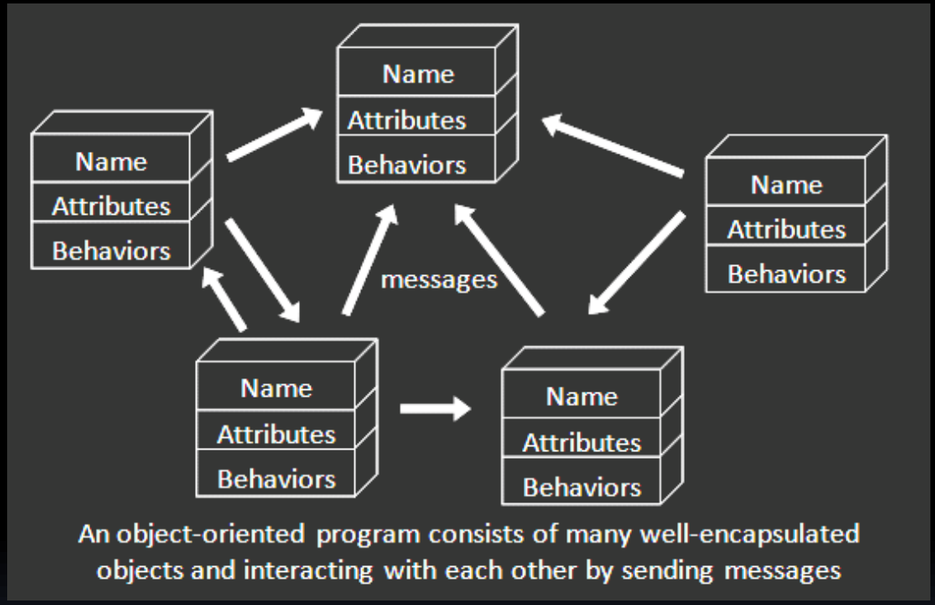
$$f(x) = (x+3) * \sqrt{x}$$

McKinsey

Object-Oriented Programming

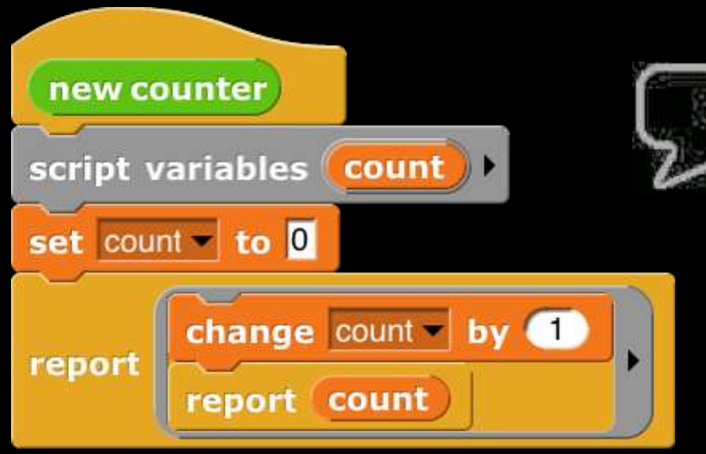
Object-Oriented Programming (OOP)

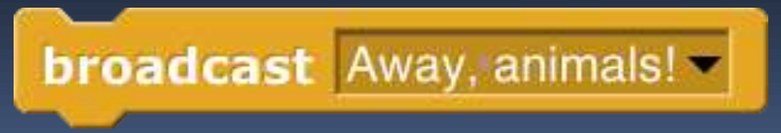
- Objects as data structures
  - With methods you ask of them
    - These are the behaviors
  - With local state, to remember
    - These are the attributes
- Classes & Instances
  - Instance an example of class
  - E.g., Fluffy is instance of Dog
- Inheritance saves code
  - Hierarchical classes
  -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



www3.ntu.edu.sg/home/ehchua/programming/java/images/OOP-Objects.gif

OOP in Snap!





McKinsey

Which of the following is true?

a) Objects can only delete other objects

b) Objects can only contain other objects

c) Objects can both delete and contain other objects

d) Objects can neither delete or contain other objects, they can only send messages to them.

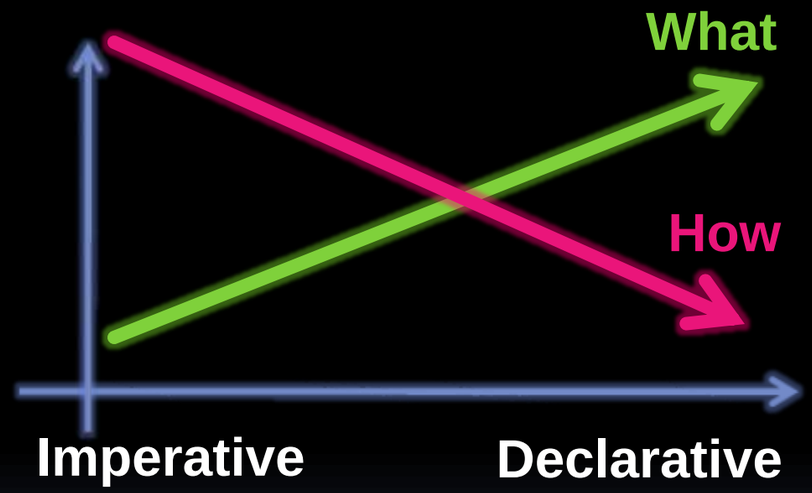
McKinsey

Declarative Programming

Declarative Programming

- Express what computation desired without specifying how it carries it out
  - Often a series of assertions and queries
  - Feels like magic!
- Sub-categories
  - Logic
  - Constraint
- Example: Prolog

Thatnks to Anders Hejlsberg
"The Future of C#" @ PDC2008
channel9.msdn.com/pdc2008/TL16/



McKinsey

Declarative Programming Example

- Five schoolgirls sat for an examination. Their parents – so they thought – showed an undue degree of interest in the result. They therefore agreed that, in writing home about the examination, each girl should make one true statement and one untrue one. The following are the relevant passages from their letters:

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

McKinsey

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, BYOB, Snap*!* too
- It’s like giving someone a juice drink (with many fruit in it) and asking to taste just one fruit!

McKinsey

Of 4 paradigms, which is the *most* powerful?

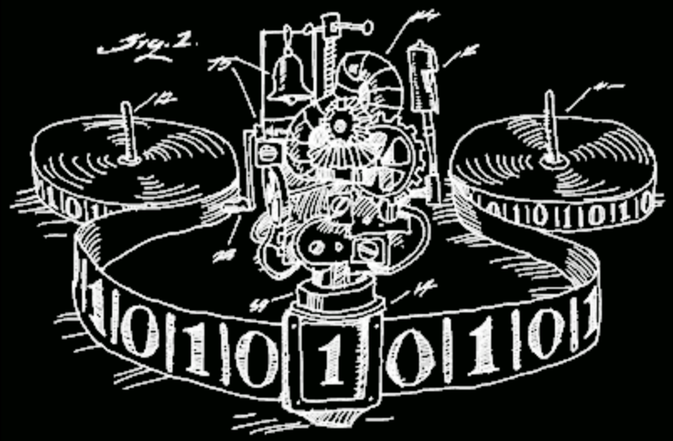
a) Functional
b) Imperative
c) OOP
d) Declarative
e) All equally powerful

McKinsey

Turing Completeness

en.wikipedia.org/wiki/Turing_completeness
ironphoenix.org/tril/tm/
aturingmachine.com/examples.php

- A Turing Machine has an infinite tape of 1s and 0s and instructions that say whether to move the tape left, right, read, or write it
  - Can simulate any one computer algorithm!
- 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
  - A way to decide that one programming language or paradigm is just as powerful as another



Turing Machine by Tom Dunne

Ways to Remember the Paradigms

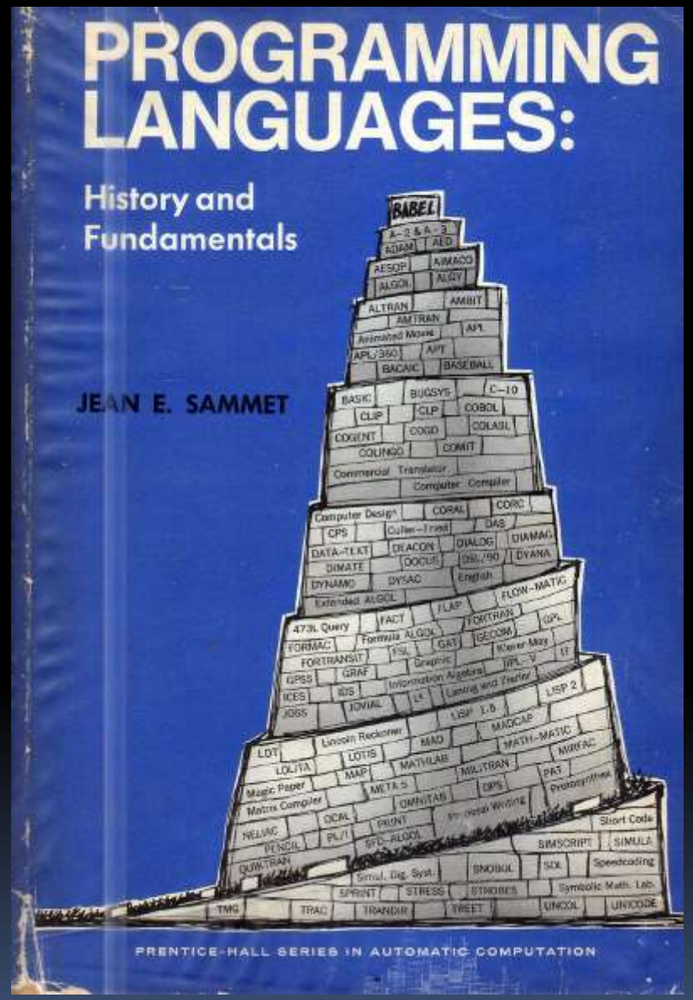
- Functional
  - Evaluate an expression and use the resulting value for something
- Object-oriented
  - Send messages between objects to simulate the temporal evolution of a set of real world phenomena
- Imperative
  - First *do this* and next *do that*
- Declarative
  - Answer a question via search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

www.cs.aau.dk/~normark/prog3-03/html/notes/paradigms_themes-paradigm-overview-section.html

McKinsey

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., Snap!
    - Can be functional
    - Can be imperative
    - Can be object-oriented
    - Can be declarative



McKinsey