



CS10 The Beauty and Joy of Computing

Lecture #5 : Programming Paradigms

2011-02-02

UC Berkeley EECS
Lecturer SOE
Dan Garcia

EGYPT CUTS OFF NET, CHINA CENSORS SEARCH WEB

Times like these make us all appreciate the value of freedom... China has blocked searches for "Egypt", and has tried to frame the demonstrations as "a chaotic affair that embodies the pitfalls of trying to plant democracy in countries that are not quite ready for it". Xiao Qiang (prof here) is local expert...

www.nytimes.com/2011/02/01/world/asia/01beijing.html



UC Berkeley CS10 "The Beauty and Joy of Computing" : Programming Paradigms (3)



Programming Paradigms Lecture

- What are they?
 - Most are Hybrids!
- The Four Primary ones
 - Functional
 - Imperative
 - Object-Oriented
 - OOP Example: Sketchpad
 - Declarative
- Turing Completeness
- Summary



UC Berkeley CS10 "The Beauty and Joy of Computing" : Programming Paradigms (2)



en.wikipedia.org/wiki/Programming_paradigm

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.



UC Berkeley CS10 "The Beauty and Joy of Computing" : Programming Paradigms (3)



byob.berkeley.edu

Of 4 paradigms, how many can BYOB be?



- a) 1 (functional)
- b) 1 (not functional)
- c) 2
- d) 3
- e) 4



UC Berkeley CS10 "The Beauty and Joy of Computing" : Programming Paradigms (4)



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!



UC Berkeley CS10 "The Beauty and Joy of Computing" : Programming Paradigms (5)

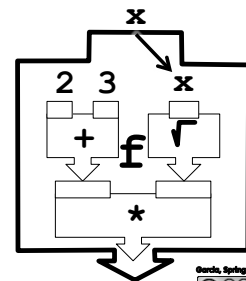


en.wikipedia.org/wiki/Functional_programming

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
 - E.g., variable assignments
 - No mutation
 - E.g., changing variable values
 - No side effects
- Examples (tho not pure)
 - Scheme, Scratch BYOB

$$f(x) = (2+3) * \sqrt{x}$$




UC Berkeley CS10 "The Beauty and Joy of Computing" : Programming Paradigms (6)



en.wikipedia.org/wiki/Imperative_programming

Imperative Programming

- **"Sequential" Programming** $f(x) = (2+3) * \sqrt{x}$
- **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{ans}
 - ans = (2+3) * ans
 - return ans
- **Examples: (tho not pure)**
 - Pascal, C




UC Berkeley CS10 "The Beauty and Joy of Computing": Programming Paradigms (7)

en.wikipedia.org/wiki/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., pianist special case of musician, a special case of performer
- **Examples (tho not pure)**
 - Java, C++




An object-oriented program consists of many well-encapsulated objects and interacting with each other by sending messages
www3.rtu.edu.sg/home/ehchua/programming/java/images/OOP-Objects.gif

UC Berkeley CS10 "The Beauty and Joy of Computing": Programming Paradigms (8)

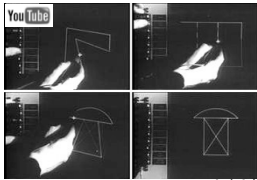
en.wikipedia.org/wiki/Sketchpad

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

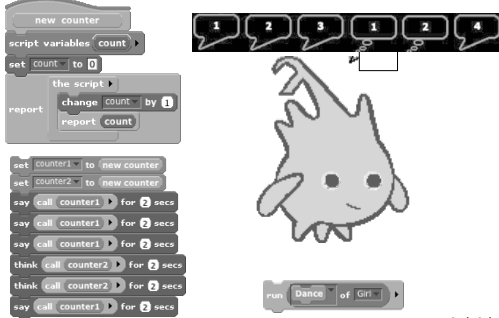


Spent the past few years doing research @ Berkeley in EECS dept!



UC Berkeley CS10 "The Beauty and Joy of Computing": Programming Paradigms (9)

OOP in BYOB




UC Berkeley CS10 "The Beauty and Joy of Computing": Programming Paradigms (10)

en.wikipedia.org/wiki/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
 - We saw in Sketchpad!
- **Example: Prolog**




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

UC Berkeley CS10 "The Beauty and Joy of Computing": Programming Paradigms (11)

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



UC Berkeley CS10 "The Beauty and Joy of Computing": Programming Paradigms (12)

Of 4 paradigms, what's the most powerful?



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



UC Berkeley CS10 "The Beauty and Joy of Computing": Programming Paradigms (18)



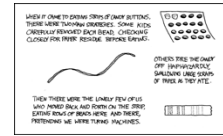
Turing Completeness

en.wikipedia.org/wiki/Turing_completeness
ironphoenix.org/tril/tm/

- 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 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



Xkcd comic "Candy Button Paper"



UC Berkeley CS10 "The Beauty and Joy of Computing": Programming Paradigms (14)



Ways to Remember the Paradigms

en.wikipedia.org/wiki/Programming_paradigm

- | | |
|--|--|
| <ul style="list-style-type: none"> ▪ Functional <ul style="list-style-type: none"> ▫ Evaluate an expression and use the resulting value for something ▪ Imperative <ul style="list-style-type: none"> ▫ First <i>do this</i> and next <i>do that</i> | <ul style="list-style-type: none"> ▪ Object-oriented <ul style="list-style-type: none"> ▫ Send messages between objects to simulate the temporal evolution of a set of real world phenomena ▪ Declarative <ul style="list-style-type: none"> ▫ Answer a question via search for a solution |
|--|--|

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

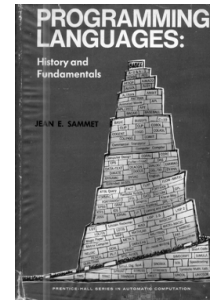


UC Berkeley CS10 "The Beauty and Joy of Computing": Programming Paradigms (18)



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



UC Berkeley CS10 "The Beauty and Joy of Computing": Programming Paradigms (14)

