TURING TURNS 100

If you visited google.com on Saturday, you saw a tribute to this founding father of computer science who broke the German Enigma code during WW2.

[Link to Wikipedia page on Alan Turing] en.wikipedia.org/wiki/Alan_Turing
What paradigm is that language?
- Most are hybrids!

Four Primary Paradigms
- Functional
- Imperative
- Object-Oriented
  - OOP Example: Sketchpad
- Declarative

Turing Completeness

Summary
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.
Of 4 paradigms, how many can BYOB be?

a) 1 (functional)
b) 1 (not functional)
c) 2
d) 3
e) 4
Most Languages Are Hybrids

- This makes it hard to teach paradigms, because most languages can express several
  - Called “Multi-paradigm” languages
  - Scratch & BYOB too!
- It’s like giving someone a juice drink (with many fruits 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
    - E.g., variable assignments
  - No mutation
    - E.g., changing variable values
  - No side effects

- **Examples (not all pure)**
  - Scheme, Scratch, BYOB

\[ f(x) = (x + 3) \cdot \sqrt{x} \]
**Imperative Programming**

- **“Sequential” Programming**
- Computation a series of steps
  - Assignment allowed
    - Setting variables
  - Mutation allowed
    - Changing variables
- Like writing a recipe
  - Procedure \( f(x) \):
    - \( \text{ans} = x \)
    - \( \text{ans} = \sqrt{\text{ans}} \)
    - \( \text{ans} = (x+3) \cdot \text{ans} \)
    - return \( \text{ans} \)
- Examples (not all pure)
  - Pascal, C
Object-Oriented Programming (OOP)

- **Objects** are data structures
  - With **methods** you ask of them
    - These are the behaviors
  - With **local state**, to store info
    - 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., singer is a special case of musician, musician is a special case of person

- **Examples (not all pure)**
  - Java, C++

---


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

Chun, Summer 2012
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

- It was the first:
  - Object-oriented system
  - Graphical user interface
  - non-procedural language

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

new counter

script variables count

set count to 0

report

count

the script

change count by 1

change count by 1

change count by 1

change count by 1

change count by 1

change count by 1

change count by 1

change count by 1

change count by 1

change count by 1

change count by 1

Dance of Girl

run

UC Berkeley CS10 “The Beauty and Joy of Computing” : Programming Paradigms (10)
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/
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 2\textsuperscript{nd}
  - I was 3\textsuperscript{rd}
- Ethel
  - I was on top
  - Joan was 2\textsuperscript{nd}
- Joan
  - I was 3\textsuperscript{rd}
  - Ethel was last
- Kitty
  - I came out 2\textsuperscript{nd}
  - Mary was only 4\textsuperscript{th}
- Mary
  - I was 4\textsuperscript{th}
  - Betty was 1\textsuperscript{st}
Of 4 paradigms, what’s the most powerful?

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

- 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”
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