Most Languages Are Hybrids!

- This makes it hard to teach to students, because most languages have facets of several paradigms!
  - Called “Multitasking” 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
    - E.g., variable assignments
  - No mutation
    - E.g., changing variable values
  - No side effects
- Examples
  - Scheme, Scratch BYOB

Imperative Programming

- AKA “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 = ans + 3
    - ans = (2+3) * ans
    - return ans
- Examples: 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: Java, C++
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

---

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

---

Dr. Ivan Sutherland

- "Father of Computer Graphics"
- 1988 Turing Award ("Nobel prize" for CSI)
- Wrote Sketchpad for his foundational 1963 thesis
- The most impressive software ever written

First…

- Object-oriented system
- Graphical user interface
- non-procedural language

---

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.
- 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.

---

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