CS316 9-01-09

- No required textbook
  - Suggested textbook: Concepts of Programming Languages, 8th ed
    by Robert W. Sebesta & Addison Wesley

- Anatomy of Programming Languages
  - Common principles about how programming languages work
- Modern Programming Language Paradigms
  - Raw machine language
    - Binary code
  - Assembly language
    - Assigned easy-to-remember names to binary instructions
      - Example: “Store,” “Load”
  - High-Level Languages
    - Procedural Languages
      - Assignment statement (“ x=E; ”)
        - Right hand side of the assignment can use any algebraic/mathematical expression
      - Conditional Branches
        - Example: if(B) S {“if some boolean condition, B, is satisfied, execute S”}
          if(B) S1 else S2 {“if B is satisfied, execute S1, otherwise execute S2”}
      - Loop Constructs
        - Example: while(B) S
          do S while(B)
          for loops
      - Function Calls
        - Encapsulate and parameterize common computational problems
      - High-Level Data Structures
        - Examples: integer, float, boolean, character, string
      - Languages:
        - Fortran
        - ALGOL60
        - C
        - Pascal
        - Modula
        - ADA, etc.
    - Object-Oriented Languages
      - Built on the powerful features introduced in procedural languages
      - Introduced the concepts of class and inheritance
      - Changed the focus of programming process to:
        - abstract data types
        - sets of data objects
        - operations (method functions)
        - inheritance hierarchies
        - inheritance polymorphism – redefinition and dynamic binding of function body

code
        - single inheritance (ex - Java)
            - hierarchy = tree, has the root class as the root
        - multiple inheritance (ex – C++)
            - hierarchy = general lattice, has root class as root and branches can connect with other branches
    - Example of Inheritance:
        - class pocketDevice – turnOn(), turnOff() {supply different body code for turnOn(), turnOff() to be suitable for specific subclasses of devices}
            - class cellPhone
            - class audioPlayer
            - class camera
    - Dynamic Binding – Powerful feature for extensibility of class hierarchies
        - pocketDevice x;
          x.turnOn();
          x.turnOff();
  - The price to be paid compared to procedural languages:
    - Execution speed is somewhat slower due to dynamic binding of function body code and heavy use of object references
- Functional Languages
  - “Functions are all we need”
  - Programs are just sets of pure function definitions
  - No assignment statements
  - No loop constructs
    - Iteration is done by recursive functions
  - Example:

    -
      ```
      int f(x)
      {
              if(x==3) return g(h(x-1);
              else return i(k(x+2));
      }
      ```
  - Pros of functional languages: (very high-level languages)
    - Programming process is easier and faster
    - Reasoning and verification of programs are easier
  - Cons of functional languages:
    - Execution speed is fundamentally much slower than procedural/object-oriented languages
    - Bunch of function calls (including recursive function calls) are expensive (timewise) compared to assignments/loops
- Logic Languages: (very high-level languages)
  - Programs are sets of logical statements
  - Computation (program execution) is logical deduction process
  - Example language: Prolog (used for artificial intelligence and symbolic computation applications)
  - No assignment statements
  - No loop constructs

    - Iteration is done by recursive functions
- Same pros and cons as functional languages