# 8/24/17

* Programming Language: Notational system for expressing computation in both human readable and machine readable format.
* Control abstraction
  + If stmts, loops, subroutines, libraries
* Data abstraction;
  + Double y = 3.47821;
  + Int x = 34;
  + Structs/records
  + arrays
* Object oriented programming (O.O.)
  + Combines methods and structures.
  + Encapsulation
  + Private
  + Grouping of data and methods
  + Abstraction
* Soft real time: streaming audio/video,
  + It can “fail a little”
* Hard real time
  + It **cannot** fail
* Readability criteria
  + Orthogonality
  + Support for abstraction
  + Expressivity

# 9/5/17

* Evaluation of languages for figuring out which language to use.
  + Cost
  + Speed of execution
  + Ease to learn
* Computer Architecture influence
  + Well-known computer architect: Von Neumann
  + Paradigms:
    - Imperative:
      * Sequence of steps
      * Algorithm
      * Variables to hold state
      * Loops
  + 50s: Worried about computer efficiency
  + late 1960s People efficiency becomes important.
* Logic:
  + A program is a set of facts and rules

## 9/7/17

* ALGOL is the first true high level language
  + Introduced:
    - Blocks
    - Parameters
    - Recursion
  + Problem is it didn’t have a compiler
  + Was never widely used
    - Because hard to compile
    - And lack of support from IBM
    - everyone used Fortran already
* COBOL
  + Was created to manipulate data much better
* Basic
  + Was created to make it easier for programmers and easier to learn for non-science students.

# 9/14/17

* Lexical: find tokens/terminals
  + Ex: if while
  + Identifiers
  + Literal values
  + Symbols & operators { } > = etc.
  + Regular expressions find these
* Syntax: Tokens in a valid order
  + Specified with a “Grammar”
* CFG
  + Context free grammar
    - Things that look the same no matter where they are used.
  + BNF: Bakus-Naur Form is a way of writing a CFG.
* Program -> begin statements end.

**PGM**

Begin stmts end. By(1)

Begin stmt stmts end by ()

* BNF Grammar:
  1. Set of terminals (token)
     + {begin end. = ; output xyz 0…9}
  2. set of non-terminals
     + {pgm stmts stmt assignment output ident value}
  3. set rules/productions of form
     + nonterm 🡪 any combination of non-terminals & terminals.
  4. Start symbol: non termento.
     + Start derivation or parse tree by default: 1st one = pgm