Notes 1-17

Language Evaluation Criteria:

Get 1-4 from someone

5. Syntax Design (rules, how we write)

- Rules for forming identifier names

- Use of reserve words (can’t redefine what it means) vs. keywords (can redefine)

- ‘if’ in C++ is a reserve word

- ‘if’ and ‘while’ in Fortran are keywords

TEST QUESTION!!!!!!!!

6. Reliability

- Characteristic that a program written in that language will perform to specification under all conditions

- Type checking: Correct types are supplied to operations

- Ex: Ada- good type checking

- Exception handing: exception- rare event that must be handled appropriately (division by 0)

7.Orthogonality

- Relatively small number of language primitives / constructs which can be formed into larger constructs

- Ex: ALGOL

8. Cost

- Low cost (memory, time, to learn the language)

- Easy language examples: BASIC, Pascal, Python,

- Hard language examples: Ada, ALGOL,

- Cost of reliability

- Cost to maintain

9. Portability

- easy to transfer the program from one platform to another w/o major modification

10. Support for abstraction:

- Is it a good fit for the problem

10. Well-defined-ness:

- syntax (how we write)

- semantics (meaning)

For first exam, there will be a lot of this vocab, maybe 3 of the language evaluation criteria

Language Categories:

One way to look at languages:

* Imperative languages: Sequence of commands: C
* Object oriented language: C++
* Functional language: depends on applying functions
* Logic-oriented language: modeled after how we reason

Another way to look at languages:

* General purpose: Can be used across domains (Python, C, C++, Pearl, Ruby)
* Special purpose: Simula for simulations, Verilog for hardware, MATLAB

Influences on Language Design:

Computer Architecture:

* Affects language design
* Von Neuman Architecture- data and program s are stored in the same memory

Implementation Methods:

* Compilation: programs are translated into machine code which can be directly executed on computer
  + Slow translation
  + Fast execution because once you have compiled that program you don’t have to keep recompiling
* Pure interpretation- read the high-level language, works line-by-line, interpreter is present during program run (translate to intermediate language)
  + Slow execution
* Hybrid implementation: translate a high level language , to intermediate language for a non-existent virtual CPU: Just in time (JIT), Java(JVM)