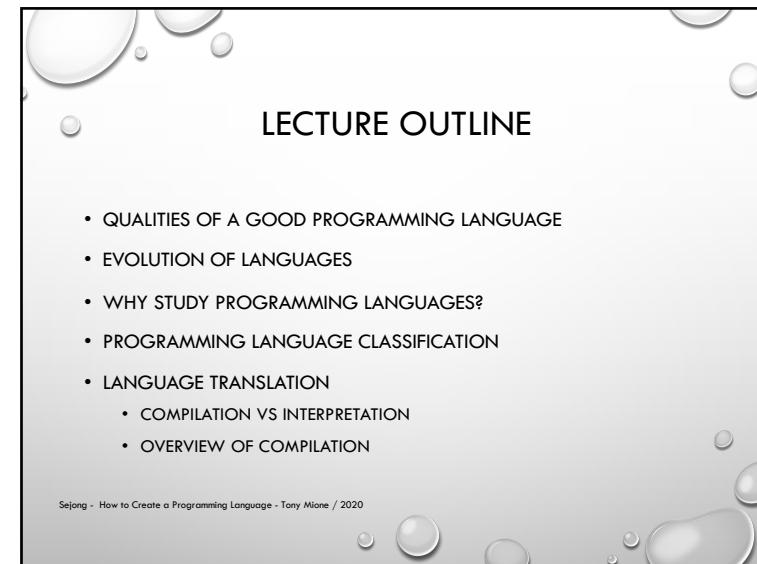


How to Create a Programming Language

LECTURE 1: INTRODUCTION TO PROGRAMMING LANGUAGES

1

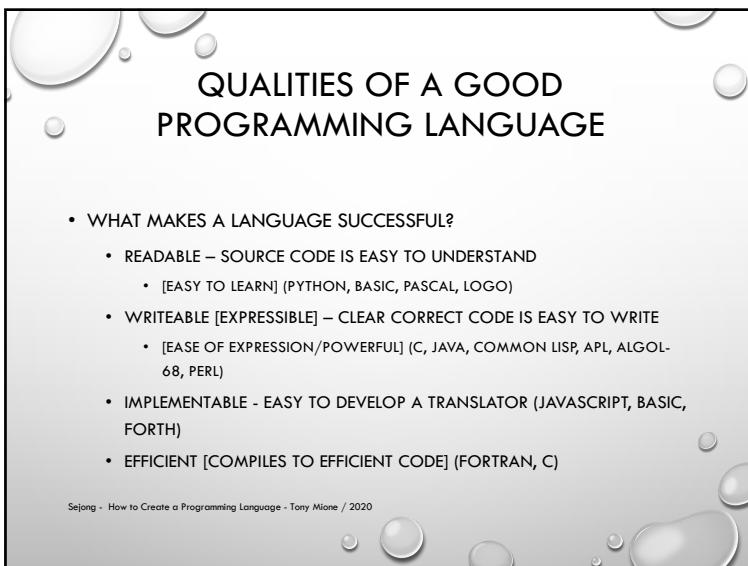


LECTURE OUTLINE

- QUALITIES OF A GOOD PROGRAMMING LANGUAGE
- EVOLUTION OF LANGUAGES
- WHY STUDY PROGRAMMING LANGUAGES?
- PROGRAMMING LANGUAGE CLASSIFICATION
- LANGUAGE TRANSLATION
 - COMPILE VS INTERPRETATION
 - OVERVIEW OF COMPILE

Sejong - How to Create a Programming Language - Tony Mione / 2020

2

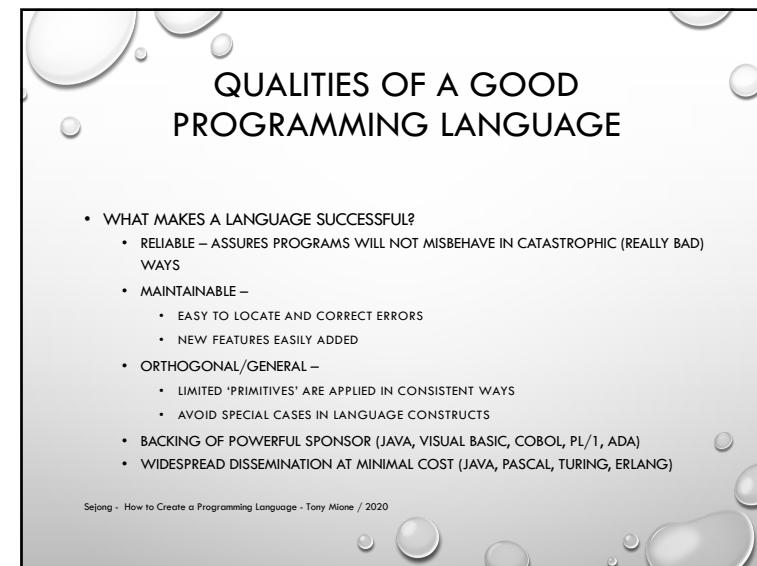


QUALITIES OF A GOOD PROGRAMMING LANGUAGE

- WHAT MAKES A LANGUAGE SUCCESSFUL?
 - READABLE – SOURCE CODE IS EASY TO UNDERSTAND
 - [EASY TO LEARN] (PYTHON, BASIC, PASCAL, LOGO)
 - WRITEABLE [EXPRESSIBLE] – CLEAR CORRECT CODE IS EASY TO WRITE
 - [EASE OF EXPRESSION/POWERFUL] (C, JAVA, COMMON LISP, APL, ALGOL-68, PERL)
 - IMPLEMENTABLE - EASY TO DEVELOP A TRANSLATOR (JAVASCRIPT, BASIC, FORTH)
 - EFFICIENT [COMPILES TO EFFICIENT CODE] (FORTRAN, C)

Sejong - How to Create a Programming Language - Tony Mione / 2020

3



QUALITIES OF A GOOD PROGRAMMING LANGUAGE

- WHAT MAKES A LANGUAGE SUCCESSFUL?
 - RELIABLE – ASSURES PROGRAMS WILL NOT MISBEHAVE IN CATASTROPHIC (REALLY BAD) WAYS
 - MAINTAINABLE –
 - EASY TO LOCATE AND CORRECT ERRORS
 - NEW FEATURES EASILY ADDED
 - ORTHOGONAL/GENERAL –
 - LIMITED ‘PRIMITIVES’ ARE APPLIED IN CONSISTENT WAYS
 - AVOID SPECIAL CASES IN LANGUAGE CONSTRUCTS
 - BACKING OF POWERFUL SPONSOR (JAVA, VISUAL BASIC, COBOL, PL/I, ADA)
 - WIDESPREAD DISSEMINATION AT MINIMAL COST (JAVA, PASCAL, TURING, ERLANG)

Sejong - How to Create a Programming Language - Tony Mione / 2020

4

INTRODUCTION

- WHY DO WE HAVE PROGRAMMING LANGUAGES? WHAT IS A LANGUAGE FOR?
 - **WAY OF THINKING** – WAY TO EXPRESS ALGORITHMS
 - LANGUAGES FROM THE USER'S POINT OF VIEW
 - **ABSTRACTION OF VIRTUAL MACHINE** – WAY TO SPECIFY WHAT YOU WANT HARDWARE TO DO WITHOUT GETTING INTO THE BITS
 - LANGUAGES FROM THE IMPLEMENTOR'S POINT OF VIEW

Sejong - How to Create a Programming Language - Tony Mione / 2020

5

EVOLUTION OF LANGUAGES

- EARLY COMPUTERS PROGRAMMED DIRECTLY WITH MACHINE CODE
 - PROGRAMMER HAND WROTE BINARY CODES
 - PROGRAM ENTRY DONE WITH TOGGLE SWITCHES
 - SLOW. VERY ERROR-PRONE
- WATCH HOW TO PROGRAM A PDP-8!
 - [HTTPS://WWW.YOUTUBE.COM/WATCH?V=DPIOENTAHUY](https://www.youtube.com/watch?v=DPIOENTAHUY)

Sejong - How to Create a Programming Language - Tony Mione / 2020

6

EVOLUTION OF LANGUAGES

- ASSEMBLY LANGUAGE ADDED MNEMONICS
 - ONE-TO-ONE CORRESPONDENCE WITH MACHINE INSTRUCTIONS
 - DATA REPRESENTED WITH SYMBOLS (NAMES)
 - 'ASSEMBLER' PROGRAM TRANSLATED SYMBOLIC CODE TO MACHINE CODE

Sejong - How to Create a Programming Language - Tony Mione / 2020

7

EVOLUTION OF LANGUAGES

- EXAMPLE INTEL X86 ASSEMBLER:


```
pushl %ebp
movl %esp, %ebp
pushl %ebx
subl $4, %esp
andl $-16, %esp
call getint
movl %eax, %ebx
call getint
cmpl %eax, %ebx
je C
A: cmpl %eax, %ebx
...
```

Sejong - How to Create a Programming Language - Tony Mione / 2020

8

EVOLUTION OF LANGUAGES

- 'MACROS' ADDED TO ASSEMBLERS
 - PARAMETERIZED TEXT EXPANSION
 - PROGRAMMERS PUT COMMON INSTRUCTION SEQUENCES INTO MACRO DEFINITIONS
- EASIER. STILL ERROR-PRONE

Sejong - How to Create a Programming Language - Tony Mione / 2020

9

EVOLUTION OF LANGUAGES

- HIGH-LEVEL LANGUAGES
 - SYNTAX FOR SELECTION (IF/THEN) AND ITERATION (LOOPS)
 - ONE-TO-ONE CORRESPONDENCE IS GONE
- EARLIEST 'HIGH-LEVEL' LANGUAGES – 1958/60
 - FORTRAN I
 - ALGOL-58, ALGOL-60
- TRANSLATORS ARE NOW 'COMPILERS'
 - MORE COMPLEX THAN ASSEMBLERS

Sejong - How to Create a Programming Language - Tony Mione / 2020

10

WHY STUDY PROGRAMMING LANGUAGES?

- HELPS CHOOSE A LANGUAGE:
 - C VS. C++ FOR SYSTEMS PROGRAMMING
 - MATLAB VS. PYTHON VS. R FOR NUMERICAL COMPUTATIONS
 - JAVA VS. JAVASCRIPT FOR WEB APPLICATIONS
 - PYTHON VS. RUBY VS. COMMON LISP VS. SCHEME VS. ML FOR SYMBOLIC DATA MANIPULATION
 - JAVA RPC (JAX-RPC) VS. C/CORBA FOR NETWORKED PC PROGRAMS

Sejong - How to Create a Programming Language - Tony Mione / 2020

11

WHY STUDY PROGRAMMING LANGUAGES?

- MAKE IT EASIER TO LEARN NEW LANGUAGES
 - SOME LANGUAGES SIMILAR – RELATED ON A 'FAMILY TREE' OF LANGUAGES →
 - [HTTP://RIGAUX.ORG/LANGUAGE-STUDY/DIAGRAM.PNG](http://RIGAUX.ORG/LANGUAGE-STUDY/DIAGRAM.PNG)
 - CONCEPTS HAVE MORE SIMILARITY
 - THINKING IN TERMS OF SELECTION, ITERATION, RECURSION
 - UNDERSTANDING ABSTRACTION HELPS EASE ASSIMILATION OF SYNTAX AND SEMANTICS
 - ANALOGY TO HUMAN LANGUAGES: GOOD GRASP OF GRAMMAR [SOMETIMES] MAKES IT EASIER TO PICK UP NEW LANGUAGES

Sejong - How to Create a Programming Language - Tony Mione / 2020

12

WHY STUDY PROGRAMMING LANGUAGES?

- HELPS MAKE BETTER USE OF WHATEVER LANGUAGE IS BEING USED:
 - UNDERSTAND TRADE-OFFS/IMPLEMENTATION COSTS BASED ON UNDERSTANDING OF LANGUAGE INTERNALS
- EXAMPLES:
 - USE $X*X$ RATHER THAN X^{**2}
 - USE C POINTERS OR PASCAL 'WITH' STATEMENT TO FACTOR ADDRESS CALCULATIONS
 - AVOID CALL-BY-VALUE WITH LARGE ARGUMENTS IN PASCAL
 - AVOID THE USE OF CALL-BY-NAME IN ALGOL-60
 - CHOOSE BETWEEN COMPUTATION AND TABLE LOOKUP

Sejong - How to Create a Programming Language - Tony Mione / 2020

13

WHY STUDY PROGRAMMING LANGUAGES?

- LEARN HOW TO DO THINGS NOT SUPPORTED BY LANGUAGE
 - LACK OF SUITABLE CONTROL STRUCTURES IN FORTRAN
 - USE COMMENTS AND PROGRAMMER DISCIPLINE FOR CONTROL STRUCTURES
 - LACK OF RECURSION IN FORTRAN
 - WRITE A RECURSIVE ALGORITHM USING MECHANICAL RECURSION ELIMINATION
 - LACK OF NAMED CONSTANTS AND ENUMERATIONS IN FORTRAN
 - USE VARIABLES THAT ARE INITIALIZED ONCE AND NEVER CHANGED
 - LACK OF MODULES IN C AND PASCAL
 - USE COMMENTS AND PROGRAMMER DISCIPLINE

Sejong - How to Create a Programming Language - Tony Mione / 2020

14

PROGRAMMING LANGUAGE CLASSIFICATION

- **IMPERATIVE** – FOCUS: **HOW** THE COMPUTER SHOULD DO A TASK
- **DECLARATIVE** – FOCUS: **WHAT** THE COMPUTER SHOULD DO

Sejong - How to Create a Programming Language - Tony Mione / 2020

15

IMPERATIVE PROGRAMMING LANGUAGES

- **VON NEUMANN** – BASED ON MODIFICATION OF VARIABLES/STATE VIA SIDE-EFFECTS
 - C
 - FORTRAN
 - ADA
 - PASCAL
 - ETC.
- **OBJECT-ORIENTED** – BASED ON SEPARATION OF DATA AND CODE INTO SEMI-INDEPENDENT 'OBJECTS'
 - SMALLTALK
 - C++
 - JAVA
 - ETC.

Sejong - How to Create a Programming Language - Tony Mione / 2020

16

DECLARATIVE PROGRAMMING LANGUAGES

- **FUNCTIONAL** – BASED ON (POSSIBLY RECURSIVE) FUNCTIONS
 - LISP
 - ML
 - HASKELL
- **DATAFLOW** – BASED ON A ‘FLOW’ OF ‘TOKENS’ TO PROCESSING ‘NODES’
 - ID
 - VAL
- **LOGIC/CONSTRAINT-BASED** – BASED ON FINDING VALUES THAT FIT A CRITERIA (GOAL-DIRECTED SEARCH) PRINCIPLES INCLUDE PREDICATE LOGIC.
 - PROLOG

Sejong - How to Create a Programming Language - Tony Mione / 2020

17

OTHER CLASSIFICATIONS

- **MARKUP**
 - SORT OF A LANGUAGE TYPE HOWEVER THESE LACK ‘EXECUTION SEMANTICS’
- **ASSEMBLERS**

Sejong - How to Create a Programming Language - Tony Mione / 2020

18

LAB

- 30 MINS, IN TEAMS OF 2 STUDENTS
- RESEARCH (ONLINE) TWO LANGUAGES FROM **DIFFERENT** CLASSIFICATIONS
- NOTE THE DIFFERENCES
- JOT SOME IDEAS DOWN ABOUT HOW THE CLASS OF LANGUAGE HELPS ITS EFFECTIVENESS FOR SPECIFIC PROBLEM DOMAINS
 - FINANCIAL APPLICATIONS
 - EMBEDDED AVIONICS
 - FORMAL METHODS [MATHEMATICAL PROOFS OF CORRECTNESS]
 - COMPILERS FOR TRANSLATING LANGUAGES TO MACHINE CODE
 - ETC.

Sejong - How to Create a Programming Language - Tony Mione / 2020

19

QUESTIONS

Sejong - How to Create a Programming Language - Tony Mione / 2020

20