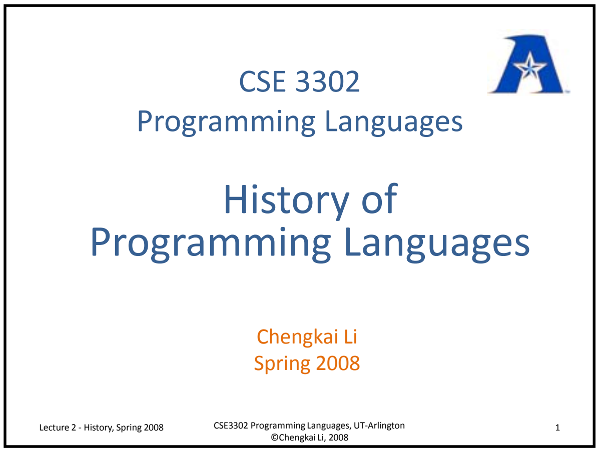
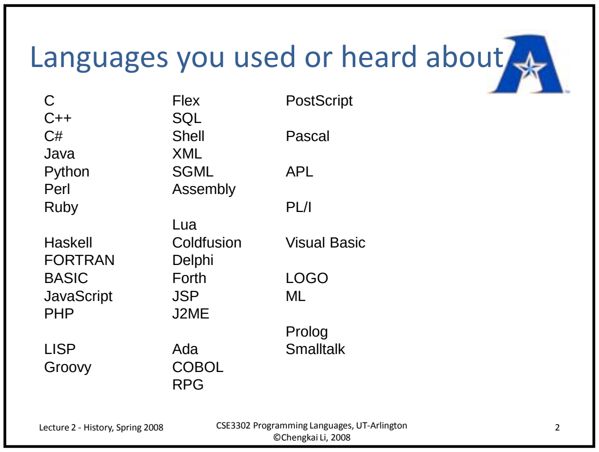


CSE 3302
Programming Languages

History of Programming Languages

Chengkai Li
Spring 2008



Languages you used or heard about

C
C++
C#
Java
Python
Perl
Ruby

Haskell
FORTRAN
BASIC
JavaScript
PHP

LISP
Groovy

Flex
SQL
Shell
XML
SGML
Assembly

Lua
Coldfusion
Delphi
Forth
JSP
J2ME

Ada
COBOL
RPG

PostScript

Pascal

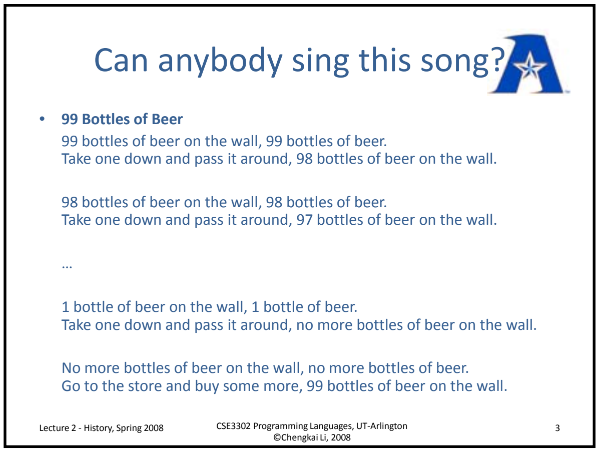
APL

PL/I

Visual Basic

LOGO
ML

Prolog
Smalltalk



Can anybody sing this song?

- **99 Bottles of Beer**

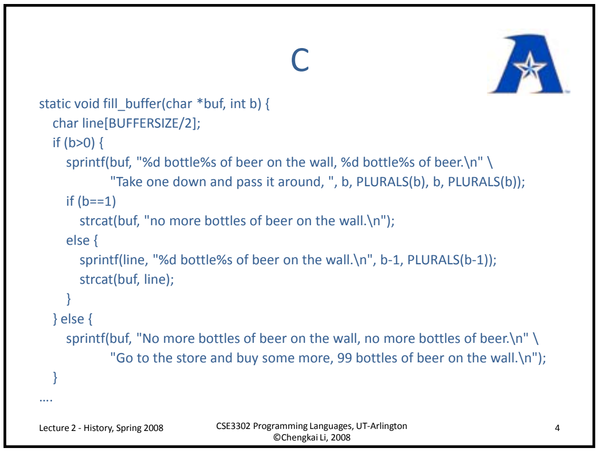
  99 bottles of beer on the wall, 99 bottles of beer.
  Take one down and pass it around, 98 bottles of beer on the wall.

  98 bottles of beer on the wall, 98 bottles of beer.
  Take one down and pass it around, 97 bottles of beer on the wall.

  …

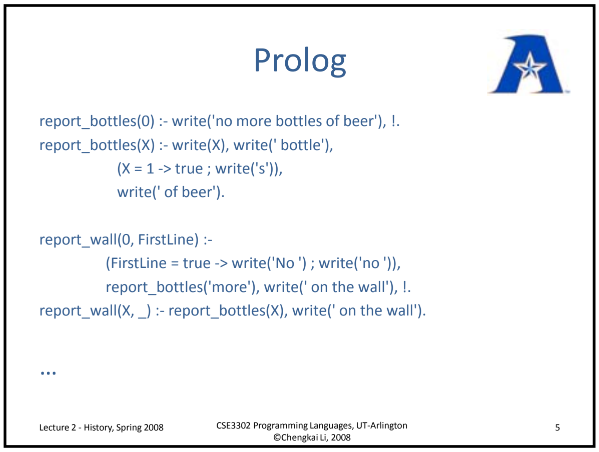
  1 bottle of beer on the wall, 1 bottle of beer.
  Take one down and pass it around, no more bottles of beer on the wall.

  No more bottles of beer on the wall, no more bottles of beer.
  Go to the store and buy some more, 99 bottles of beer on the wall.



C

```
static void fill_buffer(char *buf, int b) {
  char line[BUFFERSIZE/2];
  if (b>0) {
    sprintf(buf, "%d bottle%s of beer on the wall, %d bottle%s of beer.\n" \
            "Take one down and pass it around, ", b, PLURALS(b), b, PLURALS(b));
    if (b==1)
      strcat(buf, "no more bottles of beer on the wall.\n");
    else {
      sprintf(line, "%d bottle%s of beer on the wall.\n", b-1, PLURALS(b-1));
      strcat(buf, line);
    }
  } else {
    sprintf(buf, "No more bottles of beer on the wall, no more bottles of beer.\n" \
            "Go to the store and buy some more, 99 bottles of beer on the wall.\n");
  }
....
```

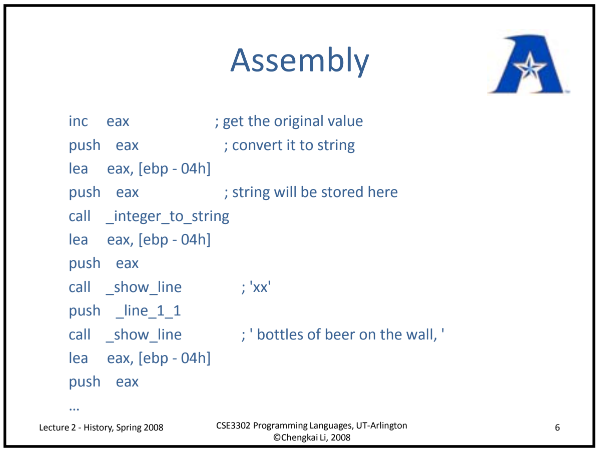


Prolog

```
report_bottles(0) :- write('no more bottles of beer'), !.
report_bottles(X) :- write(X), write(' bottle'),
          (X = 1 -> true ; write('s')),
          write(' of beer').

report_wall(0, FirstLine) :-
         (FirstLine = true -> write('No ') ; write('no ')),
         report_bottles('more'), write(' on the wall'), !.
report_wall(X, _) :- report_bottles(X), write(' on the wall').


…
```



Assembly

```
inc    eax                 ; get the original value
push   eax                 ; convert it to string
lea    eax, [ebp - 04h]
push   eax                 ; string will be stored here
call   _integer_to_string
lea    eax, [ebp - 04h]
push   eax
call   _show_line          ; 'xx'
push   _line_1_1
call   _show_line          ; ' bottles of beer on the wall, '
lea    eax, [ebp - 04h]
push   eax
…
```

?

```
select
   CASE (a.aa * 10 + b.bb)
      WHEN 0 THEN 'No more bottle of beer on the wall, no more bottles of beer. ' +
              'Go to the store and buy some more, 99 bottles of beer on the wall.'
      WHEN 1 THEN '1 bottle of beer on the wall, 1 bottle of beer. ' +
               'Take one down and pass it around, no more bottles of beer on the wall.'
      WHEN 2 THEN '2 bottles of beer on the wall, 2 bottles of beer. ' +
               'Take one down and pass it around, 1 bottle of beer on the wall.'
      ELSE
         cast((a.aa * 10 + b.bb) as varchar(2)) + ' bottles of beer on the wall, ' +
         cast((a.aa * 10 + b.bb) as varchar(2)) + ' bottles of beer. ' +
         'Take one down and pass it around, ' +
         cast((a.aa * 10 + b.bb)-1 as varchar(2)) + ' bottles of beer on the wall.'
   END
from
   (select 0 as aa union select 1 union select 2 union select 3 union select 4
   union select 5 union select 6 union select 7 union select 8 union select 9) a
cross join
   (select 0 as bb union select 1 union select 2 union select 3 union select 4
   union select 5 union select 6 union select 7 union select 8 union select 9) b
order by a.aa desc, b.bb desc
```

?

```
bottles = [98:-1:3]; % bottles 98 to 3 (99, 2 & 1 are treated as special case)
lines = 3; % need the number of bottles at the beginning of 3 lines

num_array = ones(lines,1) * bottles; % bottles is a (1x96) array

format_plural1 = '%d bottles of beer on the wall,\n%d bottles of beer,\n';
format_plural2 = 'Take one down, pass it around,\n%d bottles of beer on the
    wall.\n\n';
format_sing1 = '%d bottle of beer on the wall,\n%d bottle of beer,\n';
format_sing2 = 'Take one down, pass it around,\n%d bottle of beer on the wall.\n\n';
format_none2 = 'Take it down, pass it around,\nNo bottles of beer on the wall.\n';

fprintf([format_plural1 format_plural2], 99,99,num_array,2)
fprintf([format_plural1 format_sing2], 2,2,1)
fprintf([format_sing1 format_none2], 1,1)
```

Whitespace

```
Whitespace version of 99		bottles		of beer (Bottles.ws) 2003-04-01
See http://compsoc.dur.ac.uk/whitespace/ for	details+interpreter
Example by Andrew	Kemp <ajwk@pell.uklinux.net>

(*All*		space/tab/linefeed characters		are			significant!)
```

Cow

```
moOMoOMoOMoOMoOmoOMoOMoOMoOMoOMoOMoOMoOMoOMoOM
    oOMMMmoOMMMMoOMoOMoOMoOMoOMoO
MoOMoOMoOMoOMoOMoOMoOMoOMoOMoOMoOMoOMoOMoOMoOM
    oOMMMmoOMMMOmmMoOMoOMoOMoOMoO
MoOMoOMoOMoOMoOMoOMoOMMMmoOMMMMoOMoOMMMmoOM
    MMMoOMoOMoOMoOMoOMoOMoOMoOMoOMoO
MoOMoOMoOMoOMoOMoOMoOMoOMoOMoOMoOMoOMoOMoOMoOM
    oOMoOMoOMoOMoOMoOMoOMoOMoOMoO
MoOMoOMoOMoOMoOMoOMoOMoOMoOMoOMoOMoOMoOMoOMoOM
    oOMMMmoOMMMMoOMMMmOomOomOomOo
mOomOoMMMMoOmoOmoOmoOmoOmoOmoOMMMmoOMMMMoOMo
    OMMMmoOMMMMoOMMMmoOMMMMoOMMMmoO
MMMMoOMoOMMMmoOMMMMoOMMMmoOMMMMoOMoOMMMmoO
    MMMMoOMMMmoOMMMMoOMoOMMMmoOMMMMoO
```

Piet (@!@#$%!^!)

If you are interested

- http://www.99-bottles-of-beer.net/

Thousands out there

- http://en.wikipedia.org/wiki/Timeline_of_programming_languages
- http://www.oreilly.com/news/graphics/prog_lang_poster.pdf

The most popular PLs?

- Popularity:
  - Most widely used?
  - Most lines of codes?
  - Most jobs?
  - Most courses/projects?
  - Most search engine queries?
- http://www.welton.it/articles/language_popularity.html
- http://www.developer.com/lang/article.php/3390001

History

- Early History : The First Programmer
- The 1940s: The First Computers
- The 1950s: The First Programming Languages
- The 1960s: An Explosion in Programming Languages
- The 1970s: Simplicity, Abstraction, Study
- The 1980s: New Directions and OO
- The 1990s: Consolidation, Internet, Libraries, and Scripting
- The Future

The First Programmer

- Before the birth of computers
- Jacquard loom (early 1800s)
  - translated card patterns into cloth designs.
  - *http://www.youtube.com/watch?v=Sn337QHU0AY*

http://www.smith.edu/hsc/silk/History/jacquard.html

www.nndb.com/people/799/000097508/

The First Programmer

- Charles Babbage's analytical engine (1830s and 1840s)
  - Devoted entirely to computation
  - Programs: cards with data and operations
  - Difference Engine: which inspired Analytical Engine (the design was realized in 1991)

    http://www.youtube.com/watch?v=KL_wy-CxBP8
- Ada Lovelace – first programmer (daughter of Byron)

http://en.wikipedia.org/wiki/Charles_babbage

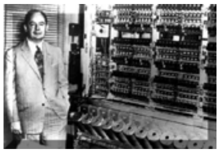
http://en.wikipedia.org/wiki/Ada_Lovelace

The First Computers

- **ENIAC (1943)**
  - First electronic computer
  - U. Penn
- **EDVAC (1945)**
  - John von Neumann
  - von Neumann architecture

    "Stored program": data and programs in the same space

http://en.wikipedia.org/wiki/ENIAC



www.agers.cfwb.be/apsdt/figinfo26.htm

The First Computers

- **Z3 (1941)**
  - Konrad Zuse
  - First digital computer
  - Electromechanical, rather than electronic

irb.cs.tu-berlin.de/.../de/Konrad_Zuse_2.html

- **Plankalkul (Plan Calculus) : (1945)**
  - Eventually published in 1972
  - First compiler implemented in 2000

http://en.wikipedia.org/wiki/Konrad_Zuse

Machine Codes and Assembly Language

- **Machine code**: bit sequences

```
000000 00001 00010 00110 00000 100000
100011 00011 01000 00000 00001 000100
000010 00000 00000 00000 10000 000001
```

- **Assembly program**: symbolic representation of machine codes

```
LDA   SUB
CMA
INC
ADD  MIN
STA  DIF
```

The 1950s: The First Programming Languages

- FORTRAN: the first higher-level programming language

Languages following FORTRAN

- COBOL
- Algol60
- LISP
- APL

FORTRAN

- The first language
  1954-1957
  John Backus, et. al. (IBM)

- Scientific and engineering applications (FORmula TRANslation).
- Goal: generate fast machine code. Its compiler is still among the most efficient.
- Contributions: array, loops by indexed variables, if-statement
- Still widely used today (Fortran, II, III, IV, 66, 77, 90, 95, 2003, 2008).

FORTRAN

- John Backus: IBM group

1977 ACM Turing Award: “for profound, influential, and lasting contributions to the design of practical high-level programming systems, notably through his work on FORTRAN, and for seminal publication of formal procedures for the specification of programming languages.”

http://www.youtube.com/watch?v=xQtT2sRkOEw

www.columbia.edu/acis/history/backus.html

Major languages following FORTRAN

- COBOL
- Algol60
- LISP
- APL

COBOL

- COmmon Business-Oriented Language
  1959-1960
  Grace Hopper, et. al. (US Department of Defense)

- Business applications: banks and corporations
- Still widely used

http://en.wikipedia.org/wiki/Grace_Murray_Hopper

COBOL

- Goal: allow nonprogrammers to read/understand programs
- Consequences:
  - Very wordy, like English
    - *C++* vs. *ADD 1 TO COBOL GIVING COBOL*
    - Can be difficult to write complex algorithms
  - Human readability improved, or only complicated?
- Contributions:
  - Record structure
  - Separate data structures from execution
  - Output formatting by examples
- COBOL 2002: OO programming

Algol60

- ALGOrithmic Language, 1958-1960
  a committee of European and American computer scientists (John Backus and John McCarthy involved)
- Contributions:
  - free-format (which modern language is not free-format?)
  - Backus-Naur forms (BNF) for defining syntax
  - type declarations for variables,
  - block-structure, begin-end
  - recursion,
  - pass-by-value parameters
- Impacts:
  - one of the most influential programming languages
  - most imperative languages are derivatives of Algol: Pascal, C/C++, Ada, Java.
  - standard way of describing algorithms in research papers for 30 years.

LISP

- LISt Processor
  late 1950s
  John McCarthy (MIT, at Stanford now)
  1971 Turing Award for contributions in AI.
- AI applications, still dominating
- Contributions:
  - first one to depart from imperative/procedural paradigm: functional programming language
  - Garbage collection
  - Recursion, s-expression
- Limitations;
  - Could not run efficiently on von Neumann architecture
  - LISP-specific machines
- Variants: Common LISP, Scheme
- Following LISP: ML, Haskell

APL

- A Programming Language
  late 1950s to early 60s
  Kenneth E. Iverson (Harvard and IBM)

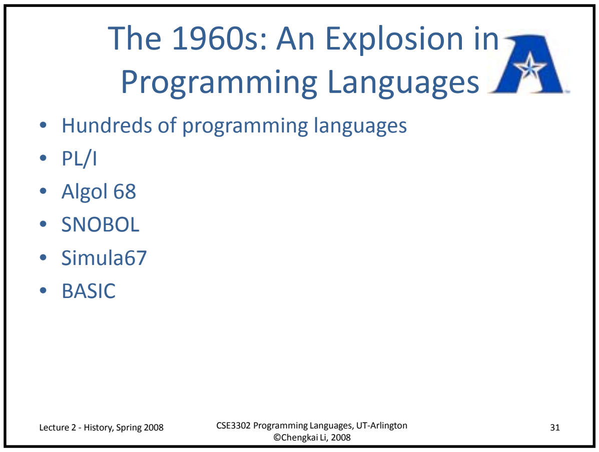
  1979 Turing Award for contributions to mathematical notation and PL theory.

- A language for programming mathematical computations
  - arrays and matrices
- Functional style, influenced FP and modern function languages
- Drawbacks:
  - No structuring
  - Greek symbol, requires special terminal keyboard
  - Extremely difficult to read

Summary of 1950s

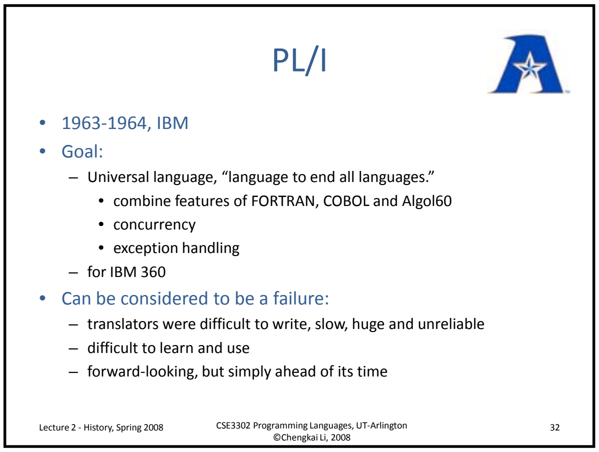
- Huge success and big impacts:
  - Pioneered imperative and functional programming
  - Still used much today
  - Many derivatives

- The 1960s is not equally fruitful



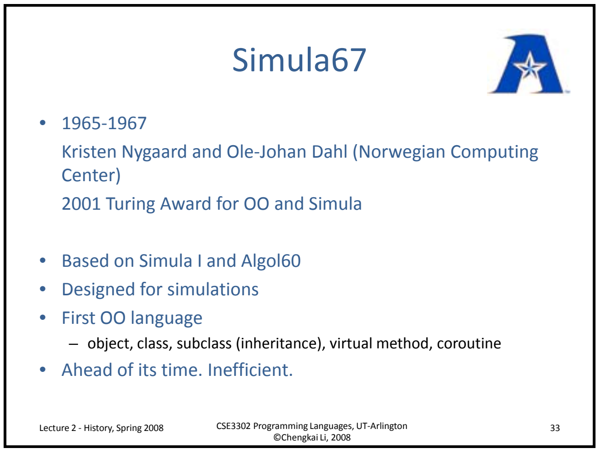
The 1960s: An Explosion in Programming Languages

- Hundreds of programming languages
- PL/I
- Algol 68
- SNOBOL
- Simula67
- BASIC



PL/I

- 1963-1964, IBM
- Goal:
  - Universal language, “language to end all languages.”
    - combine features of FORTRAN, COBOL and Algol60
    - concurrency
    - exception handling
  - for IBM 360
- Can be considered to be a failure:
  - translators were difficult to write, slow, huge and unreliable
  - difficult to learn and use
  - forward-looking, but simply ahead of its time



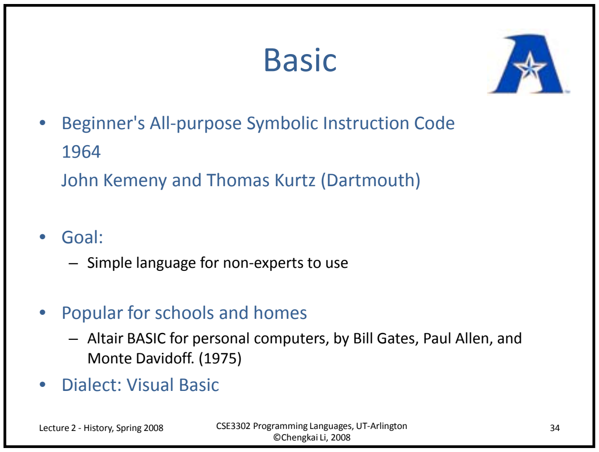
Simula67

- 1965-1967

  Kristen Nygaard and Ole-Johan Dahl (Norwegian Computing Center)

  2001 Turing Award for OO and Simula

- Based on Simula I and Algol60
- Designed for simulations
- First OO language
  - object, class, subclass (inheritance), virtual method, coroutine
- Ahead of its time. Inefficient.



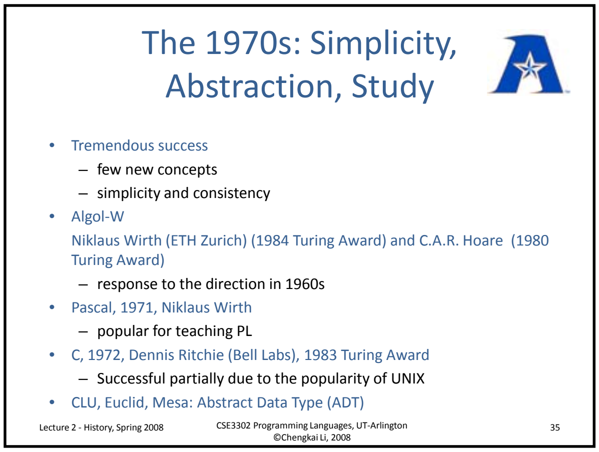
Basic

- Beginner's All-purpose Symbolic Instruction Code

  1964

  John Kemeny and Thomas Kurtz (Dartmouth)

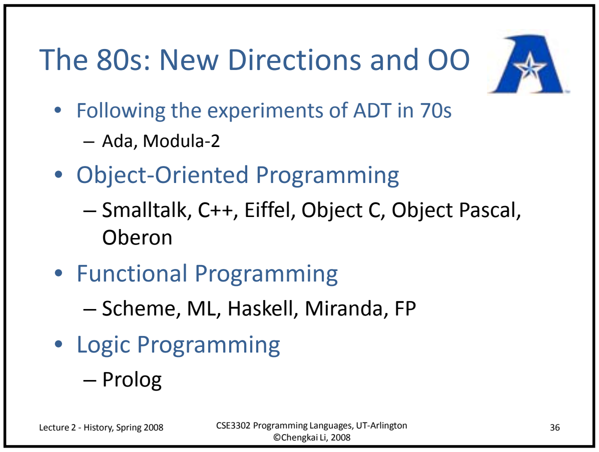
- Goal:
  - Simple language for non-experts to use
- Popular for schools and homes
  - Altair BASIC for personal computers, by Bill Gates, Paul Allen, and Monte Davidoff. (1975)
- Dialect: Visual Basic



The 1970s: Simplicity, Abstraction, Study

- Tremendous success
  - few new concepts
  - simplicity and consistency
- Algol-W

  Niklaus Wirth (ETH Zurich) (1984 Turing Award) and C.A.R. Hoare (1980 Turing Award)
  - response to the direction in 1960s
- Pascal, 1971, Niklaus Wirth
  - popular for teaching PL
- C, 1972, Dennis Ritchie (Bell Labs), 1983 Turing Award
  - Successful partially due to the popularity of UNIX
- CLU, Euclid, Mesa: Abstract Data Type (ADT)



The 80s: New Directions and OO

- Following the experiments of ADT in 70s
  - Ada, Modula-2
- Object-Oriented Programming
  - Smalltalk, C++, Eiffel, Object C, Object Pascal, Oberon
- Functional Programming
  - Scheme, ML, Haskell, Miranda, FP
- Logic Programming
  - Prolog

Ada

- Named after Ada Lovelace
  1980
  Department of Defense
  - Hundreds of languages were used by DoD
  - Required in DoD projects, 1987-1997.
- Contributions:
  - ADT (package)
  - concurrency (task)
  - exception handling
- Universal language, PL/I of 80s, but didn't fail
  - carefully designed
  - required use

Smalltalk

- 1980
  Alan Kay (2003 Turing Award), Dan Ingalls, et. al. (Xerox PARC)
- Inspired by Simula67
- Contributions
  - purest OO language
  - graphical user interface, mouse (limited its use, as such hardware was not generally available)
  - Push C++ and OO into spotlight
- Still used much today

C++

- 1980
  Bjarne Stroustrup (Bell Labs, now at TAMU)
- Extensions from Simula67 and C, "C with Classes"

C++0x – An Overview

Other Paradigms

- Functional Programming:
  - Common Lisp, Scheme, ML, Haskell (pure functional programming language)
  - Logo: teach kids to program
    http://www.youtube.com/watch?v=ohgPmdZgUmE
- Logic Programming:
  - Prolog

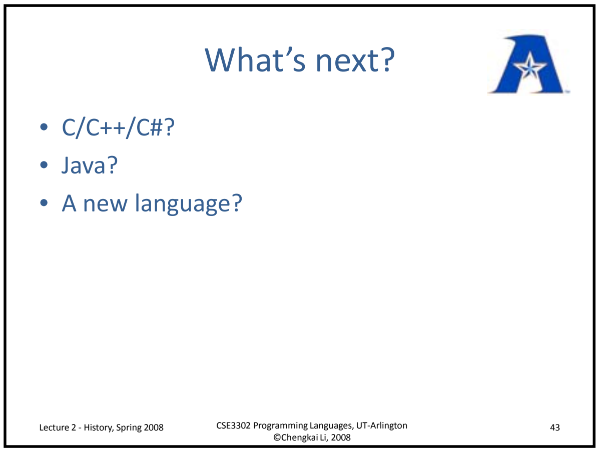
1990's: Internet, Scripting

- OO widely adopted (C++ was going to dominate)
- Then Java came
- Scripting languages, became general-purpose languages:
  Perl, Tcl, Python, PHP, …

http://www.youtube.com/watch?v=wVYsINZ5nAY

Java

- 1995, James Gosling et. al. (Sun)
- Was for embedded consumer-electronic applications (set-top box), then for Internet/Web and network applications
- Based on C++
- Differences
  - Pros: richer libraries (API), portability (compile-once, run-anywhere)
  - Cons: slower than C++, no ISO/ANSI standard (controlled by Sun).
  - references vs. pointers
  - garbage collection



What’s next?

- C/C++/C#?
- Java?
- A new language?

Lecture 2 - History, Spring 2008

CSE3302 Programming Languages, UT-Arlington
©Chengkai Li, 2008

43