Chapter 1

Introducing C

Cong Pu (Ph.D., Instructor)
cong.pu@ttu.edu

Origins of C

- C is a by-product of UNIX, developed at Bell Laboratories by Ken Thompson, Dennis Ritchie, and others.
- Thompson designed a small language named B.
- B was based on BCPL, a systems programming language developed in the mid-1960s.

Origins of C

- By 1971, Ritchie began to develop an extended version of B.
- He called his language NB (“New B”) at first.
- As the language began to diverge more from B, he changed its name to C.
- The language was stable enough by 1973 that UNIX could be rewritten in C.

Standardization of C

- *K&R C*
  - Described in Kernighan and Ritchie, *The C Programming Language* (1978)
  - De facto standard
- *C89/C90*
  - ANSI standard X3.159-1989 (completed in 1988; formally approved in December 1989)
  - International standard ISO/IEC 9899:1990
- *C99*
  - International standard ISO/IEC 9899:1999
  - Incorporates changes from Amendment 1 (1995)

C-Based Languages

- ***C++*** includes all the features of C, but adds classes and other features to support object-oriented programming.
- ***Java*** is based on C++ and therefore inherits many C features.
- ***C#*** is a more recent language derived from C++ and Java.
- ***Perl*** has adopted many of the features of C.
- Why C???

Properties of C

- Low-level
  - Access to machine-level concepts (bytes and address)
  - Operations that correspond closely to a computer's built-in instructions.
- Small
  - Limited set of features
  - Relies heavily on a "library" of standard functions
- Permissive
  - A wide degree of programming
  - Doesn't mandate the detailed error-checking

Strengths of C

- Efficiency
- Portability
- Power
- Flexibility
- Standard library
- Integration with UNIX

Weaknesses of C

- Programs can be error-prone.
- Programs can be difficult to understand.
- Programs can be difficult to modify.

Effective Use of C

- Learn how to avoid pitfalls.
- Use software tools (debuggers) to make programs more reliable.
- Take advantage of existing code libraries.
- Adopt a sensible set of coding conventions.
- Avoid “tricks” and overly complex code.
- Stick to the standard.