Introduction to Basis and Practice in Programming

Fall 2014

Jinkyu Jeong
(jinkyu@skku.edu)
Course Overview
Course Basics

- **Class hour**
  - GEDB029-45: Mon. 13:00 ~ 14:50
  - GEDB029-46: Tue. 13:00 ~ 14:50
  - 1~2 hours lectures at 400112
  - 2~3 hours lab at 400212

- **Instructor: Jinkyu Jeong**
  - jinkyu@skku.edu
  - [http://csl.skku.edu/People/Jinkyu](http://csl.skku.edu/People/Jinkyu)
  - Office: #400626 in Semiconductor Bldg.
  - Office hour: Thu. & Fri. 13:00 ~ 14:00

- **TA**
  - 김성훈, shkim@csl.skku.edu
  - 김정훈, kjh0615@csl.skku.edu
  - Office: #85561 in Corporate Collaboration Center
Prerequisites

- Prerequisite Courses
  - Engineering Computer Programming
  - ...

- Programming Skill
  - Nothing
  - You will learn how to program in C
Course Materials

- **Textbook**
  - A Book on C: Programming in C (4th Ed.),
    - Al Kelley, Ira Pohl
    - Addison-Wesley, 1998

- **References**
  - C Programming: A Modern Approach (2nd Ed.)
    - K. N. King
    - W. W. Norton & Company, 2008
  - The C Programming Language
    - Kernighan, Brian, and Dennis Ritchie

- **Course webpage**
  - [http://csl.skku.edu/GEDB029F14/Overview](http://csl.skku.edu/GEDB029F14/Overview)
Grading

- **Exams** 30%
  - Midterm
  - Final

- **Laboratory** 30%
  - Attendance
  - Completion of programming exercises

- **Programming Assignments** 30%
  - Programming assignments (up to 4)

- **Participation & quiz** 10%
  - Homework & quiz
  - Attendance
Academic Honesty

- All work submitted for credit must be your original ones.
- Cheating on lab or homework
  - “F” grade and a report to the Dean
- Cheating on examination
  - report to the president of SKKU

- No exception on dishonesty
Computers
Computers?

- A general-purpose device that can be **programmed** to carry out a finite set of **arithmetic** or **logical** operations.
- Since a sequence of operations can be readily changed, the computer can solve more than one kind of problem.
Conceptual Model of Computers

- Turing Machine

Tape

Read-write head

Table of Rules
History

- The 1st generation: Vacuum tube

  - MARK-1, 1941
  - ENIAC, 1946
  - EDSAC, 1949
  - EDVAC, 1951
  - UNIVAC 1, 1951

Von Neumann: "Stored Program"
History

• The 2nd generation: Transistor
  • Transistor invented in 1954
  • Operating system, FORTRAN, COBOL, ALGOL

• The 3rd generation: Integrated Circuit
  • IC developed by Jack St. Clair Kilby, Texas Instruments in 1959
  • Nobel Prize laureate for Physics in 2000
History

• The 4th generation: LSI, VLSI
  • Large Scale Integrated circuit
  • Minicomputers and micro computers appeared
  • Network, Data Base system
  • Computers became popular by PC
General Computer Architecture

- von Neumann machine (Stored-program computer)

Display, disks, keyboard, mouse, scanner, printer, network, …
Components of Computers

- CPU (Central Processing Unit): the hardware within a computer system which carries out the instructions of a computer program by performing the basic arithmetical, logical, and input/output operations of the system.
Components of Computers

- Memory: physical devices used to store programs (sequences of instructions) or data (e.g., program state information) on a temporary or permanent basis for use in a computer or other digital electronic device

- Examples: RAM, ROM, Disks (HDD, SSD), ...
All you have to know about computers for C programming

- Computers play with numbers only
  - everything is represented by numbers
    - pictures, numbers, music
  - binary: bit, byte, word, half word
Computational Thinking
Computers

▪ What a computer can do?
  • Everything when you can explain exactly how to do
  • E.g., “check if a number is a prime”

▪ What a computer cannot do?
  • Everything that you cannot explain exactly how to do
  • E.g., Feeling, thinking, ...

▪ What a computer can do better than you
  • Compute things extremely faster than you can do
  • 2.6 billion computations in one second
Problem Solving with Computers

- You need a language to tell computers to do what you want
  - Korean, English, ... natural languages are used by human
    - Very difficult for computers to understand

- Binary numbers are used by computers
  - 00001111 10111111 01000101 11111000
  - 00001111 10111111 01001101 11111000
    - Very difficult for human to understand
Programming Languages

- A language that describes what to do for a computer
- Machine language
  - binary
  - only an expert use it
- Assembly language
  - alphabet
  - most embedded system
- High-level language
  - C, Java, C++, ...
  - Python, Matlab, ...
  - most software

```
MOV AX, MIDSCORE
MOV CX, FINALSCORE
ADD AX CX
MOV TOTALSCORE, AX
```

```
00001111 10111111 01000101 11111000
00001111 10111111 01001101 11111000
00000011 10100001
01100110 10001001 01000101 11111010
```

TotalScore = MidScore + FinalScore;
Problem Solving with Computer: Example

- Check if a number is a prime.
- Tell me how to determine step by step.

2, 3, 5, 13, 89, 233, 1597, 28657, 514229, 433494437, 2971215073, 99194894755497, 106634041749171059581572169, ...

...
**Problem Solving with Computer:** Example

- Recall what a prime number is.

A prime number is a natural number that has exactly two distinct natural number divisors: 1 and itself
Problem Solving with Computer: Example

- A straightforward (simple ?) version:
  - Input \( x \)
  - Check if 2 divides \( x \). If True, \( x \) is not a prime
  - Check if 3 divides \( x \). If True, \( x \) is not a prime
  - ...
  - Check if \( k \) divides \( x \). If True, \( x \) is not a prime.
  - If False, \( x \) is a prime.

- What is \( k \)?
  - the largest natural number which is at most \( \sqrt{x} \)
  - Think about why??
Problem Solving with Computer: Example

Start

Declare x

Input x

2 divides x?
  T
  3 divides x?
    T
    4 divides x?
      F
      5 divides x?
        ...
        k divides x?
          T
          x is not a prime
          Stop
        F
        x is a prime

F

F

F

F

x is a prime

Stop
Problem Solving with Computer: Example

- A better version
  - Line 1: Input $x$
  - Line 2: Set $k$ with $\sqrt{x}$
  - Line 3: Assign 2 to $n$.
  - Line 4: If $n > k$, $x$ is a prime and goto Line 8
  - Line 5: If $n$ divides $x$, $x$ is not a prime and goto Line 8
  - Line 6: Increase $n$ by 1
  - Line 7: Goto Line 4
  - Line 8: Stop
Problem Solving with Computer: Example

Start

Declare n, k

Input x

Set k with $\sqrt{x}$

Assign 2 to n

$n > k$?

$n$ divides $x$?

$n \leftarrow n + 1$

$x$ is a prime

$x$ is not a prime

Stop
Programming

- **Flow chart:**
  - A type of diagram that represents an algorithm or process, showing the steps as boxes of various kinds, and their order by connecting these with arrows.

- **Why flow chart and algorithms?**
  - This diagrammatic representation can give a step-by-step solution to a given problem.
  - Flowcharts are used in analyzing, designing, documenting or managing a process or program.
A sequence of instructions written to perform a specified task for a computer
**Program**

- A list of instructions

```c
#include <stdio.h>
#include <math.h>

main() {
    int n, k, x;
    scanf( "%d", &x );
    k = sqrt(x);
    n = 2;
    lable1:
        if( n > k ) {
            printf( "%d is a prime\n", x );
            goto lable2;
        }
        if( x % n == 0 ) {
            printf( "%d is not a prime\n", x );
            goto lable2;
        }
        n = n + 1;
        goto lable1;
    lable2:
        return;
}
```

### Flowchart

1. **Start**
2. **Declare n, k**
3. **Input x**
4. **Set k with the proper value**
5. **Assign 2 to n**
6. **n > k ?**
   - **T** → **x is a prime**
   - **T** → **n divides x ?**
     - **T** → **x is not a prime**
     - **n <= n + 1**
     - **Stop**
   - **F** → **n = n + 1**
   - **goto lable1**
7. **lable2:**
   - **return;**
8. **Stop**
• What are programs for?

```c
#include <stdio.h>
#include <math.h>

main() {
    int n, k, x;
    scanf("%d", &x);
    k = sqrt(x);
    n = 2;

    lable1:
    if (n > x) {
        printf("%d prime\n", x);
        goto lable2;
    }
    n = n + 1;
    goto lable1;

    lable2: return;
}
```

test.c

test.exe


Programming

• What is programming?
  • the process of designing, writing, testing, debugging, and maintaining the source code of computer programs

• What is programming language?
  • an artificial language designed to express computations that can be performed by a computer
  • C, C++, Java, Perl, Basic, Pascal, Fortran, COBOL, ...
Homework

- **Hour of Code**
  - [http://code.org](http://code.org)
  - Tutorials for beginners
Introduction to C Language
History of C Language

- Developed at Bell Lab., 1972 for system-level programming.
- Used for implementing Unix OS

- BCPL (Basic Combined Programming Language)
  - B language (Ken Thompson)
  - C language (Dennis Ritchie)
History of C Language

- **1972: developed by Dennis Ritchie**
  - To develop an OS (Unix) for PDP-11
  - Small and efficient

- **1989: ANSI C**
  - Portable
  - (= C90 by ISO)

- **C99**
  - By ISO
  - Not by MS

- **C11**
  - By ISO
Advantages of C Language

- **Efficient**
  - Developed for low-level (machine-level) execution

- **Portability**
  - Applicable to virtually all platforms from PCs to Supercomputers

- **Powerful**
  - Provides various data types and operators

- **Flexibility**
  - Applicable from system-level to application-level programming

- **Many Standard Libraries**
  - Input/Output, String handling, Storage allocation, ...
Disadvantages of C Language

- **Error Prone**
  - Difficult to detect errors resulting from its flexibility

- **Difficulty**
  - Difficult to understand and modify it due to many functionalities
Programming Development Cycle

- **Making goals**
  - Understanding of requirements in given problems

- **Writing algorithms**
  - Writing pseudo codes or flow charts

- **Coding**
  - Translate the algorithm to C programming language

- **Compile & Link**
  - Translate a C program to machine codes

- **Execution & Debugging**
  - Test whether the program generates correct results
  - Modifying programs to correct errors found in testing

- **Maintenance**
  - Keep improving and fixing issues in the program
Programming Tools

- **Text Editor**
  - Store the written program (a simple text file) to storage
  - Replace the extension with c (e.g., filename.c)
  - Use vi or emacs editor on Unix
  - Source file means a human-readable code before compiling it

- **Preprocessor**
  - As a part of compiler, it processes the lines beginning with ‘#’ in the source code before compiling the source code
### Programming Tools

- **Compiler (& linker)**
  - Check the syntax of a source program and convert it to a machine-understandable (executable) binary language

```
source program
  (***.c)**

  
  `**compiler**`

  `**exe**`
  `**(a.out)**`

executable program
  (**~.exe**)
```
Building Executable Program

Source File → Preprocessed Source File → Object File → Executable File

Preprocessing

Preprocessor

Header File

Compile

Compiler

Library

Link

Linker

Building Executable Program