上海交通大學 SHANGHAI JIAO TONG UNIVERSITY

Course organization

- Course introduction (Week 1)
  - Code editor: Emacs
- Part I: Introduction to C programming language (Week 2 - 9)
  - Chapter 1: Overall Introduction (Week 1-3)
  - Chapter 2: Types, operators and expressions (Week 4)
  - Chapter 3: Control flow (Week 5)
  - Chapter 4: Functions and program structure (Week 6, 7)
  - Chapter 5: Pointers and arrays (Week 8)
  - Chapter 6: Structures (Week 9)
  - Chapter 7: Input and Output (Week 10)
- Part II: Skills others than programming languages (Week 11- 12)
  - Debugging tools（Week 11）
  - **Keeping projects documented and manageable （Week 12）**
  - Source code managing （Week 12）
- Part III: Reports from the battle field (student forum) (week 12 – 16)

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Course review

Chaochun Wei
Shanghai Jiao Tong University
Spring 2013

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Contents

- Course review
- Final project
  - Presentation content
  - Presentation arrangement

上海交通大學 SHANGHAI JIAO TONG UNIVERSITY

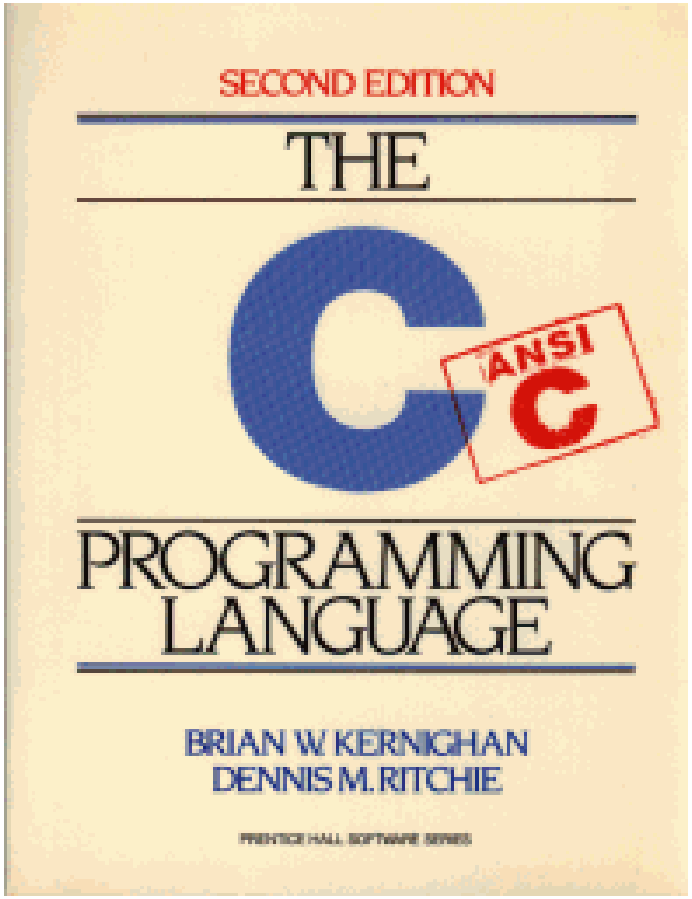
Course review

- Course introduction (Week 1)
  - Code editor: Emacs
- Part I: Introduction to C programming language (Week 2 - 9)
  - Chapter 1: Overall Introduction (Week 1-3)
  - Chapter 2: Types, operators and expressions (Week 4)
  - Chapter 3: Control flow (Week 5)
  - Chapter 4: Functions and program structure (Week 6, 7)
  - Chapter 5: Pointers and arrays (Week 8)
  - Chapter 6: Structures (Week 9)
  - Chapter 7: Input and Output (Week 10)
- Part II: Skills others than programming languages (Week 11- 12)
  - Chapter 8: GDB（Week 11）
  - Chapter 9: Make（Week 12）
- Part III: Reports from the battle field (student forum) (week 13 – 16)
  - Student presentation (week13-15)
  - Project demo (week 16)

Week 1

Text Book

The C Programming Language, Second Edition by Brian W. Kernighan and Dennis M. Ritchie. Prentice Hall, Inc., 1988.



References

- Emacs
  - tutorial: http://www.gnu.org/software/emacs/tour/
  - Manual: http://www.gnu.org/software/emacs/manual/emacs.pdf
- GDB
  - Document: http://www.gnu.org/software/gdb/documentation/

Grading

- Homework 50%
- Projects 30%
  - Design and implementation of a diff program for lists of different biological entities
- Presentation 20%

作业规定

- 作业允许合作，但是必须注明各人的贡献
- 作业报告必须用自己的语言独立完成
- 严禁抄袭
  - 抄袭者：不及格(F)
  - 被抄袭者：成绩降一级（A→B, B→C, C→D, D→F）

Week 2, 3

Emacs tutorial

- References:
  - Emacs Reference card: emacs.pdf
  - Emacs Tutorial
    - C-h t
  - Linux and Perl Tutorial
    - http://cbb.sjtu.edu.cn/~ccwei/pub/courses/2013/programming_language_for_bioinformatics/unix_and_perl_v2.3.4.pdf

Introduction to C

- Brief introduction to C program language
  - A simple C program
  - Elements of a C program
  - Source and header files
  - Preprocessor
  - Arrays and pointers
  - Basic types and operators
  - Structures
  - Control flow
    - Conditional switch
    - loop

Chapter 2. Types, operators and expressions

Week 4

Basic Types and Operators

- Basic data types
  - Types: *char, int, float and double*
  - Qualifiers: *short, long, unsigned, signed, const*
- Constant: 0x1234, 12, “Some string”
- Enumeration:
  - Names in different enumerations must be distinct
  - `enum WeekDay_t {Mon, Tue, Wed, Thur, Fri};`
    `enum WeekendDay_t {Sat = 0, Sun = 4};`
- Arithmetic: +, -, *, /, %
  - prefix ++i or --i ; increment/decrement before value is used
  - postfix i++, i--; increment/decrement after value is used
- Relational and logical: <, >, <=, >=, ==, !=, &&, ||
- Bitwise: &, |, ^ (xor), <<, >>, ~(ones complement)

2.12 Precedence and associativity of operators

Operators	Associativity
( ) [ ] -> .	Left to right
! ~ ++ -- + - * & (type) sizeof	Right to left
* / %	Left to right
+ -	Left to right
<< >>	Left to right
< <= > >=	Left to right
== !=	Left to right
&	Left to right
^	Left to right
|	Left to right
&&	Left to right
||	Left to right
?:	Right to left
= += -= *= /= %= &= ^= |= <<= >>=	Right to left
,	Left to right

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Chapter 3. Control Flow

Week 5

Chapt 5 Control Flow

- Statement
- Block
- If, else
- Switch
- Loops: for, while
- Break, continue

Chapter 4. Function and Program Structure

Week 6,7

Functions

- Break large program intro smaller ones
- Enable people to build on existing codes
- Hide details of operation
  - Clarify the whole program
  - Make it easier to modify a program

4.4 Scope rules

- Source codes can be in different files
  - Variable declaration organization
  - Variable initialization

- Declaration and definition of an external variable

  extern int sp;

  extern double val [];

  /* this is a declaration */

  Initialization goes with the definition

Program structure

- Header files
- Static variables
- Register variables
- Block structure
- Initialization
- Recursion
- The C preprocessor

Chapter 5. Points and Arrays

Week 8

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Contents

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Chapter 6 Structures

Week 9

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Contents

6.1 Basic of structures

- **A struct declaration defines a type.**

  e.g.: *struct point {int x; int y} x, y, z;*

- Access a member of a structure: *structure-name.member*

  e.g.: struct point pt; pt = {1, 100};

  printf("%d, %d", pt.x, pt.y);

- *A Structure of structures*
  - *E.g.:*

```
struct rect {
    struct point pt1;
    struct point pt2;
};
```

6.5 Self-referential structures

- Recursive declaration of a structure
  - E.g.,

```
struct tnode {
    char *word;         /* point to the text */
    int count;          /* number of occurrences */
    struct tnode *left;  /* left child */
    struct tnode *right; /* right child */
};
```

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Chapter 7 Input and Output

Week 10

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Contents

7.1 Standard input and output

- Input
  - Read from standard input (keyboard)

    int getchar(void)
  - Read characters from an file infile.

    prog < infile
  - Take input from other program otherprog

    Otherprog | prog

7.1 Standard input and output

- Output
  - output to standard output (screen)

    int putchar(int)
  - Output to a file outfile

    Prog > outfile
  - Output to other program otherprog

    prog | anotherprog

More details see hands-on example 7.1

7.2 Formatted output --printf

- printf
  - syntax of printf

```
int  printf(char *format, arg1, arg2, …)
```

  - Format string
    - Normal characters
    - Conversion characters (begins with a %)
    - A width or precision may be specified as *
  - E.g. , to print at most max characters from a string s:

```
printf("%.*s", max, s);
```

7.2 Formatted output --printf

Format string (%)

Character	Argument type; printed as
d, i	Int; decimal number.
o	Unsigned int; unsigned octal number (without a leading zero)
X, x	Unsigned int; unsigned hexadecima number (without a leading 0x or 0X), using abcdef or ABCDEF for 10,11, 12, 13, 14 and 15.
u	Unsigned int; unsigned decimal number
c	Int; single character.
s	Char *; print a string, until a '\0' or the number of characters given by the precision
f	Double; [-]m.dddddd, where the number of d's is given by the precision (default 6)
e, E	Double; [-]m.dddddd e±xx or [-]m.dddddd E±xx, where the number of d's is given by the precision (default is 6)
p	Void *; pointer (implementation-dependent representation)
%	No argument is converted; print a %

7.5 File access

- Read, write, append
- Open a file

```
FILE *fp;
FILE *fopen(char *name, char *mode);
```

Mode

“r”: read
“w”: write
“a”: append
“b”: binary files

7.8 Miscellaneous Functions

- Storage management
  - ***void *malloc(size_t n);***
    - Returns a pointer to n bytes of uninitialized storage, or NULL if the request can not be satisfied
  - ***void *calloc (size_t n, size_t size)***
    - Returns a pointer to an array of n objects of the specified size, or NULL if failed.
  - ***void *realloc(void *p, size_t size);***
    - Changes the size of the object pointed by p to size. Returns a pointer to the new space or NULL if the request can not be satisfied, in which case *p is unchanged

More details see hands-on experiment 7.8

Chapter 8 GDB in Emacs

Week 11

Contents

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Chapter 9 the Make tool

Week 12

上海交通大學 SHANGHAI JIAO TONG UNIVERSITY

Contents

Reference: GNU make
http://www.gnu.org/software/make/manual/make.html#Top

9.1 Make

- Make is a Unix utility tool, which
  - Contains a set of instruction to build a large program;
  - Determines automatically which pieces of the program should be recompiled, and
  - runs the compilation automatically
- can be used to describe any task where some files depends on others
- To use make, you need to create a file called *Makefile*

See more details on Makefile_1

The Final Project

- Final project (50)
  - Report (30)
  - **Demo**
  - Presentation (20)
    - content
- Presentation and demo arrangement

Presentation and demo

- Presentation
  - Content
    - 50% about your project (project design and/or implementation)
    - 50% about the C program language
      - A chapter will be assigned to everyone
      - Your memorable C programming experience
  - Time: 8 minutes (6 + 2)
- Demo
  - Content
    - Show and tell your project
    - Test your program with different input files
    - Time: 4 minutes

Random number generation

- Function rand( ), frand( )

- Set the seed for rand()
  - srand(unsigned)

See details about the randomization in presentation_assignment.c