ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

Input/output file

Department of Information System

SoICT, HUST

Standard input/output streams

- 3 standard streams are opened by a program:
  - stdin: for input
  - stdout: for output
  - stderr: for error
- The direction of these streams to peripherals depends on the program, the default is keyboard for **`stdin`**, screen for **`stdout`** and **`stderr`**
- scanf() and printf() are functions that read/write in **`stdin`** and **`stdout`**
- perror() prints the errors to **`stderr`**

Example

Input.c

```
#include <stdio.h>
void main()
{
  int a;
  if ( scanf("%d", &a) != 1 )
    perror("This is not integer\n");
  else
    printf("Input number%d", a);
}
```

```
$input ↵
10 ↵
Input number10
$input ↵
abc ↵
This is not integer
$input >out.txt ↵
10 ↵
$input >out.txt ↵
abc ↵
This is not integer
```

Redirect stdout to file out.txt

Input/output file

- Files need to be *opened* before use.
- Associate a "***file handler***" to each file
- Modes: read, write, or append
- File input/output functions use the file handler (*not* the filename).
- Need to *close* the file after use.
- Basic file handling functions: **fopen()**, **fclose()**, **fscanf()**, **fprintf()**.
- FILE * is the file handler type

Example

```
#include <stdio.h>

int main()
{
  FILE * out = fopen("hello.txt", "w");

  if (out == NULL)
  {
    perror("Unable to write to the file.\n");
    return 1;
  }

  fprintf(out, "Hello world");
  fclose(out);

  return 0;
}
```

Open file to write

Write data to file

Close file when terminate

Modes in open file

- r: read
- w: write
- a: append
- r+: read/write on a new file if not exist
- w+: write on a new file if not exist
- a+: append on a new file if not exist

fprintf() và printf()

- fprintf works exactly as printf except the output on stdout.
- printf(…) = fprintf(stdout, …)
- Similarly we have other output streams:
  - fputs(char*, FILE*) and puts(char*)
  - fputc(char, FILE*) and putchar(char)

fscanf() and scanf()

- fscanf work exactly as scanf except the input on stdin.
- The return type of fscanf() and scanf() is the number of elements read.
- Similarly we have other input streams:
  - char* fgets(char*, int *maxlen*, FILE*) and
  - char*gets(char*);
  - int fgetc(FILE*) and int getchar(void)

Input data

- Both **scanf()** and **fscanf()** return:
  - the number of input items converted and assigned successfully
  - or the constant value **EOF** when an error or end-of-file occurs
- Therefore we can also check EOF using function fscanf
- The input process is the process of scanning data on the buffer according to a specific data type.
- After each successful scan, the buffer's pointer shifts to the next space in order to scan data for the next reading time.
- When there is no more data in the buffer, the buffer's pointer points to EOF.
  - To check whether the pointer is at the EOF position or not, using function int feof(FILE*)

Input formats

- Input number following formats %d, %l, %x,…, will skip spaces and ↵
- %s scans a string not including spaces and ↵.
- %c scans any character at the pointer's position (including spaces and ↵)
- Example, if we enter "**12 ab↵**"
  - **"%d%s" gives us a number** 12 and a string "ab"
  - **"%d%c%s" gives us a number** 12, a space and a string "ab"
  - **"%d %c%s" gives us a number** 12, a character a and a string "b"
  - **"%s%s" gives us two strings** "12" and "ab"
  - **"%d%s%c" give us a number 12, a string "ab" and a character** ↵

fflush()

- Function fflush(<*stream*>) is used to clean an input/ouput buffer
- When a file is closed, its buffer will be automatically cleaned
- fflush() should be used before scanning a character or a string with gets() or fgets()
- Like enter a character, gets() does not skip any character when scanning. This function scans all spaces and stops at the first ↵. However, ↵ does not include in the target string.

Example

Input.c

```
#include <stdio.h>

void main()
{
  int a;
  char s[20];
  printf("Input a number: ");
  scanf("%d", &a);

  fflush(stdin);
  printf("Input a string: ");
  gets(s);

  printf("number %d, string %s",
  a, s);
}
```

C:\>input ↵
Input a number: 12↵
Input a string: ab↵
number 12, string ab

%d only gets two characters '12' to convert to number, the redundant character ↵ is cleaned by fflush() before enter a string by gets()

Calculate total words of a file

```
#include <stdio.h>

int main()
{
  int count = 0;
  char s[80];
  FILE * f = fopen("text.txt", "r");
  if (f == NULL)
  {
    perror("Failure when opening text file.txt\n");
    return 1;
  }
  while (!feof(f))
     dem += fscanf(f, "%s", s);
  fclose(f);
  printf("Total number of words: %d", dem);
  return 0;
}
```

Open file to read

Read a word each time

fgetc() and fputc()

```
FILE *input, *output;
input = fopen( "tmp.c", "r" );
output = fopen( "tmpCopy.c", "w+" );

ch = fgetc( input );
while( ch != EOF ) {
  fputc( ch, output );
  ch = fgetc( input );
}

fclose(input);
fclose(output);
```

fgets()

```
#include <stdio.h>
#define LINE_LENGTH 80

main()
{
   FILE* fp;
   char line[LINE_LENGTH];
   int count=0;
   fp=fopen("input.txt","r");
   while ( fgets(line, LINE_LENGTH, fp) != NULL)
        count++;
   printf("File contains %d lines.\n", count);
   fclose(fp);
}
```

Text file vs. binary file

- There is no difference among byte data in binary file
- In text file, byte data are categorized as displayed character and control character.
- A text file is marked as end by a control character (e.g., 26 in DOS)
- To open a file in text mode, we add ‘t’ in the open mode ("r+t", "wt", ...).
- To open a file in binary mode, we add ‘b’ in the open mode ("r+b", ...).

Input/ouput in binary mode

```
size_t fread(void* buf, size_t size,
                size_t num, FILE* f);
size_t fwrite(void* buf, size_t size,
                 size_t num, FILE* f);
```

- Read and write data in the memory with the pointer *buf*, with the total elements *num*, size of each element *size*

Example:

```
int a[10];
f=fopen("integer.dat", "r+b");
fread(a, 10, sizeof(int), f);
```

Exercises

1. Write a program to create a text file F3 by concatenate two text files F1 and F2

F1 = "ha noi"; F2 = " viet nam" F3 = "ha noi viet nam"

2. Write a program to remove all comments from a C program which is stored in a file. The name of the file is entered from the keyboard. Assume that the program does not have syntax errors.

3. Assume that a data file consisting information about weather in a year has the format for each line as follow: \

<day>/<month> <lowest temperature>-<highest temperature> <humidity>

1/1 11-17 70

2/1 12-17 75

…

4. Write a program read data from this file and print the average temperature of all months in a year, the most humid month and the dryest month.

ĐẠI HỌC BÁCH KHOA

25 YEARS ANNIVERSARY SOICT

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

Thank you for your attentions!

soict.hust.edu.vn/ fb.com/groups/soict