Using Files

- Declare a variable, called file pointer, of type FILE *
- Use function fopen to open a named file and associate this file with the file pointer
- Use the file pointer and I/O functions to access the opened file
- Use function fclose to close the file

1

An Example

```
#include <stdio.h>
#include <stdlib.h>
main() {
   char ch;
   FILE *fp;
   if ((fp = fopen("data.txt", "r")) == NULL) {     /* 開啟檔案 */
      printf("Error: can't open file\n");
      exit(1);
   }
   while (fscanf(fp, "%c", &ch) != EOF)             /* 輸入字元 */
      printf("%c", ch);                             /* 輸出字元 */
   fclose(fp);                                      /* 關閉檔案 */
}
```

2

File Pointers

- A file pointer points to a memory location that contains information about the file
- The location of a buffer
- The current character position in the buffer
- Whether the file is being read or written
- Whether errors or end of file have occurred

3

Opening a File

- The fopen function has the following form
  fp = fopen(file-name, mode)
- The mode may be
  - "r": The file is open for reading, is used for input
  - "w": The file is open for writing, is used for output, erases existing data
  - "a": The file is open for appending, is used for output, keeps existing data

4

Closing a File

- The fclose function has the following form
  fclose(fp)
- You should be sure to close any files you open
- Exiting from a program automatically closes any open files

5

File I/O

- Input
  fscanf(fp, format, …)
  ch = getc(fp)
- Output
  fprintf(fp, format, …)
  putc(ch, fp)

6

An Example

```
#include <stdio.h>
#include <stdlib.h>
main() {
   char ch;
   FILE *infp, *outfp;
   if ((infp = fopen("data.txt", "r")) == NULL) {      /* 開啟輸入檔案 */
      printf("Error: can't open input file\n");  exit(1);
   }
   if ((outfp = fopen("result.txt", "w")) == NULL) { /* 開啟輸出檔案 */
      printf("Error: can't open output file\n");  exit(1);
   }
   while (fscanf(infp, "%c", &ch) != EOF)            /* 輸入字元 */
      fprintf(outfp, "%c", ch);                      /* 輸出字元 */
   fclose(infp);   fclose(outfp);                    /* 關閉檔案 */
}
```

7

An Example

```
#include <stdio.h>
#include <stdlib.h>
main() {
   char ch;
   FILE *infp, *outfp;
   if ((infp = fopen("data.txt", "r")) == NULL) {      /* 開啟輸入檔案 */
      printf("Error: can't open input file\n");  exit(1);
   }
   if ((outfp = fopen("result.txt", "a")) == NULL) {  /* 開啟增添檔案 */
      printf("Error: can't open append file\n");  exit(1);
   }
   while (fscanf(infp, "%c", &ch) != EOF)            /* 輸入字元 */
      fprintf(outfp, "%c", ch);                      /* 輸出字元 */
   fclose(infp);   fclose(outfp);                    /* 關閉檔案 */
}
```

8

An Example

```
#include <stdio.h>
#include <stdlib.h>
main() {
   char ch;
   FILE *infp, *outfp;
   if ((infp = fopen("data.txt", "r")) == NULL) {      /* 開啟輸入檔案 */
      printf("Error: can't open input file\n");  exit(1);
   }
   if ((outfp = fopen("result.txt", "w")) == NULL) { /* 開啟輸出檔案 */
      printf("Error: can't open output file\n");  exit(1);
   }
   while ((ch = getc(infp)) != EOF)                    /* 輸入字元 */
      putc(ch, outfp);                                 /* 輸出字元 */
   fclose(infp);   fclose(outfp);                      /* 關閉檔案 */
}
```

9

Standard Files

- The library stdio.h defines three special file pointers. These files are *automatically* opened at the beginning of the program and closed at the end of the program
- stdin: standard input file, console keyboard
- stdout: standard output file, console screen
- stderr: standard error file, console screen

10

Standard Files

```
#include <stdio.h>
#include <ctype.h>
main() {
   char ch;
   while (fscanf(stdin, "%c", &ch) != EOF) {/* 由標準輸入檔輸入字元 */
      if (isupper(ch) {
         fprintf(stderr, "%c", ch);              /* 由標準錯誤檔輸出字元 */
      }
      fprintf(stdout, "%c", ch);                 /* 由標準輸出檔輸出字元 */
   }
   /* fscanf(stdin, "%c", &ch) ≡ scanf("%c", &ch) */
   /* fprintf(stdout, "%c", ch) ≡ printf("%c", ch) */
}
```

11

Redirect Standard Files

- Redirect an input file to standard input
  copyfile < infile
- Redirct standard output to an output file
  copyfile > outfile
  copyfile < infile > outfile
- Redirect standard error to an error file
  copyfile >& errorfile

12

Updating a File

- Open the original file for input
- Open a temporary file for output with a different name
- Copy the input file to the temporary file, updating the data in the file
- Close both files
- Delete the original file
- Rename the temporary file so that it once again has the original name

13

Updating a File

- Generate a new file name
  char *tmpnam(char s[]);
  tmpnam(NULL);
- Remove a file
  int remove(char *filename);
- Rename a file
  int rename(char *oldname, char *newname);

14

An Example

```
#include <stdio.h>
#include <ctype.h>
main() {
   char ch, *temp;
   FILE *infp, *outfp;
   if ((infp = fopen("data.txt", "r")) == NULL) { error(); }
   temp = tmpnam(NULL);
   if ((outfp = fopen(temp, "w")) == NULL) { error(); }
   while ((ch = getc(infp)) != EOF)
      putc(toupper(ch), outfp);
   fclose(infp);   fclose(outfp);
   if (remove("data.txt") != 0 || rename(temp, "data.txt") != 0) {
      error();
   }
}
```

15

Unget a Character

- The function ungetc can put a character back into input buffer
  int ungetc(int c, FILE *fp);

16

An Example

```
void removeComments(FILE *infp, FILE *outfp) {
   int ch, nch;
   bool commentFlag;
   commentFlag = FALSE;
   while ((ch = getc(infp)) != EOF) {
      if (commentFlag) {
         if (ch = '*') {
            nch = getc(infp);
            if (nch == '/') {
               commentFlag = FALSE;
            } else {
               ungetc(nch, infp);
            }
```

17

An Example

```
            } else {
               if (ch == '/') {
                  nch = getc(infp);
                  if (nch == '*') {
                     commentFlag = TRUE;
                  } else {
                     ungetc(nch, infp);
                  }
               }
               if (!commentFlag) putc(ch, outfile);
            }
         }
      }
```

18

Line I/O

- Input a line
  char *fgets(char buf[], int n, FILE *fp);
  fgets reads at most the next n-1 characters into the array buf, stopping if a newline is encountered; the newline is included in the array, which is terminated by '\0'. It returns buf or NULL if end of file or error occurs

19

Line I/O

- Output a line
  char *fputs(char buf[], FILE *fp);
  fputs writes the string buf (which need not contain '\n') on fp; it returns non-negative, or EOF for an error

20

An Example

```
void copyFile(FILE *infp, FILE *outfp)
{
        char buf[MAXLINE];
        while (fgets(buf, MAXLINE, infp) != NULL) {
                fputs(buf, outfp);
        }
}
```

21

Scanf

- White-space characters: any amount of white space matches any amount of white space in the input
- A percent sign followed by a conversion specification
- Any other character, which must match the next character in the input

22

Scanf

- An optional assignment-suppression flag indicated by an asterisk (*), which specifies that the value from the input should be discarded rather than stored into the argument
- An optional numeric field width
- An optional size specification (h or l)

23

Scanf

- %[…]
  a string consists of characters in … is read
- %[^…]
  a string consists of characters not in … is read

24

Sscanf and Sprintf

- Convert characters in an array to arguments
  sscanf(character-array, format, …);
- Convert arguments to a character array
  sprintf(character-array, format, …);

25

An Example

Hydrogen, H, 1, 1.008
Helium, He, 2, 4.003
Lithium, Li, 3, 6.939
Beryllium, Be, 4, 9.012
Boron, B, 5, 10.811
Carbon, C, 6, 12.011
Nitrogen, N, 7, 14.007
Oxygen, O, 8, 15.999
Fluorine, F, 9, 18.998
Neon, Ne, 10, 20.183

26

An Example

Element (Symbol)	Atomic Weight
1. Hydrogen (H)	1.008
2. Helium (He)	4.003
3. Lithium (Li)	6.939
4. Beryllium (Be)	9.012
5. Boron (B)	10.811
6. Carbon (C)	12.011
7. Nitrogen (N)	14.007
8. Oxygen (O)	15.999
9. Fluorine (F)	18.998
10. Neon (Ne)	20.183

27

An Example

```
main()
{
   char name[15];
   char symbol[3];
   char namebuf[20];
   int number;
   double weight;
   char termch;
   int nscan;

   printf("     Element (symbol)    Atomic Weight\n");
   printf("-----------------------------------\n");
   / * while (…) {…} */
}
```

28

An Example

```
while (1) {
   nscan = scanf("%15[^,], %2[^,], %d, %lf%c",
                 name, symbol, &number, &weight, &termch);
   if (nscan == EOF) break;
   if (nscan != 5 || termch != '\n') {
      fprintf(stderr, "Improper file format\n");
   }
   sprintf(namebuf, "%s (%s)", name, symbol);
   printf("%3d. %-20s %8.3f\n", number, namebuf, weight);
}
```

29