

---

# 标准I/O (二)

主讲：华清远见 老师

---

# 课程目标

按字符输入（熟练）

按字符输出（熟练）

小结



:

\ 4

-

7

'

}

# 标准I/O – 读写流

流支持不同的读写方式:

**读写一个字符:** fgetc()/fputc() 一次读/写一个字符

慢

**读写一行:** fgets()和fputs() 一次读/写一行

文本 二进制不行

**读写若干个对象:** fread()/fwrite() 每次读/写若干个对象, 而每个对象具有相同的长度

文本 二进制都可 (推荐)

# 标准I/O – 按字符输入

下列函数用来输入一个字符:

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

```
int fgetc(FILE *stream);
```

→ stream 接判

```
int getc(FILE *stream); //宏
```

```
int getchar(void);
```

- 成功时返回读取的字符; 若到文件末尾或
- 出错时返回EOF (-1) ,
- getchar()等同于fgetc(stdin)
- getc和fgetc区别是一个是宏一个是函数

# 标准I/O – fgetc – 示例

```
int ch;
```

```
ch = fgetc(stdin);
```

→ 键盘输入, 打印.

```
printf( "%c\n" , ch);
```

```
FILE *fp;
```

```
int ch, count = 0;
```

```
if ((fp = fopen(argv[1], "r" )) == NULL) {
```

```
    perror( "fopen" ); return -1;
```

```
}
```

```
while ((ch = fgetc(fp)) != EOF) {
```

```
    count++;
```

```
}
```

```
printf( "total %d bytes\n" , count);
```

→ 不断读字符, 直到失败 ( 如 open, 有字;  
里 count++, 2后打印 ).

# 标准I/O – 按字符输出

下列函数用来输出一个字符:

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

```
✓ int fputc(int c, FILE *stream);
```

→

c

stream

Q why 用 int 来着?

```
✓ int putc(int c, FILE *stream);
```

```
✓ int putchar(int c);
```

➤ 成功时返回写入的字符; 出错时返回EOF

✓ ➤ putchar(c)等同于fputc(c, stdout)

# 标准I/O – fputc – 示例

```
fputc( 'a' , stdout);  
putchar( '\n' );
```

```
FILE *fp;
```

```
int ch;
```

```
if ((fp = fopen(argv[1], "w" )) == NULL) {
```

```
    perror( "fopen" ); return -1;
```

```
}
```

```
for(ch = 'a' ; ch <= 'z' ; ch++) {
```

```
    fputc(ch, fp);
```

```
}
```

→ 把a-z写入参数文件 → open argv  
fputc for

封装  
XF 核心 辅助 周围 声明

# 标准I/O – 小结

fopen

fgetc

fputc



# 标准I/O – 思考和练习

如何利用fgetc / fputc实现文件的复制?

1. 通过命令行参数传递源文件和目标文件名

2. 通过fgetc返回值判断是否读到文件末尾

*fopen s*

*fopen d*

*while fgetc.  
fputc*

*→ 不断读字符, 直到失败*

*同时 写到目标文件*

*核心、辅助、周围, 最重要主要的*

