Make it robust.

厦門大學 XIAMEN UNIVERSITY
信息学院 (特色化示范性软件学院) School of Informatics
黄 炜 博士·副教授 Wei Huang

C语言程序设计

C Programming

8

字符输入输出和输入验证

理论课程

厦門大學

XIAMEN UNIVERSITY

信息学院

(特色化示范性软件学院)

School of Informatics

黄 炜

博士·副教授

Wei Huang

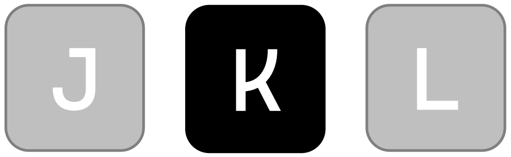
内容要点

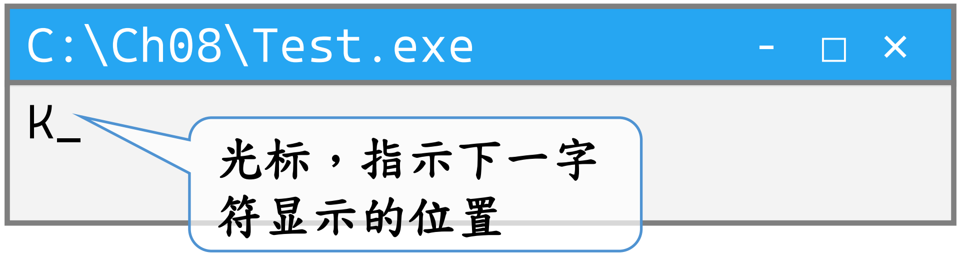
- 输入和输出缓冲
- 创建友好的用户界面

目 录

区分输入显示和输出显示

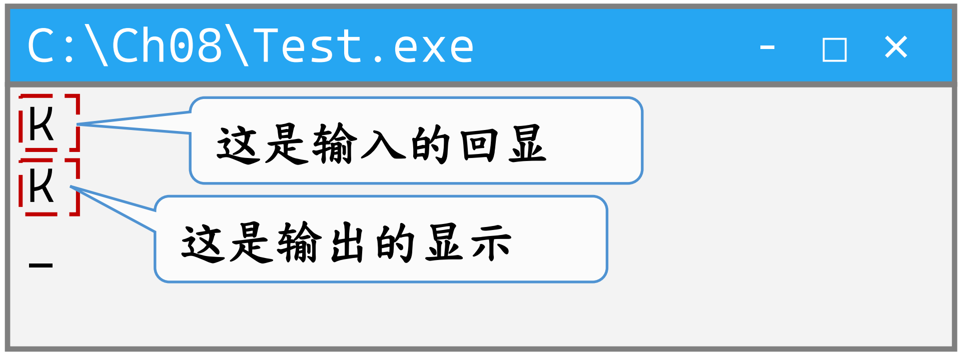
- 输入时的显示
  - 当单击键盘的可见字符时，屏幕将立即显示该字符





- 输出时的显示
  - 当程序向屏幕输出时，屏幕显示该内容

```
printf("K\n");
```



输入输出缓冲

- 缓冲（ buffer ）
  - 缓冲区是临时存储数据的一段内存区域。
  - 缓冲技术用于协调速度相差大的设备间传送数据。
    - 如：CPU和键盘的速度相差很大
- 输入输出缓冲区
  - 输入缓冲区（以键盘为例）
    - 键盘将字符送入缓冲区，程序从缓冲区中读取数据
  - 输出缓冲区（以屏幕为例）
    - 程序将字符送入缓冲区，缓冲区清空时显示在屏幕上

缓冲的分类

- 分类
  - 完全缓冲：缓冲区满时发送至目标，并清空缓冲区
  - 行缓冲：遇到回车时发送至目标，并清空缓冲区
  - 无缓冲：一旦需要读写，立刻发送至目标
- 优缺点
  - 减少调用输入输出的负担（如：数据库）
  - 交互式程序需要非缓冲

```c
/* echo.c -- repeats input */
#include <stdio.h>
int main(void)
{
    char ch;

    while ((ch = getchar()) != '#')
        putchar(ch);

    return 0;
}
```

```
This is my brother.↲
This is my brother.
$#$@$#↲
$
```

缓冲输入

- 输入有缓冲与无缓冲的区别

键盘	输入缓存	getch()返回值（有缓冲）
H	H	暂停运行，等待缓存区清空
i	H,i	暂停运行，等待缓存区清空
!	H,i,!	暂停运行，等待缓存区清空
Enter	H,i,! ,\n	第1次调用返回字符H 第2次调用返回字符i 第3次调用返回字符! 第4次调用返回字符\n
	（空）	程序继续运行
A	A	第5次调用则等待

键盘	getch()返回值（无缓冲）
H	第1次调用返回字符H
i	第2次调用返回字符i
!	第3次调用返回字符!
Enter	第4次调用返回字符\n
A	第5次调用返回字符A

缓冲输出

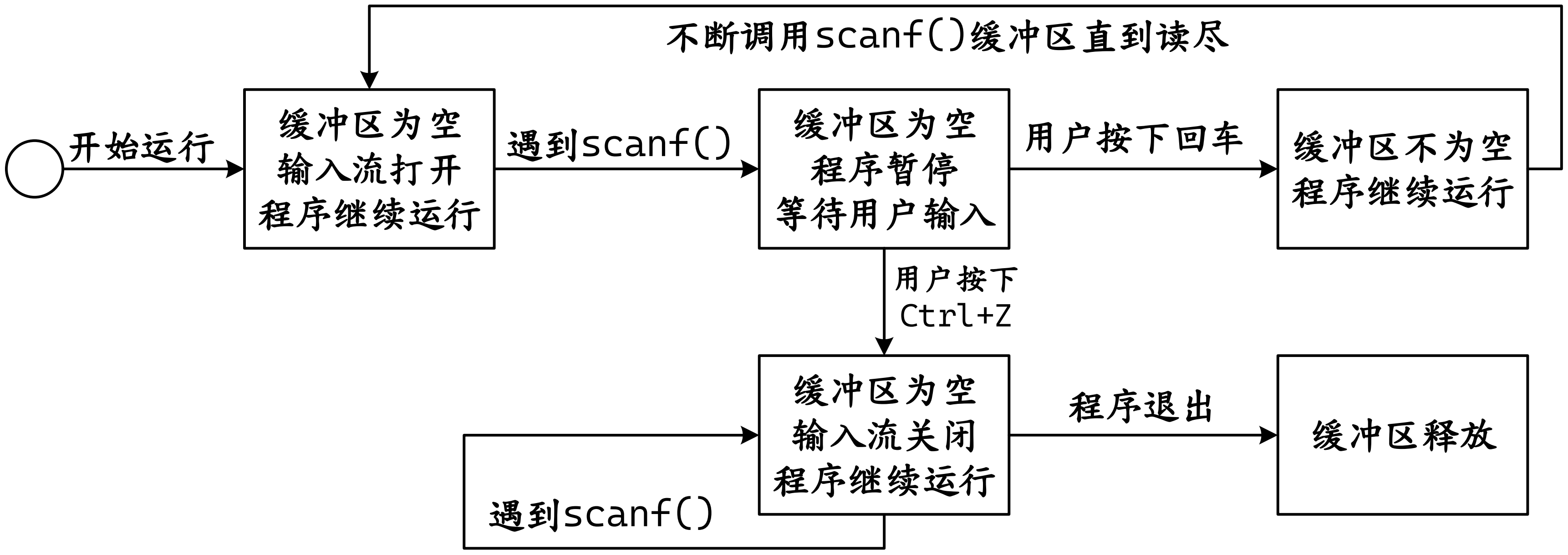
- 输出有缓冲与无缓冲的区别

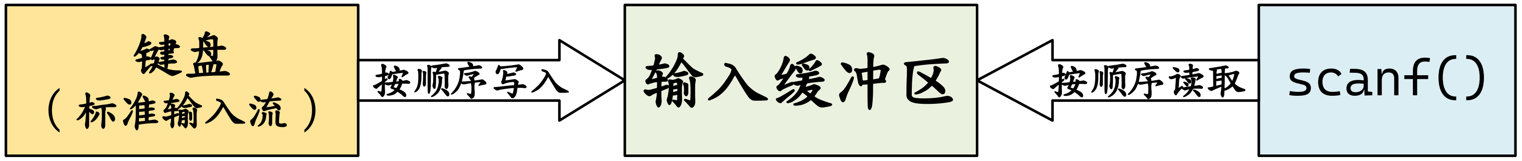
putch()参数值	输出缓存	显示器（行缓冲）
H	H	无显示
i	H,i	无显示
!	H,i,!	无显示
\n	H,i,! ,\n	Hi!↲
	（空）	Hi!↲
A	A	Hi!↲

putch()参数值	显示器（无缓冲）
H	H
i	Hi
!	Hi!
\n	Hi!↲
A	Hi!↲ A

键盘输入的暂停

- 用户按下回车则缓冲区生效
- 缓冲区空则暂停并接受输入

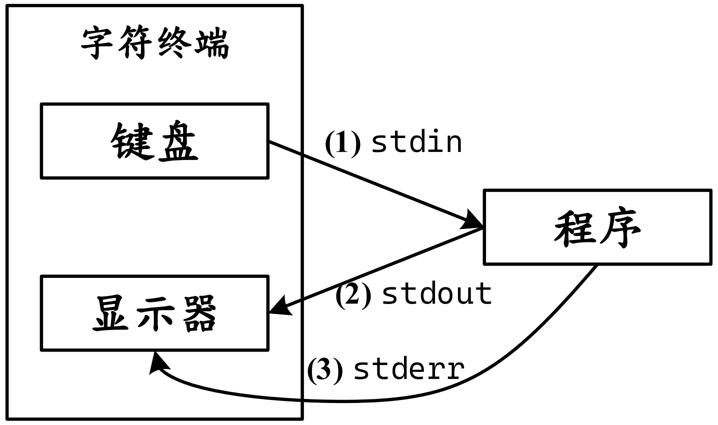

不断调用scanf()缓冲区直到读尽
开始运行
缓冲区为空 输入流打开 程序继续运行
遇到scanf()
缓冲区为空 程序暂停 等待用户输入
用户按下回车
缓冲区不为空 程序继续运行
用户按下 Ctrl+Z
缓冲区为空 输入流关闭 程序继续运行
程序退出
缓冲区释放
遇到scanf()



目录

输入输出流（stream）

- 流是一段时间内可用的数据元素序列。
  - 流的概念掩盖了不同设备的底层差异，提供统一接口
    - 替用户和操作系统底层I/O打交道



常见的设备	标准流	打开 (open)	关闭 (close)	读取 (read)	写入 (write)
屏幕	标准输出流 stdout 标准错误流 stderr	自动打开	不可	不可	可
键盘	标准输入流 stdin	自动打开	不可	可	不可
打印机	打印流	可	可	不可	可
文件	文件流	可	可	可	可
网络	网络流	可	可	可	可

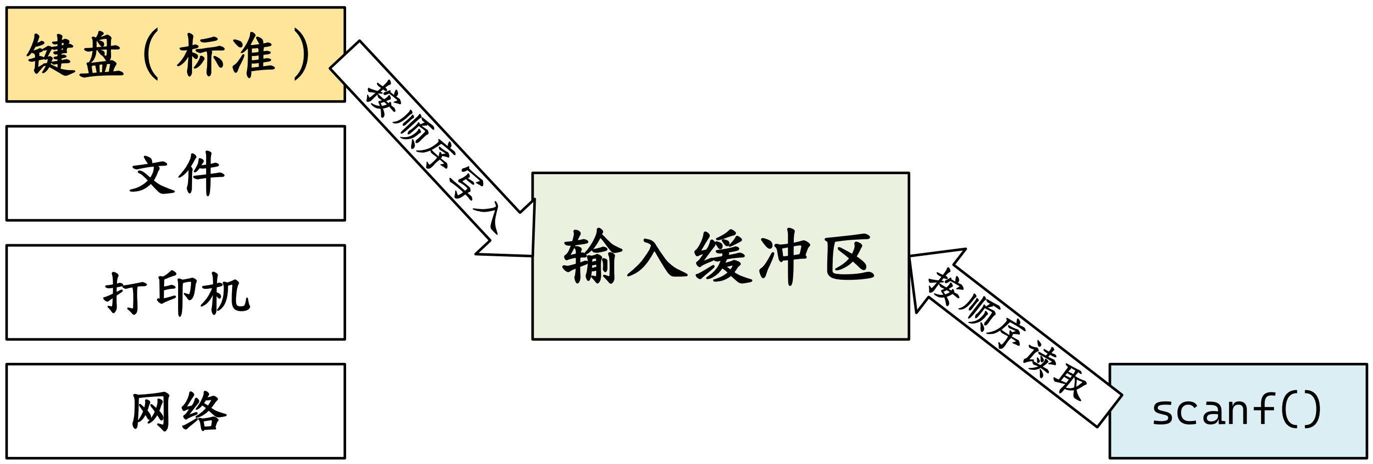
缓冲输入流的行为

- 缓冲区默认为空
- 程序从程序缓冲区中读取数据
  - 已读数据从缓冲区删除
  - 未读取的数据仍存储于缓冲区中
- 程序读取数据而程序缓冲区为空时
  - 补充缓冲（如为键盘需暂停等待）
  - 如果缓冲区用尽或关闭，返回EOF，继续程序
- 程序结束后，缓冲区释放

```
#define EOF (-1)
```

输入流的终止

- 输入流是可以终止的
  - 文件读取到末尾时，终止文件输入
  - 同时按下 Ctrl Z 终止键盘输入
  - 同时按下 Ctrl C 终止程序同时终止键盘输入



```c
/* echo_eof.c -- repeats input to end of file */
#include <stdio.h>

int main(void)
{
    int ch;

    while ((ch = getchar()) != EOF)
        putchar(ch);

    return 0;
}
```

```
This is my joke.↲
This is my joke.
Ctrl + C
```

流的重定向

- 通过命令行将标准流重定向（redirect）到其它流
  - 输入输出可以从键盘或显示器重定向到文件
  - 输出流可以重定向至打印机

重定向的目的	命令行实例
将标准输入流（键盘）重定向到文件	*exe_path < in_file*
将标准输出流（屏幕）到文件	*exe_path > out_file*
同时重定向标准输入和输出流到不同的文件中（命令行中次序先后无关）	*exe_path < in_file > out_file*
	exe_path > out_file < in_file
将标准输出流到文件追加到文件尾	*exe_path >> out_file*
将标准错误流（屏幕）重定向到文件	*exe_path 1> out_file 2> err_file*

流的重定向

freopen

功能	每次向标准输入流写入一个字符，并将该字符返回。	
格式	FILE * freopen(const char * *filename*, const char * mode, FILE * *stream*);	
参数	*filename*	字符串，用于关联到文件流的文件名。
	mode	字符串，新文件的访问模式，可选项为r, w, a, r+, w+, a+（读、写、追加，及其扩展模式）。
	stream	文件指针，需要重定向的流。
返回值	成功	新打开的文件流。
	失败	空（宏NULL）。
头文件	stdio.h	
说明	1. 无。	

流的重定向

- 通过函数将标准流重定向到其它流

重定向的目的	命令行实例
将标准输入流（键盘）定向到文件	`freopen("in.txt", "r", stdin);`
将标准输出流（屏幕）定向到文件	`freopen("out.txt", "w", stdout);`
将标准错误流（屏幕）定向到文件	`freopen("err.txt", "w", stderr);`
将标准错误流定向追加到文件尾	`freopen("err.txt", "a", stderr);`

- 使用场景
  - 在测试程序时，输入输出特别长，重定向能节省时间
  - 读取程序输出时，重定向到文件更方便操作

目 录

创建一个更友好的用户界面

- 软件工程更应该注重的地方
  - 主体功能是基础
  - 软件在用户各种可能的输入下不崩溃也很重要
- 应解决初级用户的输入错误
  - 需要解决多余回车的问题
  - 需要解决不慎按错键的问题

```c
/*****************************************************
* Input the initial velocity, acceleration and time,
* calculates the shift, by the equation
*   s = v0*t + 1.0 / 2 * a*t*t;
* Author:  Wei Huang <whuang@xmu.edu.cn>
* Version: 1.0 [Jun. 11, 2014]
* Version: 2.0 [Oct. 27, 2016]
*****************************************************/
#include <stdio.h> /* Standard I/O header */
double getDouble(const char* message); // prototype
int main(void) {
    double v0 = 0; //initial velocity
    double a = 0;  //acceleration
    double t = 0;  //time
    double s = 0;  //shift
    v0 = getDouble("Enter the Initial Velocity (m/s): ");
    a = getDouble("Enter the Acceleration (m/(s*s)): ");
    t = getDouble("Enter the Time (s): ");
    if (t < 0) { //time cannot be negative
        printf("The time CANNOT be negative.\n");
        return -1; //error code -1: time<0
    }
    s = v0 * t + 1.0 / 2 * a * t * t;    //main equation
    printf("The total shift is %.3lf m.\n", s);
    return 0;     //return code 0: no error
}

/*
    Read input from keyboard and scan as a double variable. If
    users type wrong characters, it will be ignored.
    Parameters:
        message --- Welcome message
    Return:
        a double variable that the user entered.
*/
double getDouble(const char* message) {
    double dbl;
    printf("%s", message);
    while (scanf("%lf", &dbl) == 0) {
        char c;
        while (scanf("%c", &c) == 1 && c != '\n');
        printf("%s\n%s", "Error input.", message);
    }
    return dbl;
}
```

```c
/* guess.c -- an inefficient and faulty number-guesser */
#include <stdio.h>
int main(void) {
    int guess = 1;
    printf("Pick an integer from 1 to 100. I will try to guess ");
    printf("it.\nRespond with a y if my guess is right and with");
    printf("\nan n if it is wrong.\n");
    printf("Uh...is your number %d?\n", guess);
    while (getchar() != 'y') /* get response, compare to y */
        printf("Well, then, is it %d?\n", ++guess);
    printf("I knew I could do it!\n");
    retur
}
```

```
Pick an integer from 1 to 100. I will try to guess it.
Respond with a y if my guess is right and with
an n if it is wrong.
Uh...is your number 1?
n↲
Well, then, is it 2?
Well, then, is it 3?
↲
Well, then, is it 4?
y↲
I knew I could do it!
```

```
/* showchar1.c -- program with a BIG I/O problem */
#include <stdio.h>
void display(char cr, int lines, int width);
int main(void) {
    int ch;              /* character to be printed    */
    int rows, cols;      /* number of rows and columns */
    printf("Enter a character and two integers:\n");
    while ((ch = getchar()) != '\n') {
        scanf("%d %d", &rows, &cols);
        display(ch, rows, cols);
        printf("Enter another character and two integers;\n");
        printf("Enter a newline to quit.\n");
    }
    printf("Bye.\n");
    return 0;
}
```

```c
void display(char cr, int lines, int width) {
    int row, col;

    for (row = 1; row <= lines; row++) {
        for (col = 1; col <= width; col++)
            putchar(cr);
        putchar('\n'); /* end line and start a new one */
    }
}
```

```
Enter a character and two integers:
c 1 4↲
cccc
Enter another character and two integers;
Enter a newline to quit.
Bye.
```

```c
/* showchar2.c -- prints characters in rows and columns */
#include <stdio.h>
void display(char cr, int lines, int width);
int main(void) {
    int ch;             /* character to be printed      */
    int rows, cols;     /* number of rows and columns   */
    printf("Enter a character and two integers:\n");
    while ((ch = getchar()) != '\n') {
        if (scanf("%d %d", &rows, &cols) != 2)
            break;
        display(ch, rows, cols);
        while (getchar() != '\n')
            continue;
        printf("Enter another character and two integers;\n");
        printf("Enter a newline to quit.\n");
    }
    printf("Bye.\n");
    return 0;
}
```

```c
void display(char cr, int lines, int width) {
    int row, col;
    for (row = 1; row <= lines; row++) {
        for (col = 1; col <= width; col++)
            putchar(cr);
        putchar('\n'); /* end line and start a new one */
    }
}
```

```
Enter a character and two integers:
c 1 3↲
ccc
Enter another character and two integers;
Enter a newline to quit.
* 2 3↲
***
***
Enter another character and two integers;
Enter a newline to quit.
↲
Bye.
```

```c
// checking.c -- validating input
#include <stdio.h>
#include <stdbool.h>
// validate that input is an integer
long get_long(void);
// validate that range limits are valid
bool bad_limits(long begin, long end, long low, long high);
// calculate the sum of the squares of the integers a through b
double sum_squares(long a, long b);
int main(void) {
    const long MIN = -10000000L;  // lower limit to range
    const long MAX = +10000000L;  // upper limit to range
    long start;                   // start of range
    long stop;                    // end of range
    double answer;
    printf("This program computes the sum of the squares of "
          "integers in a range.\nThe lower bound should not "
          "be less than -10000000 and\nthe upper bound "
          "should not be more than +10000000.\nEnter the "
          "limits (enter 0 for both limits to quit):\n"
          "lower limit: ");
    start = get_long();
    printf("upper limit: ");
    stop = get_long();
    while (start !=0 || stop != 0) {
        if (bad_limits(start, stop, MIN, MAX))
            printf("Please try again.\n");
        else {
            answer = sum_squares(start, stop);
            printf("The sum of the squares of the integers ");
            printf("from %ld to %ld is %g\n", start, stop, answer);
        }
        printf("Enter the limits (enter 0 for both "
               "limits to quit):\n");
        printf("lower limit: ");
        start = get_long();
        printf("upper limit: ");
        stop = get_long();
    }
    printf("Done.\n");
    return 0;
}
```

```
long get_long(void) {
    long input;
    char ch;
    while (scanf("%ld", &input) != 1) {
        while ((ch = getchar()) != '\n')
            putchar(ch);  // dispose of bad input
        printf(" is not an integer.\nPlease enter an ");
        printf("integer value, such as 25, -178, or 3: ");
    }
    return input;
}
double sum_squares(long a, long b) {
    double total = 0;
    long i;
    for (i = a; i <= b; i++)
        total += (double)i * (double)i;
    return total;
}
```

```c
bool bad_limits(long begin, long end, long low, long high) {
    bool not_good = false;
    if (begin > end) {
        printf("%ld isn't smaller than %ld.\n", begin, end);
        not_good = true;
    }
    if (begin < low || end < low) {
        printf("Values must be %ld or greater.\n", low);
        not_good = true;
    }
    if (begin > high || end > high) {
        printf("Values must be %ld or less.\n", high);
        not_good = true;
    }
    ret
}
```

```
This program computes the sum of the squares of integers in a range.
The lower bound should not be less than -10000000 and
the upper bound should not be more than +10000000.
Enter the limits (enter 0 for both limits to quit):
lower limit: 1973↲
upper limit: 2931↲
The sum of the squares of the integers from 1973 to 2931 is 5.8393e+009
Enter the limits (enter 0 for both limits to quit):
lower limit: 0↲
upper limit: 0↲
Done.
```

菜单浏览

- 显示一个菜单，并接受用户输入一个选项
- 根据选项执行相应操作

```c
/* menuette.c -- menu techniques */
#include <stdio.h>
char get_choice(void);
char get_first(void);
int get_int(void);
void count(void);
int main(void) {
    int choice;
    void count(void);
    while ( (choice = get_choice()) != 'q’) {
        switch (choice) {
            case 'a' :  printf("Buy low, sell high.\n");
                break;
            case 'b' :  putchar('\a');  /* ANSI */
                break;
            case 'c' :  count();
                break;
            default  :  printf("Program error!\n");
                break;
        }
    }
    printf("Bye.\n");
    return 0;
}
void count(void) {
    int n,i;
    printf("Count how far? Enter an integer:\n");
    n = get_int();
    for (i = 1; i <= n; i++)
        printf("%d\n", i);
    while ( getchar() != '\n')
        continue;
}
char get_choice(void) {
    int ch;
    printf("Enter the letter of your choice:\n");
    printf("a. advice          b. bell\n");
    printf("c. count           q. quit\n");
    ch = get_first();
```

```c
    while (  (ch < 'a' || ch > 'c') && ch != 'q’) {
        printf("Please respond with a, b, c, or q.\n");
        ch = get_first();
    }
    return ch;
}
char get_first(void) {
    int ch;
    ch = getchar();
    while (getchar() != '\n')
        continue;
    return ch;
}
int get_int(void) {
    int input;
    char ch;
```

```
Enter the letter of your choice:
a. advice                b. bell
c. count                 q. quit
c↵
Count how far? Enter an integer:
5↵
1
2
3
4
5
Enter the letter of your choice:
a. advice                b. bell
c. count                 q. quit
b↵
Enter the letter of your choice:
a. advice                b. bell
c. count                 q. quit
q↵
Bye.
```

```
    while (scanf("%d", &input) != 1)
    {
        while ((ch = getchar()) != '\n')
            putchar(ch);  // dispose of bad input
        printf(" is not an integer.\nPlease enter an ");
        printf("integer value, such as 25, -178, or 3: ");
    }

    return input;
}
```

目 录

C语言程序设计

C Programming

8

谢谢观看

理论课程

厦門大學
XIAMEN UNIVERSITY

信息学院
(特色化示范性软件学院)
School of Informatics

黄 炜
博士/副教授
Wei Huang