



# 第3章 键盘输入与屏幕输出

## ——getchar()之深入分析



哈尔滨工业大学

苏小红

sxh@hit.edu.cn

# 本节要讨论的主要问题

---

- 使用getchar()输入字符时的需要注意什么问题？



# 使用getchar()输入字符时的怪象

```
#include <stdio.h>
int main()
{
    char ch1, ch2;
    ch1 = getchar();
    printf("ch1=%c\n", ch1);
    ch2 = getchar();
    printf("ch2=%c\n", ch2);
    return 0;
}
```

ab✓  
ch1=a  
ch2=b

a✓  
ch1=a  
ch2=

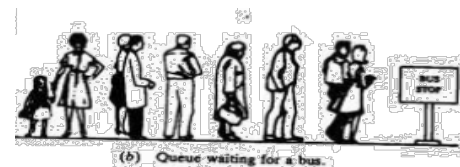
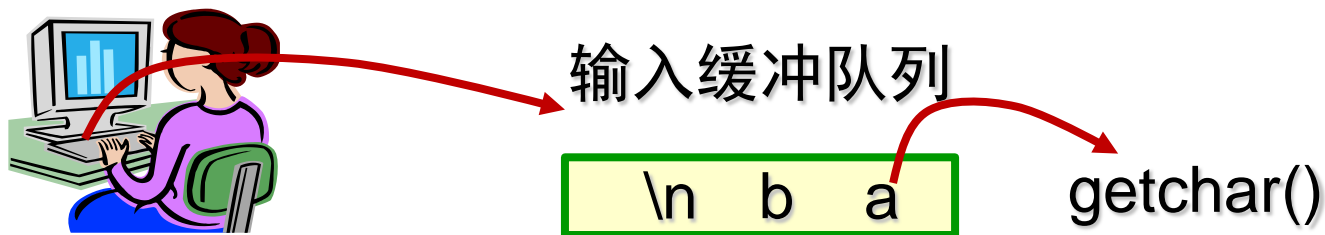


- 以回车符'\n'结束字符的输入
- 输入的字符（包括回车符）都放在输入缓冲区中

# 怪象背后的原因

## ■ 行缓冲（Line-buffer）输入方式

- \* 将输入字符先放入输入缓冲队列中，再从缓冲队列读取字符
- \* 直到键入回车符或文件结束符EOF时，程序才认为输入结束
- \* 一行输入结束，`getchar()`才开始从输入缓冲队列读取字符，前面函数没读走的数据仍在缓冲队列中，将被下一个函数读取



# 怪象背后的原因

```
#include <stdio.h>
int main()
{
    char ch1, ch2;
    ch1 = getchar();
    printf("ch1=%c\n", ch1);
    ch2 = getchar();
    printf("ch2=%d\n", ch2);
    return 0;
}
```

输出第2次读入的  
字符的ASCII码值

a ✓  
ch1=a  
ch2=10

输入缓冲队列

\n a

- 回车符 '\n' 的ASCII码是10

# 使用getchar()输入字符时的怪象

```
#include <stdio.h>
int main()
{
    char ch1, ch2;
    ch1 = getchar();
    printf("ch1=%c\n", ch1);
    ch2 = getchar();
    printf("ch2=%c\n", ch2);
    return 0;
}
```

ab↙  
ch1=a  
ch2=b

a  
ch1=a  
b  
ch2=b

输入缓冲队列

\n b a



- 相当于一次性把键盘输入的一行字符都放入输入缓冲区
- 然后再从输入缓冲区逐个读取字符

## 更深层的原因

- 为什么getchar()要读到一个回车符或文件结束符EOF才进行一次处理操作呢？
- 为什么getchar()以行（而非字符）为单位读取字符呢？
  - \* 实际是按文件的方式读取字符
  - \* 文件一般都是以行为单位的


## 另一个需要注意的问题

### ■ 有时getchar()也可能返回负值

- \* 若在Unix/Linux下遇到组合键Ctrl+D（Windows下为Ctrl+Z），则返回EOF（一般定义为-1）

```
char ch;  
ch = getchar();  
...
```

返回终端输入的字符  
ASCII码值（通常非负）



```
int ch;  
ch = getchar();  
...
```



# 讨论

```
#include <stdio.h>
int main()
{
    char ch1, ch2;
    ch1 = getchar();
    printf("ch1=%c\n", ch1);
    ch2 = getchar();
    printf("ch2=%c\n", ch2);
    return 0;
}
```

a✓  
ch1=a  
b✓  
ch2=b

如何修改这个程序使其  
得到这样的输出结果？





# 解决方法

```
#include <stdio.h>
int main()
{
    char ch1, ch2;
    ch1 = getchar();
    printf("ch1=%c\n", ch1);
    getchar();
    ch2 = getchar();
    printf("ch2=%c\n", ch2);
    return 0;
}
```

读走多余的回车符

a✓  
ch1=a  
b✓  
ch2=b