中南大學
CENTRAL SOUTH UNIVERSITY

第十三讲

字　　符　　串

13、字符串

13.1 字符串字面量

- 一对双引号括起来的字符序列

```
"When you come to a fork in the road, take it."
```

- 转义序列

```
"Candy\nIs dandy\nBut liquor\nIs quicker.\n  --Ogden Nash\n"
```

```
Candy
Is dandy
But liquor
Is quicker.
  --Ogden Nash
```

- 延续字符串字面量

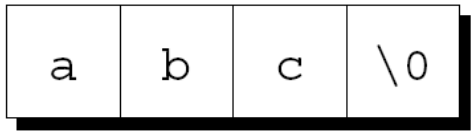
```
printf("Central South \
University");
```

```
printf("Central South"
       "University");
```

13.1 字符串字面量

- 如何存储字符串字面量
  - C语言将字符串字面量作为字符数组处理
  - 长度为n的字符串字面量，分配长度为n+1的空间，‘\0’（空字符）结束

"abc"



空字符-- **'\0'**--ASCII码为0　　零字符-- **'0'**--ASCII码为48

空字符串""

13.1 字符串字面量

- 字符串字面量的操作

```
char *p;

p = "abc";
```

```
char ch;

ch = "abc"[1];
```

```
char digit_to_hex_char(int digit)
{
  return "0123456789ABCDEF"[digit];
}
```

- 字符串字面量与字符常量

"a" 和'a'

```
char *p = "abc";

*p = 'd';    /*** WRONG ***/
```

13.2 字符串变量

- 任何一维字符数组都可以用来存储字符串
- 存储最多80个字符的字符串

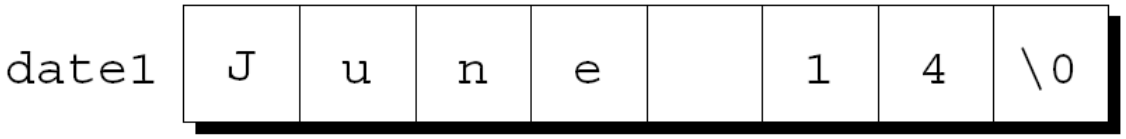
```
#define STR_LEN 80
…
char str[STR_LEN+1];
```

字符串长度达到80时，后面还可以存一个空字符‘\0’

13.2 字符串变量

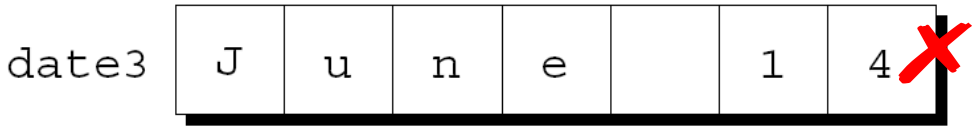
初始化字符串变量

```
char date1[8] = "June 14";
```




```
char date2[9] = "June 14";
```


```
char date3[7] = "June 14";
```




```
char date4[] = "June 14";
```

13.2 字符串变量

字符数组与字符指针

定义与初始化

```
char str[]="China!";
```

```
char *p="China!";
```

赋值操作

```
char str[80];
str="China!";
str=str+3;
```

✗

```
char *p;
p="China!";
p=p+3;
```

✓

元素操作

```
char str[20];
scanf("%s",str);
str[0] = 'J';
```

✓

```
char *p;
scanf("%s",p);
*p = 'J';
```

✗

13.3 字符串的读和写

用printf函数和puts函数写字符串

```
char str[] = "Are we having fun yet?";
printf("%s\n", str);
printf("%.6s\n", str);
```

Are we having fun yet?

Are we

```
char str[]="1234a567";
printf("%s,%5.2s,%c,%c",str,str+2,str[3],str[6]);
```

1234a567, 34,4,6

```
char str1[]="1234\0a567";
printf("%s,%c,%c",str1,str1[3],str1[6]);
```

1234,4,5

```
puts(str);
```

puts函数会加一个额外的换行符

13.3 字符串的读和写

- 用scanf函数和gets函数写字符串

```
scanf("%s", str);
```

- scanf函数会**跳过空白字符**，然后读入字符并存储到str中，直到遇到**空白字符**为止
- gets不会在开始读字符串前跳过空白字符，持续直到找到**换行符**为止。gets不会存最后读到的换行符。

```
char sentence[SENT_LEN+1];

printf("Enter a sentence:\n");
scanf("%s", sentence);

Enter a sentence:       To C, or not to C: that is the question.
```

scanf 存入**"To \0 "**

gets存入整个字符串**"To C, or not to C\0"**

13.3 字符串的读和写

- 逐个字符读入字符串
  - 不会跳过空白字符
  - 在第一个换行符（不存储到字符串中）处停止读取
  - 忽略额外的字符

```
int read_line(char str[], int n){
    int ch, i = 0;
    while ((ch = getchar()) != '\n')
      if (i < n)
          str[i++] = ch;
      str[i] = '\0';/* terminates string */
      return i; /* number of characters stored */
}
```

13.4 访问字符串中的字符

- 可以使用下标来访问字符串中的字符
- 统计字符串中空格的数量

```
int count_spaces(const char s[]){
    int count = 0, i;
    for (i = 0; s[i] != '\0'; i++)
        if (s[i] == ' ')
            count++;
    return count;
}
```

```
int count_spaces(const char *s){
    int count = 0;
    for (; *s != '\0'; s++)
        if (*s == ' ')
            count++;
    return count;
}
```

13.5 使用C语言字符串库

- #include <string.h>
- **strcpy (String Copy)**
  - 将s2中的字符复制到s1中直到遇到s2中第一个空字符为止（空字符也复制）
  - 返回s1，指向目标字符串的指针

```
char *strcpy(char *s1, const char *s2);
strcpy(str2, "abcd");
strcpy(str1, str2);
```

- **strncpy**

```
strncpy(str1, str2, sizeof(str1));
str1[sizeof(str1)-1] = '\0';
```

13.5 使用C语言字符串库

- **strlen** (String Length)
- 返回字符串s第一个空字符之前的字符个数，不含空字符

```
size_t strlen(const char *s);
```

```
int len;

len = strlen("abc");  /* len is now 3 */
len = strlen("");     /* len is now 0 */
strcpy(str1, "abc");
len = strlen(str1);   /* len is now 3 */
```

13.5 使用C语言字符串库

- **strcat** (String Concatenation)
- 将字符串s2的内容追加到s1的末尾，并返回s1

```
char *strcat(char *s1, const char *s2);
```

```
strcpy(str1, "abc");
strcat(str1, "def");
/* str1 now contains "abcdef" */
strcpy(str1, "abc");
strcpy(str2, "def");
strcat(str1, str2);
/* str1 now contains "abcdef" */
```

13.5 使用C语言字符串库

- strcat (String Concatenation)
- 如果s1的长度不够长，可能出现错误

```
char str1[6] = "abc";

strcat(str1, "def");    /*** WRONG ***/
```

- strncat(str1, str2, sizeof(str1) - strlen(str1) - 1);

13.5 使用C语言字符串库

strcmp (String Comparison)

- 比较s1和s2，根据s1是小于、等于或者大于s2，返回小于0，等于0或者大于0的值

```
int strcmp(const char *s1, const char *s2);
```

```
if (strcmp(str1, str2) < 0)/* is str1 < str2? */
```

- s1<s2，当
  - s1,s2的前面i个字符相同，s1的第i+1个字符小于s2的第i+1个字符
  - “abc” < “bcd” “abd” < “abe”
  - si的所有字符跟s2一致，但是s1的长度小于s1
  - “abc” < “abcd”

13.5 使用C语言字符串库

- `strcmp` (String Comparison)
- ASCII码
  - A~Z、a~z、0~9这几组字符的ASCII码是连续的
  - 所有大写字母小于小写字母
  - 数字小于字母
  - 空格符小于所有可以打印的字符

13.5 使用C语言字符串库

- 示例：显示一个月的提醒列表

01--remind.c

```
Enter day and reminder: 24 Susan's birthday
Enter day and reminder: 5 6:00 - Dinner with Marge and Russ
Enter day and reminder: 26 Movie - "Chinatown"
Enter day and reminder: 7 10:30 - Dental appointment
Enter day and reminder: 12 Movie - "Dazed and Confused"
Enter day and reminder: 5 Saturday class
Enter day and reminder: 12 Saturday class
Enter day and reminder: 0
Day Reminder
  5 Saturday class
  5 6:00 - Dinner with Marge and Russ
  7 10:30 - Dental appointment
 12 Saturday class
 12 Movie - "Dazed and Confused"
 24 Susan's birthday
 26 Movie - "Chinatown"
```

13.6 字符串的惯用法

搜索字符串结尾

```
size_t strlen(const char *s){
    size_t n;

    for (n = 0; *s != '\0'; s++)
        n++;
    return n;
}
```

```
size_t strlen(const char *s){
    size_t n = 0;
    for (; *s; s++)
        n++;
    return n;
}
```

```
size_t strlen(const char *s){
    size_t n = 0;
    for (; *s != '\0'; s++)
        n++;
    return n;
}
```

```
size_t strlen(const char *s){
    size_t n = 0;
    for (; *s++;)
        n++;
    return n;
}
```

13.6 字符串的惯用法

- 搜索字符串结尾

```
size_t strlen(const char *s){
    size_t n = 0;
    while (*s++)
        n++;
    return n;
}
```

- 惯用法

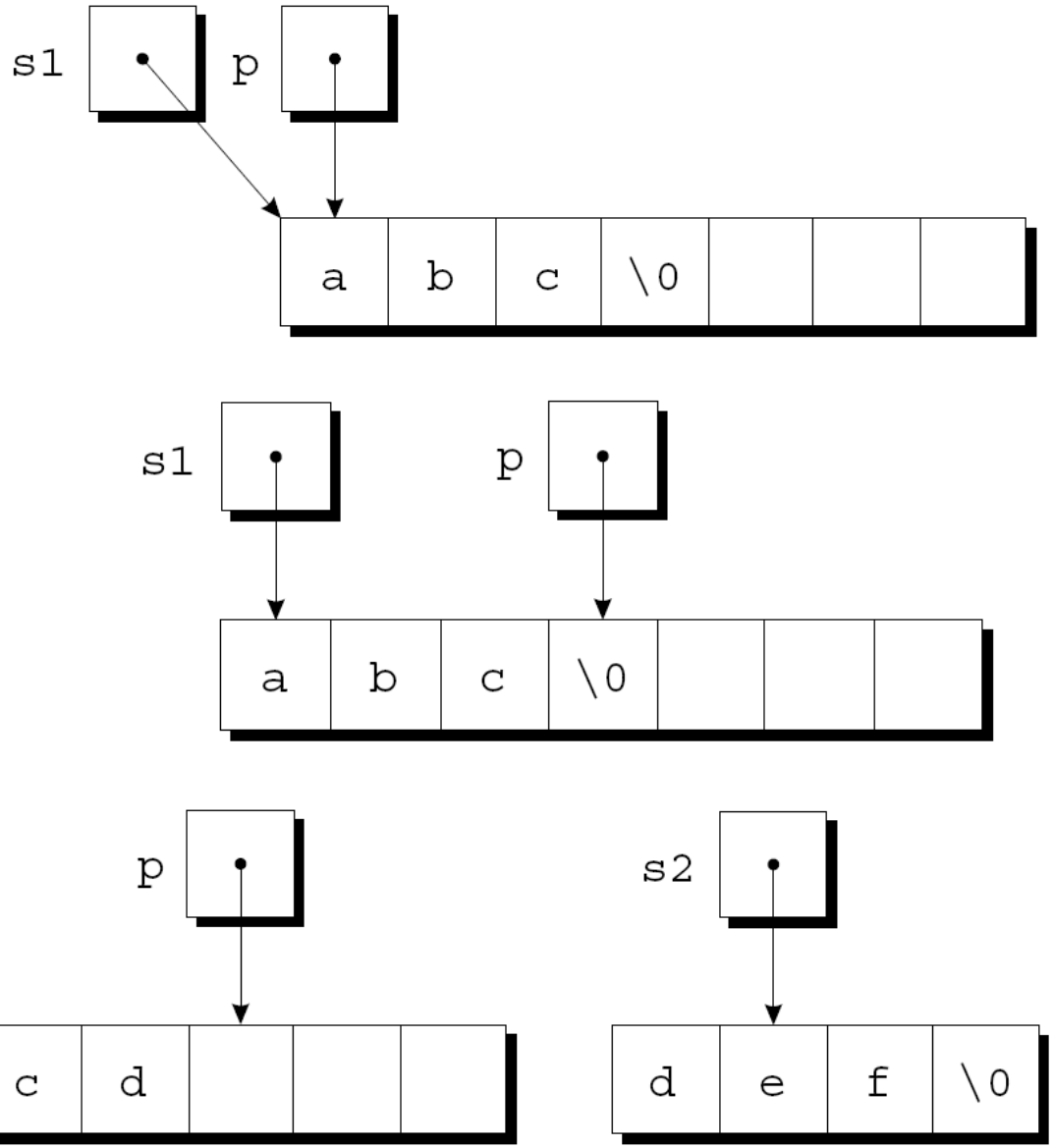
```
while (*s)
    s++;
```

```
while (*s++)
    ;
```

13.6 字符串的惯用法

字符串拼接

```
char *strcat(char *s1, const char *s2){
    char *p = s1;
    while (*p != '\0')
        p++;
    while (*s2 != '\0') {
        *p = *s2;
        p++;
        s2++;
    }
    *p = '\0';
    return s1;
}
```



13.6 字符串的惯用法

- 字符串拼接

```
char *strcat(char *s1, const char *s2) {
        char *p = s1;
        while (*p)
           p++;
        while (*p++ = *s2++)
           ;
        return s1;
}
```

- 惯用法

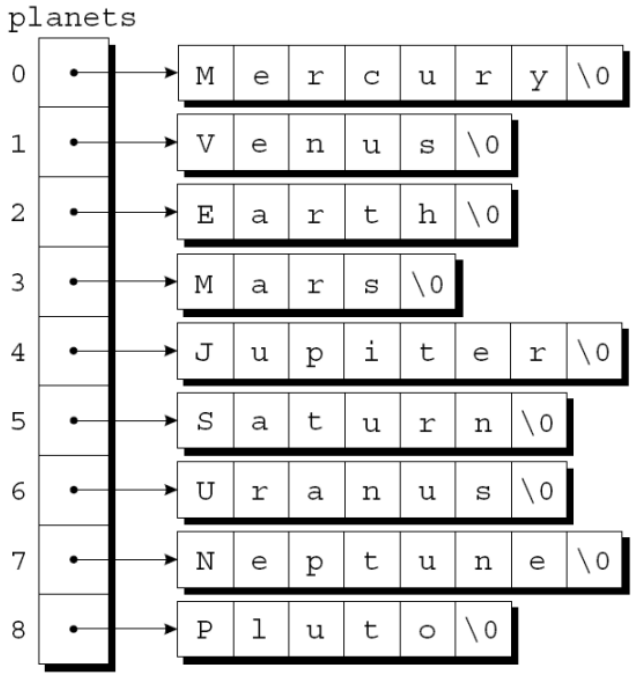
```
while (*p++=*s2++) ;
```

13、字符串

13.6 字符串数组

```
char *planets[] = {"Mercury", "Venus", "Earth",
                   "Mars", "Jupiter", "Saturn",
                   "Uranus", "Neptune", "Pluto"};
```

```
char planets[][8] = {"Mercury", "Venus", "Earth",
                     "Mars", "Jupiter", "Saturn",
                     "Uranus", "Neptune", "Pluto"};
```



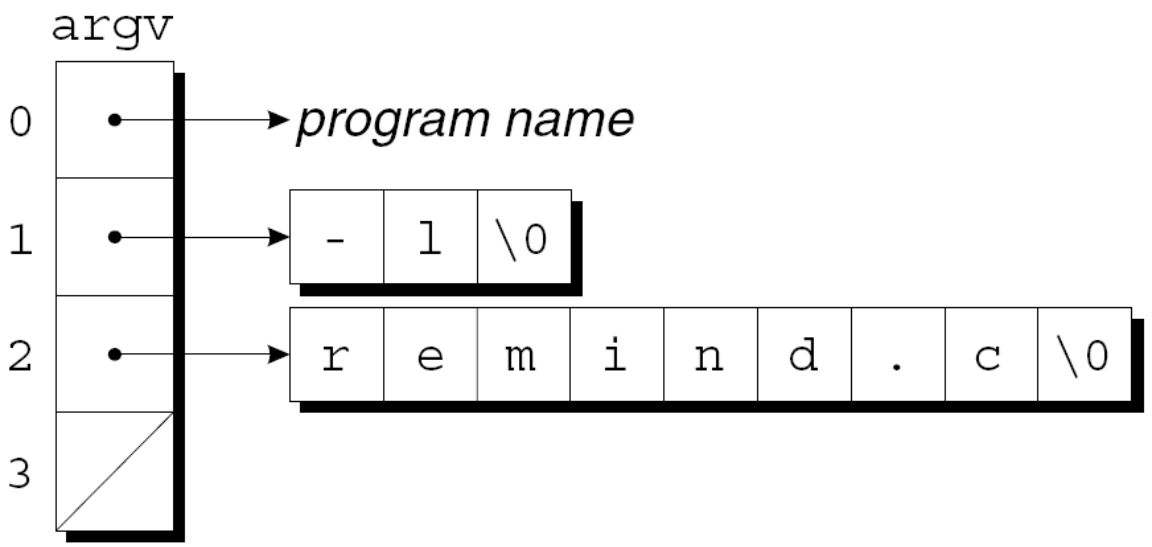
	0	1	2	3	4	5	6	7
0	M	e	r	c	u	r	y	\0
1	V	e	n	u	s	\0	\0	\0
2	E	a	r	t	h	\0	\0	\0
3	M	a	r	s	\0	\0	\0	\0
4	J	u	p	i	t	e	r	\0
5	S	a	t	u	r	n	\0	\0
6	U	r	a	n	u	s	\0	\0
7	N	e	p	t	u	n	e	\0
8	P	l	u	t	o	\0	\0	\0

```
for (i = 0; i < 9; i++)
    if (planets[i][0] == 'M')
        printf("%s begins with M\n", planets[i]);
```

命令行参数

- UNIX的ls命令: ls -1 remind.c
- DOS的copy命令: copy fs fd
- main函数可以有参数的,实参来自操作系统命令行

```
int main(int argc, char *argv[]){
    …
}
```




```
#include<stdio.h>
int main(int  argc, char *argv[]){
    int i;
    for (i = 1;i < argc; i++)
        printf("*%s",argv[i]);
}
```

- 示例
  - 核对行星的名字

02--planet.c

```
planet Jupiter venus Earth fred

Jupiter is planet 5
venus is not a planet
Earth is planet 3
fred is not a planet
```