字符串

2022 年 8 月 16 日

1. 输入一个字符串，将其中的大写字母改为小写字母。

```c
#include <stdio.h>
#include <string.h> // 字符串函数在这里定义
const int MAX_LEN = 1024; // 字符串处理往往先设定一个最大长度
int main()
{
    printf("Input a string:\n");
    char str[MAX_LEN]; // 长度为MAX_LEN的字符数组
    scanf("%s", str);
    int diff_between_upper_and_lower = 'a' - 'A'; // 字母的大写和小写对应的ASCII码的差
    for(int i = 0; i < strlen(str); i++) // strlen(str)表示字符串str的长度
    {
        if(str[i] >= 'A' && str[i] <= 'Z') // str[i]为大写字母
        {
            str[i] = str[i] + diff_between_upper_and_lower; //
                加法，ASCII码往后找，得到相应的小写字母
        }
    }
    printf("The lower case word is: %s.\n", str); // %s表示以字符串的形式输出
    return 0;
}
```

2. 输入二进制正整数，输出对应的十进制正整数。

```c
#include <stdio.h>
#include <string.h> // 字符串函数在这个库里面定义

int main()
```

```
{
    printf("Input a 0-1 string:\n");
    char str[1024]; // 定义足够长的数组
    gets(str); // 输入字符串
    // printf("%s\n", str);
    long long sum = 0; // long long是整数类型，能表达的范围比int要大
    long long coefficient = 1;
    for(int i = strlen(str) - 1; i >= 0; i--) // strlen()表示字符串的长度
    /*strlen(str)-1是最后一个有效字符的下标*/
    {
        if(str[i] == '1')
            sum += coefficient;
        coefficient *= 2; // 每到下一位，系数乘2
    }
    printf("The decimal representation is %lld", sum);
    return 0;
}
```

3. 输入一个字符串，然后程序按逆序输出它。

```
#include <stdio.h>
#include <string.h> // 字符串函数在这个库里面定义
int main()
{
    char str[1024];
    printf("Input a string:\n");
    gets(str);
    for(int i = strlen(str) - 1; i >= 0; --i)
    {
        putchar(str[i]); // putchar()表示输出一个字符
    }
    putchar('\n'); // '\n'表示回车换行
    return 0;
}
```

4. 输入三个字符串，连成一个大字符串，并输出连接后的字符串。

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

```
const int MAX_LEN = 1024;
int main()
{
    char str1[MAX_LEN], str2[MAX_LEN], str3[MAX_LEN]; // 定义3个足够长的字符串
    printf("Input the first string:\n");
    gets(str1);
    printf("Input the second string:\n");
    gets(str2);
    printf("Input the third string:\n");
    gets(str3);

    char sum_str[MAX_LEN];
    strcpy(sum_str, str1); // 复制第一个字符串过去
    strcat(sum_str, str2); // 把第二个字符串连接过去
    strcat(sum_str, str3); // 把第三个字符串连接过去
    printf("The output string: %s\n", sum_str);
    return 0;
}
```

5. 输入一个字符串，将串中大写字母变为小写，小写字母变为大写，其他符号不变。

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

#define MAX_LEN 1024 // 表示把MAX_LEN定义为1024，以后每次遇到MAX_LEN，就相当于遇到1024
const int DIFF_BETWEEN_UPPER_AND_LOWER = 'a' - 'A';

int main()
{
    printf("Please input a string:\n");
    char str[MAX_LEN]; // 相当于char str[1024];
    gets(str);
    for(int i = 0; i < strlen(str); i++)
    {
        if(str[i] >= 'A' && str[i] <= 'Z')
        {
            str[i] += DIFF_BETWEEN_UPPER_AND_LOWER; // ASCII码增加得到小写字母
        }
        else if(str[i] >= 'a' && str[i] <= 'z')
```

```
        {
            str[i] -= DIFF_BETWEEN_UPPER_AND_LOWER; // ASCII码减小得到大写字母
        }
    }
    printf("After alternating cases, the input string has become: %s\n", str);
    return 0;
}
```

6. 输入两个英文单词，输出它们在字典中的先后。

```
#include <stdio.h>
#include <string.h>
const int MAX_LEN = 1024; // 字符串空间的大小，须确保足够大
int main()
/*用户输入两个英文单词，程序返回它们在字典中的先后*/
{
   char s1[MAX_LEN], s2[MAX_LEN]; // 用于存储待比较的两个字符串
   printf("Input two string:\n");
   gets(s1);
   gets(s2);
   int i = 0, j = 0; // 用于定位需要进行比较的字符
   // 对两个单词进行扫描
   while(i < strlen(s1) - 1 && j < strlen(s2) - 1) // 当两个单词均为被扫描至末端
   {
      if(s1[i] < s2[j]) // 字符先则单词先
      {
         printf("%s comes before %s\n", s1, s2);
         return 0;
      }
      else if(s1[i] > s2[j]) // 字符后则单词后
      {
         printf("%s comes before %s\n", s2, s1);
         return 0;
      }
      else // 字符相同，因此去比较各自的下一个字母
      {
         i++; // 准备好下一个字符的位置
         j++; // 同上
      }
```

```
30      }
31
32      // 至少有一个单词被扫描至末端
33
34      if(i != strlen(s1) - 1) // 前一个输入的单词未被扫描至末端
35      {
36          printf("%s comes before %s\n", s2, s1);
37          return 0;
38      }
39
40      if(j != strlen(s2) - 1) // 后一个输入的单词未被扫描至末端
41      {
42          printf("%s comes before %s\n", s1, s2);
43          return 0;
44      }
45
46      // 两单词都扫描至末端
47      printf("%s and %s are equal\n", s1, s2);
48
49      return 0;
50  }
```