Ngôn ngữ lập trình C

Bài 9. Chuỗi ký tự

Soạn bởi: TS. Nguyễn Bá Ngọc

2021

Khái niệm chuỗi ký tự

Trong C chuỗi ký tự được biểu diễn bằng mảng 1 chiều của các ký tự với quy ước phần tử có chỉ số nhỏ nhất có giá trị 0 (ký tự null) là điểm kết thúc chuỗi (không được tính vào độ dài chuỗi).

- Trong phạm vi học phần chúng ta sử dụng biểu diễn ký tự bằng kiểu **char**, và chuỗi ký tự với **mảng kiểu char**
  - (*Tham khảo thêm về các ký tự rộng và chuỗi ký tự rộng*)
- Chúng ta có thể khởi tạo mảng ký tự bằng hằng chuỗi ký tự:
  - char a[] = "Hi"; => sizeof(a)==3; a[0]=='H', a[1]=='i', a[2]=='\0';
- Có thể lưu chuỗi ký tự Unicode được mã hóa UTF-8 trong mảng kiểu char, tuy nhiên số lượng phần tử khác null có thể nhiều hơn số lượng ký tự thực tế:
  - char a[] = "Chào"; => sizeof(a) == 6;
  - *(Tham khảo thêm về UTF-8)*

Mảng phải đủ lớn để lưu chuỗi và ký tự kết thúc chuỗi.

Độ dài chuỗi ký tự

Độ dài chuỗi được định nghĩa bằng số lượng ký tự đứng trước ký tự null.

- Để duyệt các ký tự thuộc chuỗi chúng ta chỉ cần con trỏ tới ký tự đầu tiên.
- Có thể triển khai hàm tính độ dài chuỗi ký tự như sau:

```
size_t my_strlen(const char *s) {
size_t len = 0;
while (*s != 0) { ++s; ++len; }
return len;
}
Ví dụ: char s[] = "ABC"; my_strlen(s) trả về 3.
```

Sao chép chuỗi ký tự

- Có thể định nghĩa hàm sao chép chuỗi ký tự như sau:

```
char *my_strcpy(char *dest, const char *src) {
char *p = dest;
while (*src) {*p = *src; ++p; ++src;}
*p = 0;
return dest;
}
```

- Ví dụ: char s1[20], s2[20] = "Hello world!";
  - s1 = s2; // NOK - Không gán được mảng
  - char *p1 = s2, *p2; // OK - p1 trỏ đến vùng nhớ của s2
  - p2 = p1; // Ok - Gán con trỏ, p2 trỏ tới cùng vị trí của p1
  - my_strcpy(s1, s2); // OK - Sao chép s2 sang s1

So sánh các chuỗi ký tự

- Có thể định nghĩa hàm so sánh chuỗi theo thứ tự bảng mã như sau:

```
int my_strcmp(const char *s1, const char *s2) {
for (;;) {
   if (*s1 > *s2) { return 1; }
   else if (*s1 < *s2) { return -1; }
   else if (*s1 == 0) { break; }
   ++s1; ++s2;
}
return 0;
}
```

- Ví dụ: char *s1 = "Abc", *s2 = "Abd", *s3 = "Abb";
  - s1 < s2; // so sánh con trỏ
  - my_strcmp(s1, s2); - Trả về -1
  - my_strcmp(s1, s3); - Trả về 1

Thư viện string.h

Nguyên mẫu giản lược của 1 số hàm thường gặp:

- size_t strlen (const char *s);
  - Tính độ dài chuỗi s (phiên bản ISO của my_strlen).
- char *strcpy (char *dest, const char *src);
  - Sao chép nội dung chuỗi src sang chuỗi dest và trả về con trỏ tới chuỗi dest (phiên bản ISO của my_strcpy).
- int strcmp (const char *s1, const char *s2);
  - So sánh chuỗi s1 và s2 (phiên bản ISO của my_strcmp).
- char *strcat (char *dest, const char *src);
  - Nối tiếp chuỗi src vào sau chuỗi dest.
- const char *strstr (const char *str, const char *sub);
  - Tìm chuỗi sub trong str và trả về con trỏ tới vị trí bắt đầu của sub trong str nếu tìm thấy, hoặc trả về NULL nếu không.
- const char *strchr (const char *s, int c);
  - Trả về con trỏ tới c trong s nếu có, hoặc NULL nếu không.

Thư viện ctype.h

Nguyên mẫu giản lược của một số hàm xử lý ký tự thường gặp:

- int isascii (int c); -Kiểm tra ký tự thuộc bảng mã ASCII (7-bit)
- int islower(int c); - Kiểm tra ký tự là chữ cái thường
- int isupper(int c); - Kiểm tra ký tự là chữ cái hoa
- int isalpha(int c); - Ký tự là chữ cái thường hoặc hoa
- int isdigit(int c); - Kiểm tra ký tự là chữ số (HCS 10)
- int isalnum(int c); - Kiểm tra ký tự là chữ cái hoặc chữ số
- int isxdigit(int c); - Kiểm tra ký tự là chữ số HCS 16 (Hexa)
- int isgraph(int c); - Ký tự nhìn thấy, ngoại trừ dấu cách
- int isprint(int c); - Ký tự nhìn thấy, bao gồm dấu cách
- int isblank(int c); - Kiểm tra ký tự là khoảng trắng
- int iscntrl(int c); - Kiểm tra ký tự điều khiển
- int tolower(int c); - Trả về ký tự viết thường tương ứng của c
- int toupper(int c); - Trả về ký tự viết hoa tương ứng của c

Nhập&xuất chuỗi ký tự

- Đọc từng ký tự:
  - int getchar(void);
  - scanf("%c", &c);
  - => Tự thiết kế giải thuật đọc chuỗi.
- Đọc chuỗi không chứa khoảng trắng:
  - scanf("%s", s);
- Đọc một dòng:
  - char *fgets(char *s, int n, FILE *stream); // Lưu ký tự '\n'
  - scanf("%[^\n]%*c", s); // Bỏ qua ký tự '\n'
- Xuất chuỗi ký tự:
  - int puts(const char *s); - Xuất chuỗi s và thêm ký tự xuống dòng, trả về EOF nếu phát sinh lỗi hoặc giá trị không âm nếu ngược lại.
  - printf("%s", s); // Xuất chuỗi s theo định dạng.

Ví dụ 9.1. Đọc chuỗi theo từng ký tự

vd9-1.c

```c
#include <stdio.h>

// Đọc 1 dòng nhưng không quá n ký tự và không lưu
// ký tự '\n', trả về độ dài chuỗi kết quả
int my_gets(char *s, int n) {
  int idx = 0, ch;
  while (idx < n - 1 && (ch = getchar()) != EOF) {
    if (ch == '\n') {
      break;
    }
    s[idx++] = ch;
  }
  s[idx] = 0;
  return idx;
}

int main() {
  char s[10];
  int n = my_gets(s, 10);
  printf("Chuỗi đã nhập: %s\nn: %d\n", s, n);
  return 0;
}
```

```
bangoc:$gcc -o prog vd9-1.c
bangoc:$./prog
Hello world!
Chuỗi đã nhập: Hello wor
n: 9
bangoc:$./prog
ĐHBKHN
Chuỗi đã nhập: ĐHBKHN
n: 7
bangoc:$
```

Ví dụ 9.2. Nhập chuỗi & bộ nhớ đệm

vd9-2.c

```c
#include <stdio.h>

int main() {
  char s1[20], s2[20], s3[20];

  // Nhập chuỗi không chứa khoảng trắng
  scanf("%s", s1);
  printf("scanf %%s:  %s\n", s1);
  scanf("%s", s1);
  printf("scanf %%s:  %s\n", s1);

  // Quan sát sát khác biệt nếu không xóa
  // bộ nhớ đệm
  while (getchar() != '\n') ;

  // Nhập chuỗi có chứa khoảng trắng
  scanf("%[^\n]%*c", s2);
  printf("scanf %%[^\\n]%%*c: %s\n", s2);
  fgets(s3, 20, stdin);
  printf("fgets: %s\n", s3);
  return 0;
}
```

```
bangoc:$gcc -o prog vd9-2.c
bangoc:$./prog
Hello world!
scanf %s:  Hello
scanf %s:  world!
Hello world!
scanf %[^\n]%*c: Hello world!
Hello world!
fgets: Hello world!

bangoc:$
```

Xử lý từng ký tự trong chuỗi ký tự

Vòng lặp duyệt các ký tự thuộc chuỗi:

- Sử dụng chỉ số và độ dài:

```
for (int i = 0; i < strlen(s); ++i) {
    /* Các xử lý ký tự s[i] */
}
```

- Sử dụng con trỏ và ký tự null:

```
for (char *p = s; *p != 0; ++p) {
    /* Các xử lý ký tự *p */
}
```

Ví dụ 9-3. Xử lý từng ký tự

vd9-3.c

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

int main() {
  char s[20];
  fgets(s, 20, stdin);
  int cc =0;
  for (char *p = s; *p; ++p) {
    if (isupper(*p)) {
      cc++;
      *p = tolower(*p);
    }
  }
  printf("s = %s\ncc = %d\n", s, cc);
  return 0;
}
```

```
bangoc:$gcc -o prog vd9-3.c
bangoc:$./prog
Hello WOrld!
s = hello world!

cc = 3
bangoc:$
```

Mảng chuỗi ký tự

Chuỗi ký tự là mảng 1 chiều của ký tự, vì vậy có thể biểu diễn mảng chuỗi ký tự như mảng của mảng 1 chiều của ký tự.

- Có thể biểu diễn mảng của các chuỗi ký tự cùng độ dài như mảng 2 chiều của các ký tự, ví dụ:
  - char ss[10][20]; - Có thể sử dụng ss[i] như chuỗi ký tự.
- Có thể lưu mảng chuỗi ký tự với độ dài khác nhau bằng mảng con trỏ char * tới các chuỗi (*cấp phát động*), ví dụ:
  - char *ss[10];
  - ss[i] = malloc(sizeof(char) * n[i]);
- Mảng hằng chuỗi ký tự thường được lưu như mảng con trỏ:
  - const char *digits[] = {"Không", "Một", "Hai", "Ba", "Bốn", "Năm", "Sáu", "Bẩy", "Tám", "Chín"}; - Tương đương với:
  - const char *p0 = "Không", *p1 = "Một" /*, … */;
  - const char *digits[] = {p0, p1, p2, p3, p4, p5, p6, p7, p8, p9};

Chuyển đổi chuỗi thành số và ngược lại

- Các hàm sscanf và sprintf tương tự như scanf và printf nhưng sử dụng chuỗi ký tự để nhập, xuất:
- Có thể sử dụng sscanf để chuyển đổi chuỗi (chứa giá trị số) thành số, ví dụ:
  - char *s = "3.1415"; double d;
  - sscanf(s, "%lf", &d);
- Có thể sử dụng sprintf để chuyển đối số thành chuỗi, ví dụ:
  - double d = 3.1415; char s[20];
  - sprintf(s, "%.3f", d); - s chứa "3.142"

Ví dụ 9.4. Đọc các chữ số của số nguyên

vd9-4.c

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

int main() {
  const char *digits[] = { "Không", "Một", "Hai", "Ba",
    "Bốn", "Năm", "Sáu", "Bẩy", "Tám", "Chín"};
  int n = sizeof(digits)/sizeof(digits[0]), x;
  printf("Nhập 1 số nguyên dương: ");
  scanf("%d", &x);
  char s[20], *p = s;
  sprintf(s, "%d", x);
  while (*p) {
    printf(" %s", digits[*p - '0']);
    ++p;
  }
  printf("\n");
  return 0;
}
```

```
bangoc:$gcc -o prog vd9-4.c
bangoc:$./prog
Nhập 1 số nguyên dương: 1230456879
 Một Hai Ba Không Bốn Năm Sáu Tám Bẩy Chín
bangoc:$
```

Ví dụ 9.5. Tách chuỗi ký tự

vd9-5.c

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

int main() {
  char s[] = "Hello world!";
  s[5] = '\0';

  // ss là mảng con trỏ char *
  char *ss[] = {s, s + 6};
  int n = sizeof(ss)/sizeof(ss[0]);
  for (int i = 0; i < n; ++i) {
    printf("%s\n", ss[i]);
  }
  return 0;
}
```

```
bangoc:$gcc -o prog vd9-5.c
bangoc:$./prog
Hello
world!
bangoc:$
```

Ví dụ 9.6. Xuất n chuỗi ký tự ngược thứ tự đọc

vd9-6.c

```c
#include <stdio.h>

int main() {
  char c[100][100];
  int n;
  printf("Nhập số nguyên dương n <= 100: ");
  scanf("%d", &n);
  printf("Nhập n chuỗi ký tự: \n");
  while (getchar() != '\n') ;
  for (int i = 0; i < n; ++i) {
    printf("Chuỗi %d: ", i);
    scanf("%[^\n]%*c", c[i]);
  }
  printf("Các chuỗi theo thứ tự ngược lại: \n");
  for (int i = n - 1; i >= 0; --i) {
    printf("%s\n", c[i]);
  }
  return 0;
}
```

```
bangoc:$gcc -o prog vd9-6.c
bangoc:$./prog
Nhập số nguyên dương n <= 100: 3
Nhập n chuỗi ký tự:
Chuỗi 0: AAA
Chuỗi 1: BBB
Chuỗi 2: CCC
Các chuỗi theo thứ tự ngược lại:
CCC
BBB
AAA
bangoc:$
```