CPNM Lecture 7 - Strings

Mridul Sankar Barik

Jadavpur University

2022

String Literals

- A string literal is a sequence of characters enclosed within double quotes
- When a C compiler encounter a string literal of length `n` in a program, it sets aside `n+1` bytes of memory for the string
- The null character mark the end of the string
- The null character is a byte whose bits are all zero, so it's represented by the '\0' escape sequence

String Variable

- Any one-dimensional array of charcters can be used to store a string, terminated by a null character

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

- Reading string using `scanf`

  ```
  char str[10];
  scanf("%s", str);
  ```

  - Array name is address of array, so & is not needed
  - `scanf` reads characters until whitespace encountered
  - `scanf` can write beyond end of array (**Be Careful**)

  ```
  scanf("%[^\n]s", str);
  ```

  - Reads characters until a newline is entered

Initializing String Variables

- A string variable can be initialized when it is declared

```
char date[8] = "June 14";
char date[8] = {'J','u','n','e',' ','1','4','\0'};
```

- If the initializer is short, the compiler adds extra null characters
- If the initializer is long, the compiler will omit the null character, making the array unusable as a string

Reading and Writing Strings I

- Writing strings Using `printf` and `puts`
  - `printf` writes the characters in a string one by one until it encounters a null character

    ```
    char str[] = "Hello World";
    printf("%s\n", str);
    ```

  - To print just part of a string, we can use the conversion specification %.ps, where p is the number of characters to be displayed
  - To print first 6 characters of the string str

    ```
    printf ("%.6s\n", str);
    ```

  - The `%ms` conversion specifier will display a string in a field of size `m`. (A string with more than `m` characters will be printed in full, not truncated. If the string has fewer than `m` characters, it will be right-justified within the field. To force left justification put a minus sign in front of `m`.)

Reading and Writing Strings II

  - A conversion specification of the form `%m.ps` causes the first `p` characters of a string to be displayed in a field of size `m`.
  - `puts` has only one argument (the string to be printed). After writing the string, `puts` always writes an additional new-line character

    ```
    puts(str);
    ```

- Reading strings using `gets`
  - `gets` function reads input characters into an array, then stores a null character
- Difference between `scanf` and `gets`
  - `gets` doesn't skip white space before starting to read the string (`scanf` does)
  - `gets` reads until it finds a new line character (`scanf` stops at any whitespace character)

Reading and Writing Strings III

- `scanf` and `gets` do not detect when the array is full. `scanf` can be made safer by using the conversion specification `%ns` instead of `%s`, where `n` is an integer indicating the maximum number of characters to be stored.

Accessing Characters in a String

- Example: To count number of spaces in a string

```
int count = 0, i;
for (i = 0; s[i] != '\0'; i++)
    if (s[i] == ' ')
        count++;
```

Using the C String Library

- C library provides a rich set of functions for performing operations on srings. Prototypes for these functions reside in the <`string.h`> header

String Library Functions

```
char *strcpy(char *dest, const char *src);
char *strncpy(char *dest, const char *src, size_t n);
size_t strlen(const char *s);
char *strcat(char *dest, const char *src);
char *strncat(char *dest, const char *src, size_t n);
int strcmp(const char *s1, const char *s2);
int strncmp(const char *s1, const char *s2, size_t n);
```

Array of Strings

- Using 2D array of char

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

- Ragged array: a 2D array whose rows are of different length

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