MG Thushara
Faculty
Dept. of Computer Science
Amrita School of Engineering
Amritapuri

19CSE102 Computer Programming

Topic-Strings

Strings in C Program

- Strings is a sequence or collection of characters terminated by null character.
- Strings are accessed /created using arrays or pointers.
- To use string manipulation functions **string.h** header file can be used.

Features

of Strings

- C has no native string type, we use char array.
- Null character – '\0' marks the termination of the string whose ascii value is zero.

Declaration

of String

- The general syntax of declaration of String

  char name_of_string[length];

- Here, char is the data type
- name_of_string is the user defined name given to the string variable.
- [length] – defines the size of the string

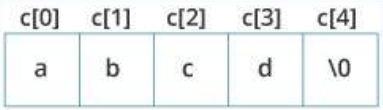
Initializing

Strings

How to initialize strings?

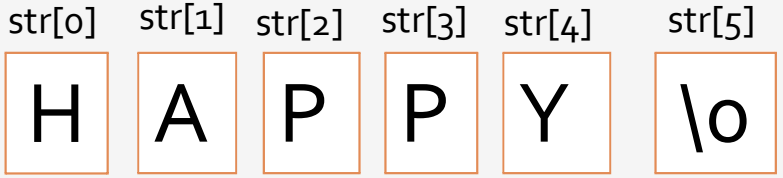
You can initialize strings in a number of ways.

```
char c[] = "abcd";

char c[50] = "abcd";

char c[] = {'a', 'b', 'c', 'd', '\0'};

char c[5] = {'a', 'b', 'c', 'd', '\0'};
```



Example

- char str[10]; //creates a string variable str of size 10.
- char str[ ] ={'H','A','P','P','Y','\0'};



I/O

with Strings

- printf ( ) prints the character up to terminating character.

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

- scanf() reads characters until a whitespace, and stores the result in the string variable and adds a null character automatically to the end of the string.

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

I/O

To read Strings

```c
#include <stdio.h>
int main()
{
    char name[20];
    printf("Enter name: ");
    scanf("%s", name);
    printf("Your name is %s.", name);
    return 0;
}
```

Output

```
Enter name: Dennis Ritchie
Your name is Dennis.
```

Even though `Dennis Ritchie` was entered in the above program, only `"Ritchie"` was stored in the `name` string. It's because there was a space after `Dennis`.

I/O

To read a line of text

```c
#include <stdio.h>
int main()
{
    char name[30];
    printf("Enter name: ");
    fgets(name, sizeof(name), stdin);  // read string
    printf("Name: ");
    puts(name);    // display string
    return 0;
}
```

Output

```
Enter name: Tom Hanks
Name: Tom Hanks
```

Here, we have used `fgets()` function to read a string from the user.

```
fgets(name, sizeof(name), stdin); // read string
```

The `sizeof(name)` results to 30. Hence, we can take a maximum of 30 characters as input which is the size of the `name` string.

To print the string, we have used `puts(name);`.

string.h

Header file to manipulate string

- string.h is the header file that contains many string manipulating functions.
- The functions in string.h have parameters or return values.
- C has a limited string library which are based on null terminated strings.

1. strlen()

string.h functions

int strlen(const char* str)

It computes the length of the string.

```c
#include<stdio.h>
#include<string.h>
int main()
{
    char str[20];
    scanf("%s",str);

    //To find the length of the string using loop
    int i;
    for (i=0;str[i]!='\0';i++);
    printf("\n String length of %s : %d",str,i);

    //To find the length of the string using string function
    printf("\n String length of %s : %I64u",str,strlen(str));

    return 0;
}
```

```
Input String :
Amrita

 String length of Amrita : 6
 String length of Amrita : 6

------------------
(program exited with code: 0)

Press any key to continue . . .
```

2. strcpy( )

string.h functions

strcpy() Function prototype

```
char* strcpy(char* destination, const char* source);
```

The `strcpy()` function copies the string pointed by `source` (including the null character) to the character array `destination`.

The function also returns the copied array.

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

int main()
{
    char str1[10]= "awesome";
    char str2[10];
    char str3[10];

    strcpy(str2, str1);
    strcpy(str3, "well");
    puts(str2);
    puts(str3);

    return 0;
}
```

Output

```
awesome
well
```

3. strcat()

string.h functions

```
char *strcat(char *dest, const char *src)
```

It takes two arguments, i.e, two strings or character arrays, and stores the resultant concatenated string in the first string specified in the argument.

```c
#include <stdio.h>
#include <string.h>
int main()
{
    char str1[] = "This is ", str2[] = "Amrita";

    //concatenates str1 and str2 and resultant string is stored in str1
    strcat(str1,str2);

    puts(str1);
    puts(str2);

    return 0;
}
```

```
This is Amrita
Amrita


------------------
(program exited with code: 0)

Press any key to continue . . .
```

4. strcmp()

string.h functions

```
int strcmp (const char* str1, const char* str2);
```

The `strcmp()` function takes two strings and returns an integer.

The `strcmp()` compares two strings character by character.

If the first character of two strings is equal, the next character of two strings are compared. This continues until the corresponding characters of two strings are

Return Value from strcmp()

Return Value	Remarks
0	if both strings are identical (equal)
negative	if the ASCII value of the first unmatched character is less than second.
positive integer	if the ASCII value of the first unmatched character is greater than second.

4. strcmp()

string.h functions

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

int main()
{
    char str1[] = "btech", str2[] = "btEch", str3[] = "btech";
    int result;

    // comparing strings str1 and str2
    result = strcmp(str1, str2);
    printf("strcmp(str1, str2) = %d\n", result);

    // comparing strings str1 and str3
    result = strcmp(str1, str3);
    printf("strcmp(str1, str3) = %d\n", result);

    return 0;
}
```

```
strcmp(str1, str2) = 1
strcmp(str1, str3) = 0


------------------
(program exited with code: 0)

Press any key to continue . . .
```

Commonly used functions

in string.h

Few commonly used string handling functions are discussed below:

Function	Work of Function
strlen()	computes string's length
strcpy()	copies a string to another
strcat()	concatenates(joins) two strings
strcmp()	compares two strings
strlwr()	converts string to lowercase
strupr()	converts string to uppercase

Thank You