C Programming
String

Dr. Asif Uddin Khan

Character Array and String

- In C programming, a string is a sequence of characters terminated with a null character \0.
- Example: char c[] = "c string";
- When the compiler encounters a sequence of characters enclosed in the double quotation marks, it appends a null character \0 at the end by default.

How to declare a string?

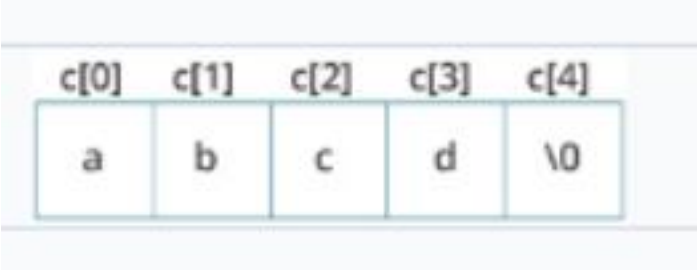
- Here's how you can declare strings:
- char s[5];

How to initialize strings?

How to initialize strings?

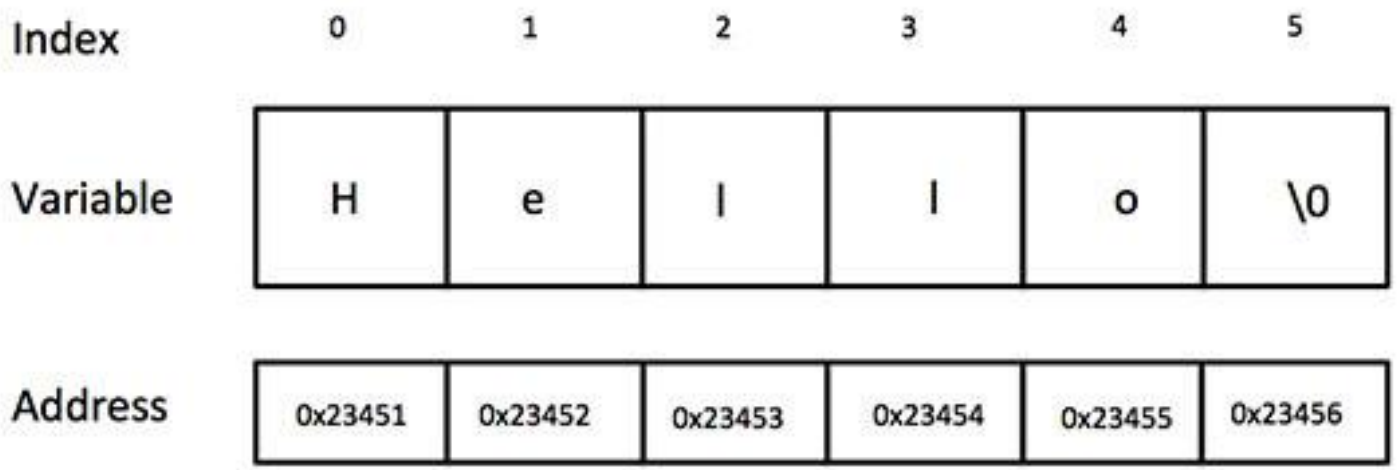
You can initialize strings in a number of ways.

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

char c[50] = "abcd";

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

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



How to initialize strings?

- char greeting[6] = {'H', 'e', 'l', 'l', 'o', '\0'};
- char greeting[] = "Hello";



Example

```
#include <stdio.h>
int main () {
char greeting[6] = {'H', 'e', 'l', 'l', 'o', '\0'};
   printf("Greeting message: %s\n", greeting );
   return 0;
}
```

Output

Greeting message: Hello

Read String from the user

- You can use the scanf() function to read a string.
- The scanf() function reads the sequence of characters until it encounters whitespace (space, newline, tab, etc.).

Example 1: scanf() to read a string

```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.
```

Read String from the user including space

Instead of writing scanf("%s",s), we must write: scanf("%[^\n]s",s) which instructs the compiler to store the string s until the new line (\n) is encountered

```
#include<stdio.h>
void main ()
{
   char s[20];
   printf("Enter the string?");
   scanf("%[^\n]s",s);
   printf("You entered %s",s);
}
```

Example 2: fgets() and puts()

```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;
}
```

Syntax of fgets() function

fgets(charArrayName,CharArraySize,stdin);

Puts() syntax

Puts(CharArrayName);

Output

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

Traversing String

- Using the length of string
- Using the null character

Using the length of string

```
#include<stdio.h>
void main ()
{
   char s[11] = "asif khan";
   int i = 0;
   int count = 0;
   while(i<11)
   {
      if(s[i]=='a' || s[i] == 'e' || s[i] == 'i' || s[i] == 'u' || s[i] == 'o')
      {
         count ++;
      }
      i++;
   }
   printf("The number of vowels %d",count);
}
```

Using the null character

```
#include<stdio.h>
void main ()
{
   char s[11] = "asif khan";
   int i = 0;
   int count = 0;
   while(s[i] != '\0')
   {
      if(s[i]=='a' || s[i] == 'e' || s[i] == 'i' || s[i] == 'u' || s[i] == 'o')
      {
         count ++;
      }
      i++;
   }
   printf("The number of vowels %d",count);
}
```

Passing Strings to Functions

- Strings can be passed to a function in a similar way as arrays

Example 3: Passing string to a Function

```c
#include <stdio.h>
void displayString(char str[]);

int main()
{
    char str[50];
    printf("Enter string: ");
    fgets(str, sizeof(str), stdin);
    displayString(str);     // Passing string to a function.
    return 0;
}
void displayString(char str[])
{
    printf("String Output: ");
    puts(str);
}
```

C Program – finding length of a String without using standard library function

```c
#include <stdio.h>
int main()
{
    char str[100],i;
    printf("Enter a string: \n");
    scanf("%s",str);

    // '\0' represents end of String
    for(i=0; str[i]!='\0'; ++i);
       printf("\nLength of input string: %d",i);

    return 0;
}
```

Copy Strings Without Using strcpy()

Copy String Without Using strcpy()

```c
#include <stdio.h>
int main() {
    char s1[100], s2[100], i;
    printf("Enter string s1: ");
    fgets(s1, sizeof(s1), stdin);

    for (i = 0; s1[i] != '\0'; ++i) {
        s2[i] = s1[i];
    }

    s2[i] = '\0';
    printf("String s2: %s", s2);
    return 0;
}
```

Output

```
Enter string s1: Hey fellow programmer.
String s2: Hey fellow programmer.
```

Concatenate Two Strings Without Using strcat()

```c
// string concatenation without using library function
#include <stdio.h>
int main() {
  char s1[100] = "programming ", s2[] = "is awesome";
  int length, j;
  // store length of s1 in the length variable
  length = 0;
  while (s1[length] != '\0') {
    ++length;
  }
  // concatenate s2 to s1
  for (j = 0; s2[j] != '\0'; ++j, ++length) {
    s1[length] = s2[j];
  }
  // terminating the s1 string
  s1[length] = '\0';

  printf("After concatenation: ");
  puts(s1);
  return 0;
}
```

Commonly Used predefined String Functions

- strlen() - calculates the length of a string
- strcpy() - copies a string to another
- strcmp() - compares two strings
- strcat() - concatenates two strings

- Header file :
- #include <string.h>

strlen()

- **strlen(s1);**
- Returns the length of string s1.

Example: C strlen() function

```c
#include <stdio.h>
#include <string.h>
int main()
{
    char a[20]="Program";
    char b[20]={'P','r','o','g','r','a','m','\0'};

    // using the %zu format specifier to print size_t
    printf("Length of string a = %zu \n",strlen(a));
    printf("Length of string b = %zu \n",strlen(b));

    return 0;
}
```

Output

```
Length of string a = 7
Length of string b = 7
```

strcpy()

- **strcpy(s1, s2);**
- Copies string s2 into string s1.

Example: C strcpy()

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

int main() {
    char str1[20] = "C programming";
    char str2[20];

    // copying str1 to str2
    strcpy(str2, str1);

    puts(str2); // C programming

    return 0;
}
```

Output

```
C programming
```

strcmp()

- **strcmp(s1, s2);**
- Returns 0 if s1 and s2 are the same; less than 0 if s1<s2; greater than 0 if s1>s2.

Example: C strcmp() function

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

int main()
{
    char str1[] = "abcd", str2[] = "abCd", str3[] = "abcd";
    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;
}
```

Output

```
strcmp(str1, str2) = 32
strcmp(str1, str3) = 0
```

strcat()

- **strcat(s1, s2);**
- Concatenates string s2 onto the end of string s1.

Example: C strcat() function

```c
#include <stdio.h>
#include <string.h>
int main() {
   char str1[100] = "This is ", str2[] = "programiz.com";

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

   puts(str1);
   puts(str2);

   return 0;
}
```

Complete Example

```
#include <stdio.h>
#include <string.h>
int main () {
   char str1[12] = "Hello";
   char str2[12] = "World";
   char str3[12];
   int  len ;
   /* copy str1 into str3 */
   strcpy(str3, str1);
   printf("strcpy( str3, str1) :  %s\n", str3 );
   /* concatenates str1 and str2 */
   strcat( str1, str2);
   printf("strcat( str1, str2):   %s\n", str1 );
   /* total lenghth of str1 after concatenation */
   len = strlen(str1);
   printf("strlen(str1) :  %d\n", len );
   return 0;
}
```

References

1. C programming by E Balaguruswami
2. Programming C by Y. kanitkar
3. Programming C by Denis Ritchie
4. NPTEL Lecture note of Dr. Partha Pratim Das, Department of Computer Science and Engineering, Indian Institute of Technology, Kharagpur.
5. https://docs.oracle.com/cd/E18752_01/html/817-6223/chp-typeopexpr-2.html
6. https://data-flair.training/blogs/escape-sequence-in-c/
7. Internet source

References

1. NPTEL Lecture note of Dr. Partha Pratim Das, Department of Computer Science and Engineering, Indian Institute of Technology, Kharagpur.
2. https://docs.oracle.com/cd/E18752_01/html/817-6223/chp-typeopexpr-2.html
3. https://data-flair.training/blogs/escape-sequence-in-c/
4. Internet source