"Mircea cel Batran" Naval Academy Scientific Bulletin, Volume XX – 2017 – Issue 2
The journal is indexed in: PROQUEST / DOAJ / Crossref / EBSCOhost/ INDEX COPERNICUS/ OAJI / DRJI / JOURNAL INDEX / I2OR / SCIENCE LIBRARY INDEX / Google Scholar / Academic Keys / ROAD Open Access / Academic Resources / Scientific Indexing Services / SCIPIO/ JIFACTOR

MATLAB FUNCTION FOR COMPARING TWO STRINGS

Paul VASILIU[1]
[1]Lecturer Ph.D. Eng. Department of Electrical Engineering and Electronics, „MirceacelBătrân" Naval Academy, No.1 Fulgerului Street, Constanţa, Romania
p_vasiliu@yahoo.com

Abstract:*Strings play an important role in the programming field. The programming languages offer to the programmers many functions to operate on strings. An important operation is the comparison of the strings. Matlab offers a set of functions for elementary operations with strings like: strcmp, strcmpi, strncmp and strncmpi. All these functions test whether two strings are identical or not. They do not offer information about the order in which the strings are compared relative to the ASCII codes order of the characters. In the C language there are defined the following functions: strcmp, stricmp, strncmp and strncimp that test the order of two strings according to the ASCII codes of the characters. In this paper, the author presents an implementation in Matlab of a function that produces that same comparison results as the strcmp function in the C language.*

Keywords:*comparison, matlab, programming, string*

INTRODUCTION

C Language offers to the developers a set of specialized functions for strings operations. The signature of these functions are defined in string.h file. One of these functions is strcmp and has as signature: intstrcmp (const char*, const char*). The function has two input arguments which are the addresses of the s1 and s2 strings. The function compares the ASCII codes of the s1 and s2 strings and returns a negative value if s1 < s2, 0 if s1 == s2 and a positive value if s1 > s2. This function introduces an order relationship over the set of strings considering the ASCII codes of the characters. The function differentiates between the ASCII codes of the lower case and upper case characters.
The following program illustrates the usage of the strcmp function in C language and highlights the results generated by the strcmp function.

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

// Function for allocating the array
char *aloc(int n)
{
char *s;
s=(char *)malloc(n*sizeof(char));
return s;
}

// The main function
int main()
{
char *s1,*s2;
if((s1=aloc(100))!=NULL

&&(s2=aloc(100))!=NULL)
{
printf(" Successful allocation \n\n");
printf(" s1 : ");
gets(s1);
printf(" s2 : ");
gets(s2);
if(strcmp(s1,s2)<0)
printf(" %s < %s \n",s1,s2);
if(strcmp(s1,s2)==0)
printf(" %s == %s \n",s1,s2);
if(strcmp(s1,s2)>0)
printf(" %s > %s \n",s1,s2);
}
else
printf(" Allocation error \n");
getch();
}
```

Below there are a couple of examples for running the program:

```
Successful allocation

s1 :abac
s2 :abAc
abac>abAc

Successful allocation

s1 :abac
s2 :abac
abac == abac

Successful allocation
```


DOI: 10.21279/1454-864X-17-I2-018

```
s1 :abAc
s2 :abac
abAc<abac
```

Matlab offers to the user a set of functions specialized in string operations. These functions can be visualized with the following command: >>help matlab\strfun. The function with signature val=strcmp(s1,s2) from section "String operation", compares the strings s1 and s2 and returns logical 1(true) if they are identical, and returns logical 0 (false) otherwise. The function does not compare the ASCII codes of the s1 and s2 strings in the same way the strcmp function from C language does in the string.h file. The Matlab function does not consider sorting the set of strings by ASCII codes. The function distinguishes between the ASCII codesof the lower and upper characters. The next script highlights how thestrcmp function works in Matlab.

```
functionstrcmp_work
s1=input('String 1 : ','s');
s2=input('String 2 : ','s');
val=strcmp(s1,s2);
message=['The returned value by strcmp is: ',...
num2str(val)];
disp(message);
ifval==1
mesaj=[s1,' == ',s2];
disp(message);
else
mesaj=[s1,' ~= ',s2];
disp(message);
end
end
```

Below there are a couple of input variations for running the script:

```
>>strcmp_work
String 1 :abac
String 2 :abAc
abac
abAc
The returned value by strcmpis : 0
abac ~= abAc

>>strcmp_work
String 1 :abac
String 2 :abac
abac
abac
The returned value by strcmp is: 1
abac == abac

>>strcmp_work
String 1 :abAc
String 2 :abac
abAc
abac
The returned value by strcmp is: 0
abAc ~= abac
```

Considering the previous results, the Matlab function strcmp cannot sort the set of string the way strcmp does it in the C language. There is a high necessity to have a Matlab function that orders a set of strings based on the their ASCII codes.

In this paper, the author presents a Matlab function that has the same behavior as the strcmp function from the C language library.

ALGORITHM

Let us consider s1 and s2 the strings that we will compare, and n1 and n2 will be the lengths of these strings. There are two cases: first case is when the strings have equal lengths, and the second case is when the strings have different lengths.

In the first case, we read the strings and compare char by char and count how many characters are equal. If this number equals the length of the two strings then the strings are identical. For the second case, we compute the n=min(n1,n2). We compare the first n characters from both strings and count how many of them match. Depending on this number, we conclude that either s1<s2 or s1>s2. The pseudo code of the comparison algorithm can be found below:

```
% Case n1==n2
if n1==n2
then
assign found←0
fori=1 to n1
do
if s1(i)==s2(i)
then
assign found←found+1
endif
endfor
if found==n1
then
assignval←0
endif
endif

% Case n1 ~=n2
assign n←min(n1,n2)
assigni←1
while s1(i)==s2(i) &i<n
do
assigni←i+1
endwhile
if n==n1 &i==n
then
assignval ← -1
```


DOI: 10.21279/1454-864X-17-I2-018

```
endif
if n==n2 &i==n
then
assignval←1
endif
if s1(i)<s2(i)
then
assignval←-1
else
assignval← 1
endif
endif
```

IMPLEMENTATION

The implementation of the algorithm is the following:

```
% Compare s1 and s2 strings
% val=-1 if s1<s2
% val=0 if s1==s2
% val=1 if s1>s2

functionval=strcmp_pv(s1,s2)
n1=length(s1);
n2=length(s2);
if n1==n2
found=0;
fori=1:n1
if s1(i)==s2(i)
found=found+1;
end
end
if found==n1
message=['String',s1,' is identical to string',s2];
disp(message);
val=0;
return;
end
end
n=min([n1 n2]);
i=1;
while s1(i)==s2(i) &i<n
i=i+1;
end
if n==n1 &i==n
message=[s1,' < ',s2];
disp(message);
val=-1;
return;
end
if n==n2 &i==n
message=[s1,' > ',s2];
disp(message);
val=1;
return;
end
if s1(i)<s2(i)
message=[s1,' < ',s2];
disp(message);
val=-1;
else
message=[s1,' > ',s2];
disp(message);
val=1;
end
end
```

Input examples for running the script are the following:

```
>> s1='abac'
>> s2='abAc'
>>strcmp_pv(s1,s2)
abac>abAc

ans =

1

>>strcmp_pv(s2,s1)
abAc<abac

ans =

   -1

>>strcmp_pv(s1,s1)
String abac is identical to string abac

ans =

    0

>> s2='abacabac'
>>strcmp_pv(s1,s2)
abac<abacabac

ans =

   -1
```

CONCLUSIONS

In this paper the author presented a new Matlab function named strcmp_pv that produces the same results as the strcmp function from the C language library.

DOI: 10.21279/1454-864X-17-I2-018

From the large spectrum of future work we will enumerate:

- Writing a Matlab function similar to stricmp from the C language library;
- Writing a Matlab function similar to strncmp from the C language library;
- Writing a Matlab function similar to strnicmp from the C language library.

BIBLIOGRAPHY

[1] Băutu A., Vasiliu P., Bazele programării calculatoarelor, Ed. A.N.M.B., Constanţa 2009.
[2] Kernigham B., Ritchie D., The C programming language, Prentice Hall,1975.
[3] Matlab, The MATH WORKS Inc., Natick, Massachusetts, 1992.
[4] Vasiliu P., Programare în Matlab, Ed. ANMB, Constanţa 2015.
[5] Vasiliu P., Băutu A. Programarea calculatoarelor în limbajul C, Ed. Europolis, Constanţa, 2006.

DOI: 10.21279/1454-864X-17-I2-018