Κεφάλαιο 9.3-9.4

Αλφαριθμητικές Σειρές Χαρακτήρων (Strings) II

(Διάλεξη 20)

Διδάσκων: Δημήτρης Ζεϊναλιπούρ

Strings στη C

- Ένα string είναι μία ακολουθία αλφαριθμητικών **χαρακτήρων**, σημείων στίξης κτλ.
- Π.χ. “Hello” “How are you?” “121212” “*Apple#123*%”

Προηγούμενη Διάλεξη

1. Εισαγωγικές Έννοιες (Αρχικοποίηση, Ανάγνωση & Εκτύπωση)
2. Πίνακες Strings
3. Συναρτήσεις Βιβλιοθήκης <string.h>

Σήμερα

Υλοποίηση Συναρτήσεων <string.h>

Η Βιβλιοθήκη string.h - Επανάληψη

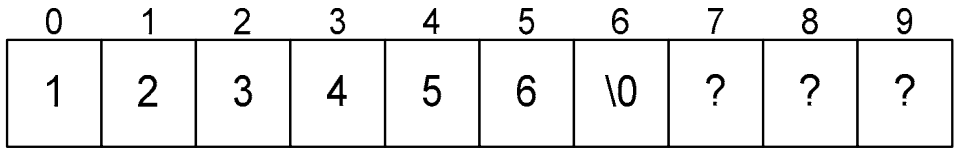
- Το αρχείο επικεφαλίδα (header file), string.h παρέχει συναρτήσεις για χειρισμό strings
- Περιέχει Διάφορες Συναρτήσεις, πχ.
  1. **strlen(s),** υπολογίζει το μέγεθος του string
  2. **strcpy(s1,s2),** αντιγράφει το s2 στο s1
  3. **strcat(s1,s2),** προσθέτει το s2 στο s1.
  4. **strcmp(s1,s2),** συγκρίνει το s1 με s2 και επιστρέφει θετική τιμή εάν s1 μεγαλύτερο (αλφαβητικά) από το s2, μηδέν αν είναι ίσα, και αρνητική τιμή εάν s1 μικρότερο από s2.

Η συνάρτηση strlen() - Επανάληψη

```
#include <stdio.h>

int mystrlen (char s[])
{
   int i=0;
   while (s[i] != '\0')
         i++;
   return i;
}

int main()
{
      char x[10] = "123456";
      printf("%d", mystrlen(x));
}
```



Η mystrlen επιστρέφει 6

1) Η συνάρτηση strcpy()

- Η συνάρτηση strcpy(ma, mb) αντιγράφει το mb στο ma

  π.χ.

```
int main()
{
    char ma[10]
    char mb[10]="HELLO";;
    strcpy(ma,mb);
    printf("ma=%s and mb=%s", ma, mb);
}
```

Πρίν

	0	1	2	3	4	5	6	7	8	9
ma	?	?	?	?	?	?	?	?	?	?
mb	H	E	L	L	O	\0	?	?	?	?

Μετά

	0	1	2	3	4	5	6	7	8	9
ma	H	E	L	L	O	\0	?	?	?	?
mb	H	E	L	L	O	\0	?	?	?	?

1) Υλοποίηση της strcpy()

- Υλοποιήστε την συνάρτηση

  mystrcpy(char to[], char from[])

  η οποία αντιγράφει το String b στο String a

Αλγόριθμος

Για κάθε στοιχείο from[i] αντίγραψε το from[i] στην θέση to[i], μέχρι να φτάσεις στο \0.

1) Υλοποίηση της strcpy()

```
#include <stdio.h>
/* To string ‘from’ αντιγράφεται στο ‘to’ */
void mystrcpy(char to[ ], char from[ ]) {
   int i=0;
   while (from[i] != '\0') {
            to[i] = from[i];
            i++;
    }
    to[i]='\0';
}

int main() {
            char ma[10];  char mb[10]="HELLO";
            mystrcpy(ma,mb);
            printf("ma=%s and mb=%s", ma, mb);
}
```

Πρίν

	0	1	2	3	4	5	6	7	8	9
to	?	?	?	?	?	?	?	?	?	?
from	H	E	L	L	O	\0	?	?	?	?

Μετά

	0	1	2	3	4	5	6	7	8	9
to	H	E	L	L	O	\0	?	?	?	?
from	H	E	L	L	O	\0	?	?	?	?

2) Η συνάρτηση strcat()

- Η συνάρτηση strcat(s1, s2) αντιγράφει το s2 στο τέλος του s1

  π.χ.

Πρίν

	0	1	2	3	4	5	6	7	8	9
s1	H	E	L	L	O	\0	?	?	?	?
s2	C	A	T	\0	?	?	?	?	?	?

Μετά

	0	1	2	3	4	5	6	7	8	9
s1	H	E	L	L	O	C	A	T	\0	?
s2	C	A	T	\0	?	?	?	?	?	?

```
int main()
{
    char ma[10]="HELLO";
    char mb[10]="CAT";
    strcat(ma,mb);
    printf("ma=%s and mb=%s", ma, mb);
}
```

2) Υλοποίηση strcat()

- Υλοποιήστε την συνάρτηση

  void mystrcat(char s1[], char s2[])

  η οποία προσθέτει το string s2 στο τέλος του s1

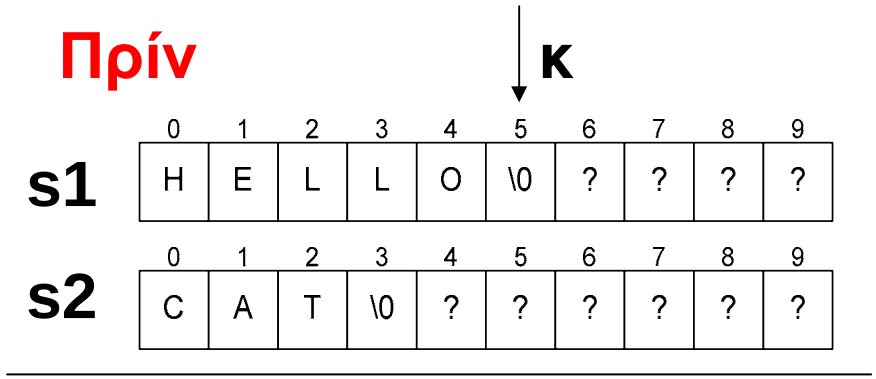
Αλγόριθμος

- Βρες το \0 στο s1 στην θέση X.
- Αντίγραψε κάθε s2[i], πέρα από το X, στην αντίστοιχη θέση s1[i].
- Πρόσθεσε \0 στο τέλος του s1.

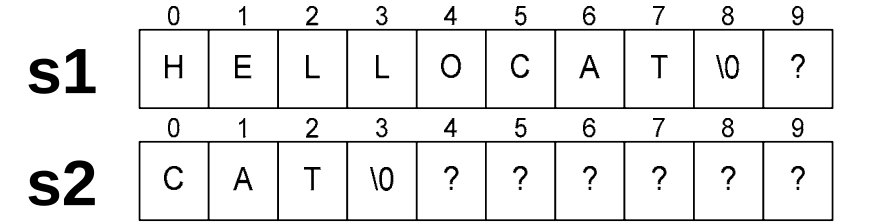
2) Υλοποίηση strcat() – έκδοση 1

```
void mystrcat(char s1[], char s2[])
{
    int i=0, k=0;
    // Εύρεση \0 στο S1
    while (s1[k] != '\0') {
          k++;
    }
    // Αντιγραφή Στοιχείων
    while (s2[i] != '\0') {
          s1[k]=s2[i];
          i++;
          k++;
    }
    s1[k]='\0';  // Προσθήκη NULL στο τέλος του s1
}
```

Πρίν



Μετά

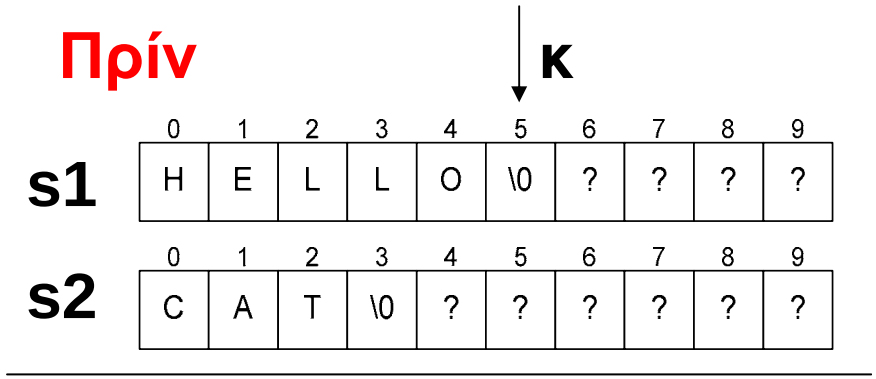


2) Υλοποίηση strcat() - έκδοση 2

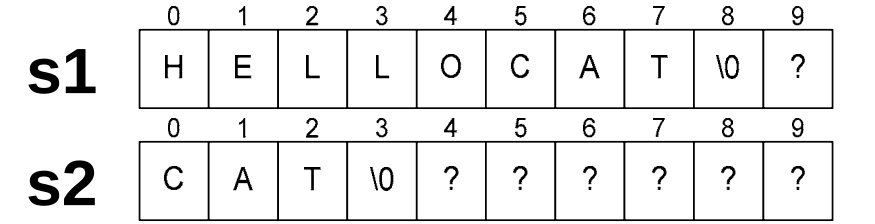
```
void mystrcat(char s1[], char s2[])
{
    int i=0, k=0;
    // Εύρεση \0 στο S1
    k = strlen(s1);

    // Αντιγραφή Στοιχείων
    while (s2[i] != '\0') {
          s1[k]=s2[i];
          i++;
          k++;
    }
    s1[k]='\0';  // προσθήκη NULL στο τέλος του s2
}
```

Πρίν



Μετά

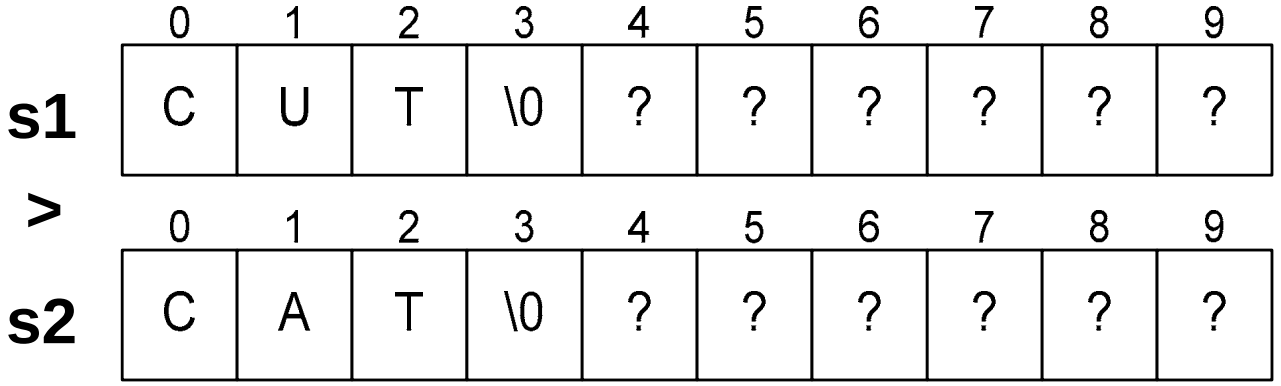


3) Η συνάρτηση strcmp()

Η συνάρτηση strcmp(s1, s2) συγκρίνει το s1 με s2 και επιστρέφει:

θετική τιμή, εάν s1>s2 μεγαλύτερο.
μηδέν, εάν s1==s2, και

αρνητική τιμή εάν s1< s2 (αλφαβητικά)



Επιστρέφει 20 διότι το CUT είναι 20 χαρακτήρες μεγαλύτερο από το CAT

3) Η συνάρτηση strcmp()

Οι συγκρίσεις γίνονται βάση του πίνακα ASCII

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Spa	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DE

3) Η συνάρτηση strcmp()

Κλήση της συνάρτησης strcmp()

```
#include <stdio.h>
int main()
{
    char s1[10]="HELLO";
    char s2[10]="CAT";
    int cmp = strcmp(s1,s2);
    if (cmp == 0)
            printf("Οι δυο λέξεις είναι οι ίδιες");
    else if (cmp > 0)
            printf("%s > %s", s1, s2);
    else
            printf("%s < %s", s1, s2);
}
```

cmp=5, επομένως εκτυπώνετε HELLO>CAT

2) Υλοποίηση strcmp()

- Υλοποιήστε την συνάρτηση

  int mystrcmp(char s1[], char s2[])

  η οποία συγκρίνει το s1 με s2 και επιστρέφει θετική τιμή εάν s1 μεγαλύτερο (αλφαβητικά) από το s2, μηδέν αν είναι ίσα, και αρνητική τιμή εάν s1 μικρότερο από s2.

Αλγόριθμος

- Έλεγξε επαναληπτικά εάν κάθε στοιχείο s1[i] είναι ίσο με s2[i].
- Εάν δεν είναι κάποιο στοιχείο ίσο επέστρεψε το s1[i]-s2[i]

2) Υλοποίηση mystrcmp()

```
int mystrcmp(char s1[], char s2[])
{
    int i=0;

    // Ελέγχουμε κάθε στοιχείο
    //  s1[i] αν είναι ίσο με s2[i]
    while (s1[i] != '\0') {
        if (s1[i] != s2[i])
            break;
        i++;
    }
    return s1[i]-s2[i];
}
```

Περίπτωση A

	0	1	2	3	4	5	6	7	8	9
s1	C	A	T	\0	?	?	?	?	?	?
>										
s2	C	\0	?	?	?	?	?	?	?	?

Περίπτωση B

	0	1	2	3	4	5	6	7	8	9
s1	C	A	T	\0	?	?	?	?	?	?
=										
s2	C	A	T	\0	?	?	?	?	?	?

Περίπτωση C

	0	1	2	3	4	5	6	7	8	9
s1	C	\0	?	?	?	?	?	?	?	?
<										
s2	C	A	T	\0	?	?	?	?	?	?

2) Υλοποίηση mystrcmp()

Γιατί είναι λάθος η πιο κάτω υλοποίηση της mystrcmp?

```
int mystrcmp(char s1[], char s2[])
{
    int i=0;

    // Ελέγχουμε κάθε στοιχείο s1[i] αν είναι ίσο s2[i]
    while (s1[i] == s2[i])  {
            i++;
    }
    return s1[i]-s2[i];
}
```