ΔΙΑΔΙΚΑΣΤΙΚΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ

5η Εβδομάδα: Χρήση Αλφαριθμητικών και Συναρτήσεις Διαχείρισης Αλφαριθμητικών

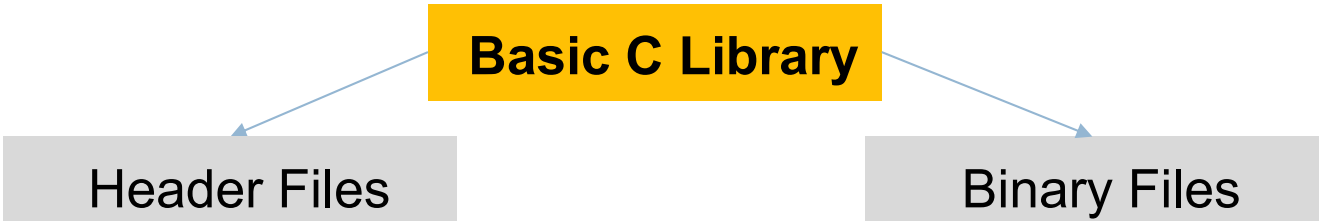
Αναφορές

Οι διαφάνειες της διάλεξης στηρίζονται, εν μέρει, σε υλικό παραδόσεων παλαιότερων ετών του **Τμήματος Ηλεκτρολόγων Μηχανικών και Τεχνολογία Υπολογιστών του Πανεπιστημίου Πατρών** καθώς και του **Τμήματος Πληροφορικής του Πανεπιστήμιο Κύπρου.**

Βιβλιοθήκες στη C

Βιβλιοθήκη είναι μια συλλογή μεταγλωττισμένων μονάδων που μπορούν να συνδεθούν (linked) στα προγράμματά μας μέσω των διεπαφών (header files - interfaces) που παρέχουν.

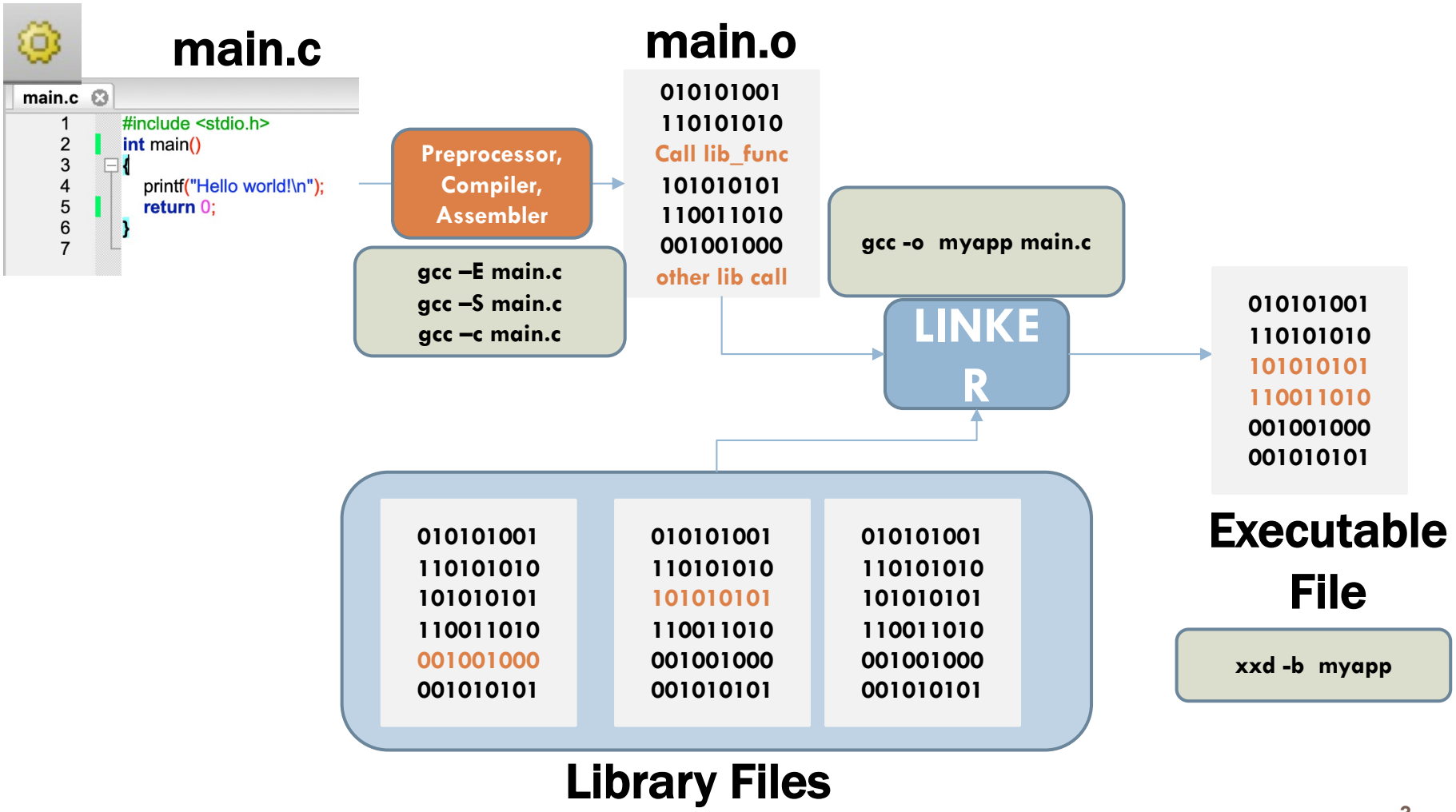
Με άλλα λόγια κάθε βιβλιοθήκη **αποτελείτε από** δυαδικά αρχεία σε object code (*.o) που περιέχουν υλοποιήσεις όλων των συναρτήσεων που έχουν δηλωθεί στα αρχεία επικεφαλίδων .h



- `<stdio.h>` **είναι η διεπαφή (interface) που περιέχει συναρτήσεις I/O.**

Πηγαίος κώδικας της βασικής βιβλιοθήκης: http://ftp.gnu.org/gnu/glibc/

Βιβλιοθήκες στη C



Συχνές λειτουργίες με string

Υπάρχουν κάποιες λειτουργίες που είναι πολύ συχνές πάνω σε Strings.

Παραδείγματα

- **Compare:** Σύγκριση δυο strings
- **Copy:** Αντιγραφή από ένα string σε άλλο
  - ολόκληρο ή μέρος του.
- **Concat:** σύνδεση δυο strings
- **Substring:** εύρεση συμβολοσειράς σε μια μεγαλύτερη ακολουθία

Η Βιβλιοθήκη <String.h> περιέχει συναρτήσεις για όλα τα πιο πάνω.

Για εξάσκηση θα υλοποιήσουμε κάποιες από τις συναρτήσεις μόνοι μας

Η Βιβλιοθήκη string.h

- Το αρχείο επικεφαλίδα (header file), string.h παρέχει συναρτήσεις για χειρισμό strings
- Περιέχει Διάφορες Συναρτήσεις, π.χ.,
  - **strlen(s)**, υπολογίζει το μέγεθος του string
  - **strcpy(s1,s2)**, αντιγράφει το s2 στο s1
  - **strcat(s1,s2)**, προσθέτει το s2 στο s1.
  - **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 σύγκριση γίνεται βάση του πίνακα ASCII)

Συναρτήσεις βασικής βιβλιοθήκης για αλφαριθμητικά

- πρότυπα στο `string.h`
- `char *strcpy (char *, const char *) ;`
- `int strcmp (const char *, const char *) ;`
- `char *strcat (char *, const char *) ;`
- `char * strtok(char *, const char *);`
- `char *strchr (const char *, char) ;`
- `size_t strcspn(const char *, const char *);`
- `size_t strlen (const char *) ;`

`size_t` -> **64-bit σε x64**

- και άλλες συναρτήσεις…

Δήλωση και ανάθεση τιμής σε αλφαριθμητικό

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

```
main ( ) {
 char name[10];

 name = "katerina";

 printf ("%s", name);

 scanf("%s", name);

 printf ("%s", name);

}
```

Λάθος (στη C)

```
#include <string.h>
main ( ) {
    char name[10];

    strcpy(name,"katerina");
    printf ("%s", name);

    scanf("%s", name);
    printf ("%s", name);

}
```

Σωστός τρόπος:

Άλλο εννοεί εδώ…

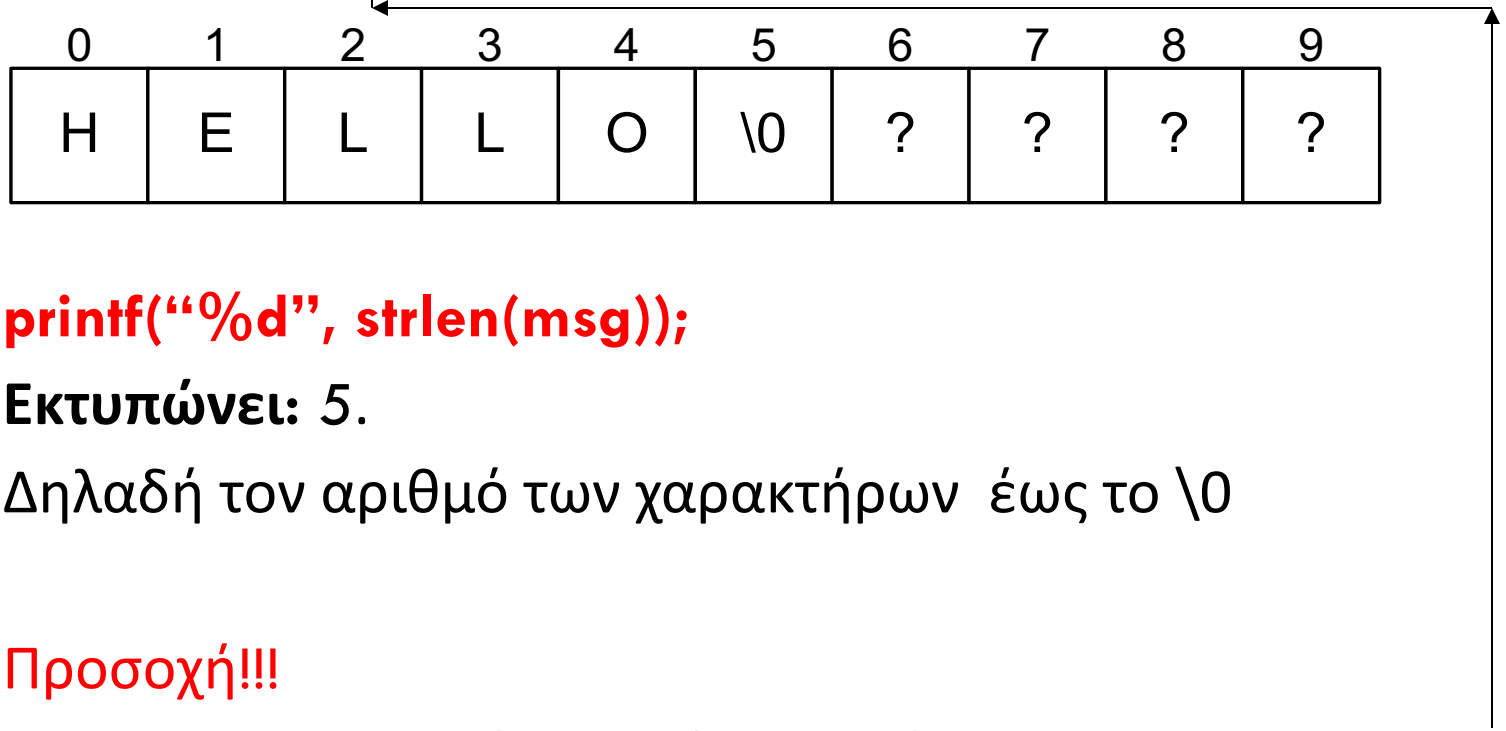
```
#include <stdio.h>

int main( )
{

  char name[100] ="Katerina";
  char user[100] ="";

  printf("Enter your name:");
  while (user != "Katerina") {
        scanf("%s",user);
        }

  printf("Hello %s\n", user);


  return 0;
}
```

- **Δεν κάνει** αυτό που χρειάζεται!!!
- Τι κάνει;
  - Συγκρίνει τη θέση που αρχίζει ο user με τη θέση στην οποία είναι αποθηκευμένο το "Katerina".

Η συνάρτηση strlen()

- Η συνάρτηση strlen μετρά το μέγεθος ενός String.

π.χ. char msg[10]="HELLO"



printf("%d", strlen(msg));

Εκτυπώνει: 5.

Δηλαδή τον αριθμό των χαρακτήρων έως το \0

Προσοχή!!!

Το strlen **δεν** μας λέει το μέγιστο μέγεθος του string.

Αυτό είναι 10 χαρακτήρες και το ξέρουμε πριν το compile ή με «printf("%ld", **sizeof msg**);»

Η συνάρτηση strlen()

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

int main()
{
    char x[10] = "123456" ;
    printf("%d", strlen(x));
    return 0;
}
```

Τυπώνει 6

Η συνάρτηση strlen()

- Ξέρουμε ότι η strlen είναι έτοιμη συνάρτηση.
- Για εξάσκηση, θα την υλοποιήσουμε μόνοι μας

ΑΣΚΗΣΗ

- Γράψετε την συνάρτηση: **int mystrlen(char c[])**

που μέτρα το μέγεθος ενός string.

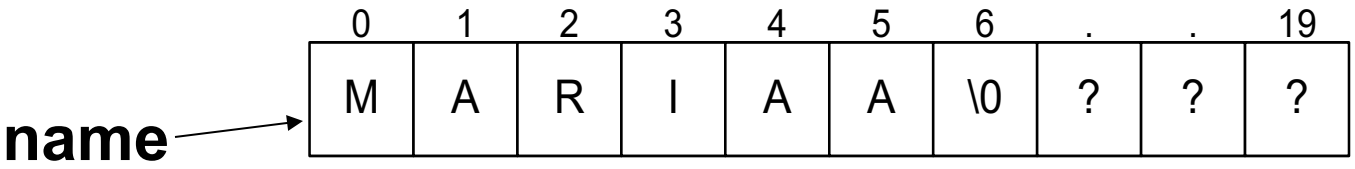
Η συνάρτηση επιστρέφει το μήκος του string

- **Τι θα επιστρέψει?**
  - mystrlen("abc") => **3**
  - char x[10] = "123456" ; mystrlen(x); => **6**
  - mystrlen("abc abc") => **7**

Η συνάρτηση strlen()

Αλγόριθμος

- Διάβασε το string από την αρχή μέχρι να βρεις το \0.
- Σε κάθε βήμα αύξησε κάποιο μετρητή κατά 1.



Ερώτηση

- Γιατί δεν μπορούμε να πάμε κατευθείαν στο τέλος του String (για να βρούμε κατευθείαν το μέγεθος);

Η συνάρτηση strlen()

```
#include <stdio.h>

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

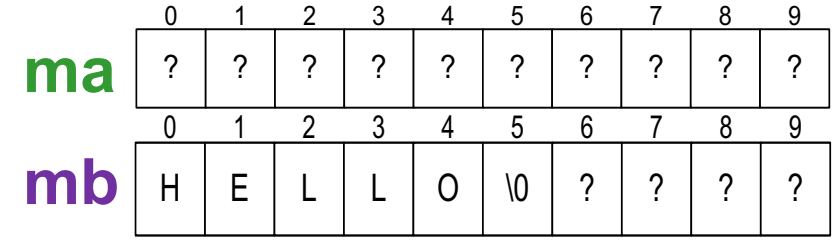
```
int main()
{
        char x[10] = "123456" ;
        printf("%d",
        mystrlen(x));
        return 0;
}
```

Η συνάρτηση strcpy()

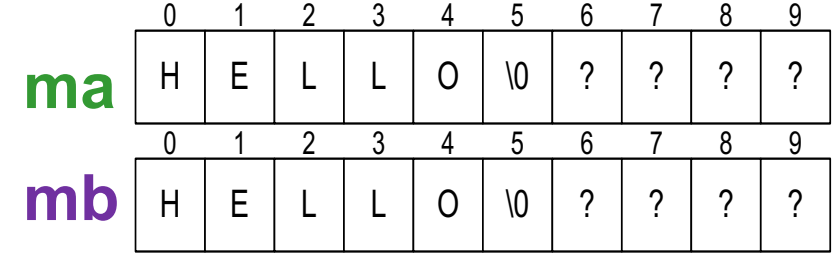
Η συνάρτηση strcpy(ma, mb) αντιγράφει το **mb** στο **ma**

π.χ.

Πρίν



Μετά




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

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.

Υλοποίηση της strcpy()

```
#include <stdio.h>
void mystrcpy(char to[ ], char from[ ]) {
   int i=0;
   while (from[i] != '\0') {
                        to[i] = from[i];
                        i++;
         }
         to[i]='\0';
return;
}
```

Πρίν

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

memcpy, memmove, memset

void pointer ως ειδοποίηση ότι μπορείτε να του περάσετε οποιοδήποτε είδος δείκτη

```
void *memcpy(void *dest, const void * src, size_t n)
```

Η strcpy () προορίζεται μόνο για συμβολοσειρές, ενώ η memcpy() είναι μια γενική συνάρτηση για την αντιγραφή bytes από την πηγή στην τοποθεσία προορισμού

```
void *memmove(void *dest, const void *src, size_t n)
```

Η συνάρτηση memmove χρησιμοποιείται για την αντιγραφή ενός καθορισμένου αριθμού byte από τη μια μνήμη στην άλλη ή για την επικάλυψη στην ίδια μνήμη

```
void *memset(void *str, int c, size_t n)
```

Υλοποίηση της memcpy()

```
void *my_memcpy(void *dest, const void * src, size_t n){
    char *mysrc = (char *)src;
    char *mydest = (char *)dest;
    int i=0;
    for (i=0; i<n; i++)
            mydest[i] = mysrc[i];

    return mydest;
}
```

```
int main(){
    int x=1;
    int y=2;
    my_memcpy(&x,&y,sizeof(int));
  return 0;
}
```

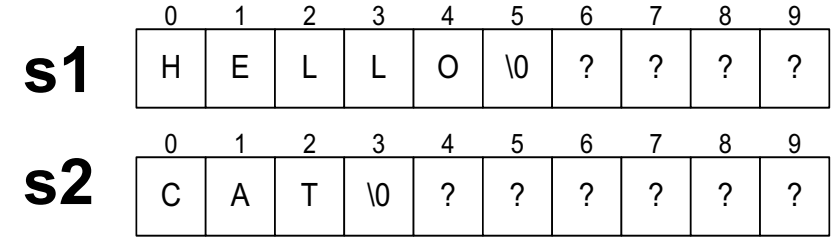
```
int main(){
    char str1[]="My String";
    char str2[30]={};
    my_memcpy(str2,str1,3);
  return 0;
}
```

Η συνάρτηση strcat()

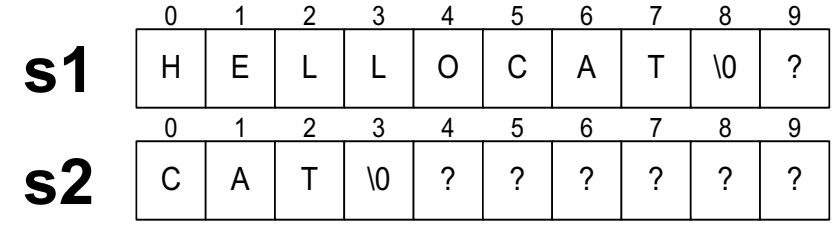
Η συνάρτηση strcat(s1, s2) αντιγράφει το s2 στο τέλος του s1 π.χ.

```
int main()
{
      char s1[10]="HELLO";
      char s2[10]="CAT";
      strcat(s1,s2);
      printf("s1=%s and s2=%s", s1, s2);
      return;
}
```

Πρίν



Μετά



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 K.
- Αντίγραψε κάθε s2[i], στην αντίστοιχη θέση s1[K+i].
- Πρόσθεσε \0 στο τέλος του s1.

Υλοποίηση strcat() – έκδοση 1

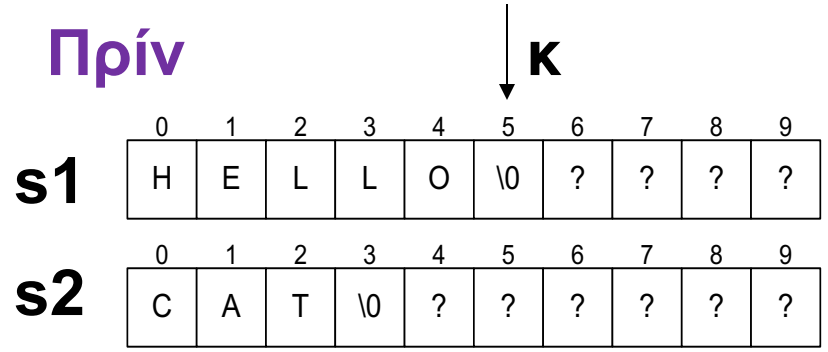
```
void mystrcat(char s1[], char s2[]){
        int i=0, k=0;
        // Εύρεση \0 στο S1
        while (s1[k] != '\0') {
                k++;
        }
        // Αντιγραφή Στοιχείων
        while (s2[i] != '\0') {
                s1[k+i]=s2[i];
                i++;
        }
        s1[k+i]='\0';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

Υλοποίηση strcat() - έκδοση 2

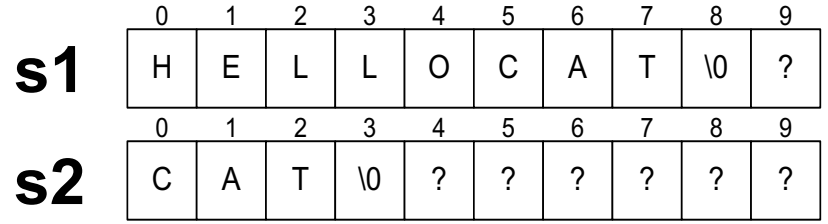
```
void mystrcat(char s1[], char s2[])
{
    int i=0, k=0;
    // Εύρεση \0 στο S1
    k = strlen(s1);
    // Αντιγραφή Στοιχείων
    while (s2[i] != '\0') {
        s1[k+i]=s2[i];
        i++;
    }
    s1[k+i]='\0';// προσθήκη NULL στο τέλος του s1
}
```

Χρήση strlen

Πρίν



Μετά



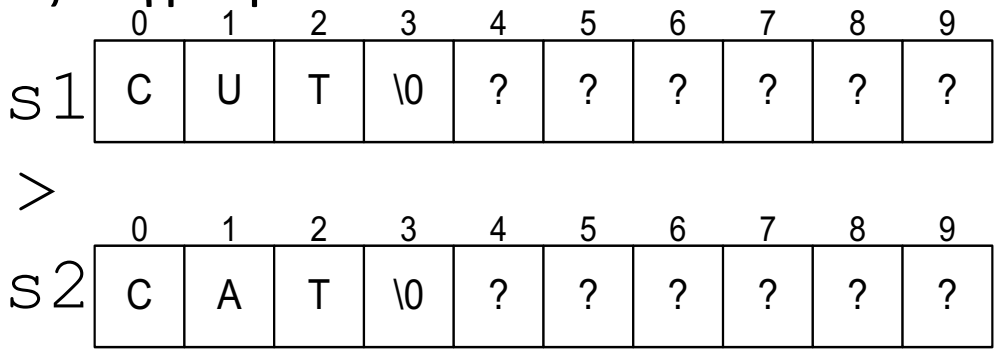
Η συνάρτηση strcmp()

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

Παράδειγμα 2 - `strcmp`

- Ζητούμενο: γράψετε συνάρτηση που συγκρίνει τα αλφαριθμητικά `s1` και `s2` και επιστρέφει «1» αν το `s1` μεγαλύτερο λεξικογραφικά, “0” αν είναι ίσα και «-1» αν `s2` μεγαλύτερο λεξικογραφικά



Επιστρέφει 1

Παράδειγμα 2 - strcmp

Α) Εκδοχή με πίνακες (from P.J. Plauger, Standard C Library)

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

    for(i=0; s1[i] == s2[i]; i++)
        if(s1[i] == 0) // i.e., NULL
            return 0;

    if (s1[i] < s2[i]) return -1;
     else return 1;
}
```

Παράδειγμα 2 - strcmp

Β) Εκδοχή με δείκτες

```
int mystrcmp(char *s1, char *s2) {

    for(; *s1 == *s2; ++s1, ++s2)
        if(*s1 == 0)
            return 0;

   if (*s1 < *s2) return -1;
    else return 1;
}
```

Παράδειγμα σύγκρισης αλφαριθμητικών

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

int main( )
{

  char name[100] ="Katerina";
  char user[100] ="";

  printf("Enter your name:");

  while (strcmp(user,name)!=0)
        scanf("%s",user);

  printf("Hello %s\n", user);


  return 0;
}
```

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

int main( )
{

  char name[100] ="Katerina";
  char user[100] ="";

  printf("Enter your name:");

  while (strcmp(user,name))
        scanf("%s",user);

  printf("Hello %s\n", user);


  return 0;
}
```

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

int main( ) {

  char name[100] = "Katerina"; /* initialization */
  char user[100] = "";

  printf("Enter your name:");

  while (strcmp(user,name) != 0)  {
       scanf("%s",user);
  }

  printf("Hello %s\n", user);
  printf("== :%d %p %p %s %s", user == name, user, name, user, name);


  return 0;
}
```

Ευχαριστώ για την προσοχή σας

Επικοινωνία

- Skype: fidas.christos
- Email: fidas@upatras.gr
- Phone: 2610 – 996491
- Web: http://cfidas.info

Άμεση Επικοινωνία μέσω Skype

SkypeID:
fidas.christos

- Ώρες γραφείου: Τετάρτη & Παρασκεύη 11:00-13:00

Join Zoom Meeting
https://upatras-gr.zoom.us/j/95080297961?pwd=MzRta0JRd3ZwVEVrREZNc09qbG1Zdz09

Το υλικό της διάλεξης είναι διαθέσιμο στο eclass

- https://eclass.upatras.gr/

ΔΙΑΔΙΚΑΣΤΙΚΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ

5η Εβδομάδα: Χρήση Αλφαριθμητικών και Συναρτήσεις Διαχείρισης Αλφαριθμητικών

Αναφορές

Οι διαφάνειες της διάλεξης στηρίζονται, εν μέρει, σε υλικό παραδόσεων παλαιότερων ετών του **Τμήματος Ηλεκτρολόγων Μηχανικών και Τεχνολογία Υπολογιστών του Πανεπιστημίου Πατρών** καθώς και του **Τμήματος Πληροφορικής του Πανεπιστήμιο Κύπρου.**

Πίνακες από Strings

- Είπαμε ότι το String είναι ένας μονοδιάστατος πίνακας από χαρακτήρες που τελειώνει σε \0.
- **Μπορούμε να έχουμε πίνακες από Strings;**

  ΝΑΙ π.χ. Λίστα με ονόματα, ημέρες, κτλ
- **Παραδείγματα**
  - char names[NUM_STUDENTS][NAME_LEN];
  - char weekdays[7][10]={“Monday”, “Tuesday”, “Wednesday”, “Thursday”, “Friday”, “Saturday”, “Sunday”};

	0	1	2	3	4	5	6			
0	M	o	n	d	a	y	\0	?	?	?
1	T	u	e	s	d	a	y	\0	?	?
6	S	u	n	d	a	y	\0	?	?	?

ΣΥΜΒΟΛΟΣΕΙΡΕΣ ► ΠΙΝΑΚΕΣ

- Πίνακες από συμβολοσειρές:
  - **char names[NUM_STUDENTS][NAME_LEN];**
  - **char weekdays[7][10]={“Monday”, “Tuesday”, “Wednesday”, “Thursday”, “Friday”, “Saturday”, “Sunday”};**

0	1	2	3	4	5	6	7	8	9
M	O	N	D	A	Y	\0			
T	U	E	S	D	A	Y	\0		
W	E	D	N	E	S	D	A	Y	\0
…	…	…	…	…	…	…	…	…	…
S	U	N	D	A	Y	\0			

Τι γίνεται όμως αν περιμένουμε δεδομένα από το πληκτρολόγιο;

Παραδείγματα


```c
#include <stdio.h>
#include <string.h>
#define W 5
#define N 9

int main()
{
    char words[W][N + 1]={{}};
    char input[N + 1]={};
    int i=0;
    while (printf("please enter word (%d) :", i+1),scanf("\n%[^\n]", input),  strcmp(input,"end")  ) {
            strcpy(words[i++], input);
            if(i==W)  break;
    }
    return 0;
}
```

```
please enter word (1) :qwertyuio
please enter word (2) :qwertyuiop
please enter word (3) :qwertyuiopa
please enter word (4) :qwertyuiopas
please enter word (5) :qwertyuiopasd
please enter word (6) :
```

Μπορούμε να ξεπεράσουμε τα όρια του πίνακα words

Παραδείγματα


```c
#include <stdio.h>
#include <string.h>
#define W 2
#define N 9
void print_w(char x[W] [N+1]);
int main()
{
    char words[W][N + 1]={{}};
    char input[N + 1]={};
    printf("%p\n",input);
    printf("%p\n",words[0]);
    printf("%p\n",words[1]);

    int i=0;
    while (printf("please enter word (%d) :", i+1),scanf("\n%[^\n]", input),  strcmp(input,"end")  ) {
            strcpy(words[i++], input);
             if(i==W)  break;

    }

    return 0;
}
```

cppcheck
invalidscanf : warning : scanf() without field width limits can crash with huge input data. Add a field width specifier to fix this problem.

```
input address: 0x7ffeeaf70a56
words[0] address: 0x7ffeeaf70a60
words[1] address: 0x7ffeeaf70a6a
```

```
please enter word [0] :12
please enter word [1] :qwertyuiopp
```

```
Word (0) :p
Word (1) :qwertyuiopp
```

Ο δέκατος χαρακτήρας του input έχει γίνει p, ο επόμενος χαρακτήρας p μπήκε στη διεύθυνση μνήμης που δείχνει το πρώτο κελί του πίνακα words[0][0]

Παραδείγματα

```
input address: 0x7ffeeaf70a56
words[0] address: 0x7ffeeaf70a60
words[1] address: 0x7ffeeaf70a6a
```



```
please enter word [0] :12
```

	0	1	2	3	4	5	6	7	8	9
input	1	2	\0							
Word[0]	1	2	\0							
Word[1]	\0									
	…	…	…	…	…	…	…	…	…	…
	…	…	…	…	…	…	…	…	…	…

```
while (printf("please enter word [%d] :", i),scanf("\n%[^\n]", input),  strcmp(input,"end")  ) {
        strcpy(words[i++], input);
        if(i==W)  break;
}
```

```
please enter word [1] :qwertyuiopp
```

	0	1	2	3	4	5	6	7	8	9
input	q	w	e	r	t	y	u	i	o	p
Word[0]	p	\0								
Word[1]	q	w	e	r	t	y	u	i	o	p
	p	\0	…	…	…	…	…	…	…	…
	…	…	…	…	…	…	…	…	…	…

Παραδείγματα


```
#include <stdio.h>
#include <string.h>
#define W 2
#define N 9
void print_w(char x[W] [N+1]);
int main()
{
    char words[W][N + 1]={{}};
    char input[N + 1]={};
    int i=0;
    while (printf("please enter word (%d) :", i+1),scanf("%9[^\n]", input),  strcmp(input,"end")  ) {
            strcpy(words[i++], input);
             if(i==W)  break;

    }
    print_w(words);
    return 0;
}

void print_w(char x[W] [N+1]){
        int i;
        for (i=0; i<W; i++) {
            printf("Word (%d) :%s\n", i, x[i]);
        }
        return;
}
```

```
please enter word [0] :123456789q
please enter word [1] :Word [0)] :123456789
Word [1)] :q
```

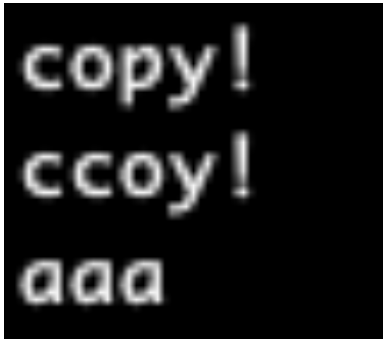
scanf("%9[^\n]", input)

```c
#include <stdio.h>
#include <string.h>
#define W 2
#define N 9
void print_w(char x[W] [N+1]);
int eat_up_remaining_input();
int main()
{
    char words[W][N + 1]={{}};
    char input[N + 1]={};
    int i=0;
    while (printf("please enter word [%d] :", i),scanf("\n%9[^\n]", input),  strcmp(input,"end")  ) {
            if( eat_up_remaining_input()>0) continue;
            strcpy(words[i++], input);
             if(i==W)  break;
    }
    print_w(words);
    return 0;
}

void print_w(char x[W] [N+1]){
       int i;
       for (i=0; i<W; i++) {
           printf("Word [%d)] :%s\n", i, x[i]);
       }
       return;
}
int eat_up_remaining_input(){
     int x=0;
      while(getchar()!='\n') x++;
      return x;
}
```

```
please enter word [0] :123456789q
please enter word [0] :123456789
please enter word [1] :qwertyuiop
please enter word [1] :qwertyuio
Word [0)] :123456789
Word [1)] :qwertyuio
```

char * strncpy(char *, const char *, size_t);

```c
#include <stdio.h>
#include <string.h>
#define N 64

int main(void) {

        char str[N] ;
        char word[] = "copy!";
        char word2[] = "aaa";

        strcpy(str, word);
        printf("%s\n", str);

        strncpy(str+1, word, 2);

        printf("%s\n", str);

        strncpy(str, word2, 4);
        printf("%s\n", str);

        return 0;
}
```



char * strstr(const char *, const char *);

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

int main(void) {
      char str[ ] = "testabcabc"
      char * ch_ptr ;

      printf("%s\n", str);

      ch_ptr = strstr(str, "abc");
      strncpy(ch_ptr, "FGH", 3);

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

```
testabcabc
testFGHabc
```

Ευχαριστώ για την προσοχή σας

Επικοινωνία

- Skype: fidas.christos
- Email: fidas@upatras.gr
- Phone: 2610 – 996491
- Web: http://cfidas.info

Άμεση Επικοινωνία μέσω Skype

SkypeID:
fidas.christos

- Ώρες γραφείου: Τετάρτη & Παρασκεύη 11:00-13:00

Join Zoom Meeting
https://upatras-gr.zoom.us/j/95080297961?pwd=MzRta0JRd3ZwVEVrREZNc09qbG1Zdz09

Το υλικό της διάλεξης είναι διαθέσιμο στο eclass

- https://eclass.upatras.gr/