Διάλεξη 3: Δείκτες και Πίνακες

Στην ενότητα αυτή θα μελετηθούν τα εξής επιμέρους θέματα:

Αριθμητική Δεικτών

Δείκτες και Πίνακες

Παραδείγματα

Διδάσκων: Δημήτρης Ζεϊναλιπούρ

Δείκτες (pointers) και διευθύνσεις

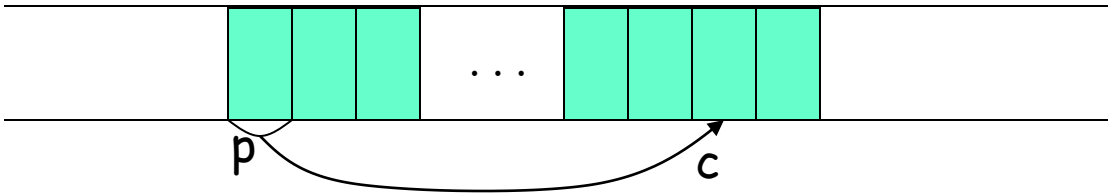
- Δείκτης είναι μια μεταβλητή που περιέχει τη διεύθυνση μιας άλλης μεταβλητής.
- Δείκτες χρησιμοποιούνται στη C γιατί
  - Μερικές φορές είναι ο μόνος τρόπος για τη διατύπωση κάποιου υπολογισμού.
  - Συνήθως οδηγούν σε πιο περιεκτικό και αποτελεσματικό κώδικα.
- Η μνήμη είναι οργανωμένη ως μια σειρά κελιών, με διαδοχική αρίθμηση, ή διευθύνσεις, που μπορούν να χρησιμοποιηθούν ένα-ένα ή σε συνεχείς ομάδες.
  - 1 κελί τους ενός byte => π.χ., char,
  - 2 κελιά τους ενός byte => π.χ., short,
  - 4 κελιά του ενός byte => π.χ., int.
- Δείκτης είναι μια ομάδα κελιών 4 (x86) ή 8 (x64) που μπορούν να κρατήσουν μια διεύθυνση.

Βασικοί Τύποι Δεδομένων C
Περίληψη (για x86)

Τύπος	Μέγεθος	Πεδίο Τιμών	Μοναδικές
char,	1byte	'a'..'z' 'A'..'Z''0'..'9'	2^8 ή 256
int,	4 bytes	$-2^{31}..2^{31}-1$	2^{32}
float,	4 bytes	$10^{-37}..10^{38}$	2^{32}
double	8 bytes	$10^{-307}..10^{308}$	2^{64}
δείκτης	4 bytes	διευθύνσεις $(0..2^{32}-1)$	2^{32}

Δείκτες και διευθύνσεις (συν.)

- Έστω c ένας int και p ένας δείκτη προς τον c. Στο επίπεδο της μνήμης, γραφικά, αυτό σημαίνει:



- Τελεστής διεύθυνσης: & (η διεύθυνση του αντικειμένου)
  - Δίνει τη διεύθυνση ενός αντικειμένου. π.χ. &c αποδίδει τη διεύθυνση του c.
  - p = &c : έχει ως αποτέλεσμα ανάθεση στο δείκτη p τη διεύθυνση του c.
- Τελεστής Έμμεσης Αναφοράς: * (το περιεχόμενο του δείκτη)
  - Όταν εφαρμόζεται σε δείκτη προσπελάζει το αντικείμενο που δείχνει ο δείκτης.
  - Το *p, έχει τιμή την τιμή του c.

Δείκτες και διευθύνσεις (συν.)

- Δήλωση δείκτη:

```
τύπος   *όνομα_μεταβλητής
```

π.χ.

```
int   *p;
int    foo(char *);
```

- Επομένως ένας δείκτης δείχνει σε αντικείμενα κάποιου συγκεκριμένου τύπου δεδομένων. (εξαίρεση: δείκτης σε void)

- Προσοχή στη χρήση του * !!
  - a*b; (Τελεστής γινομένου)
  - int *p; (Δήλωση Δείκτη)
  - x = *p + 1; (Τελεστής Έμμεσης Αναφοράς)

Παράδειγμα με δείκτες

```
int    x = 1,   y = 2,   z[10];
int    *ip,  *iq;

ip   = &x;                  /* ο ip δείχνει τώρα την x */
*ip  = *ip + 1;             /* η μεταβλητή που δείχει ο ip
                               (η x) αυξάνεται κατά 1 */
y    = *ip + 3;             /* η y παίρνει την τιμή 5 */
*ip  = 0;                   /* η x παίρνει την τιμή 0 */
ip   = &z[6];               /* ο ip δείχνει τώρα το z[6] */
iq   = ip;                  /*ο iq δείχνει εκεί που
                                 δείχνει και ο ip */
```

Δείκτες και ορίσματα συναρτήσεων

- Ξέρουμε ότι αν καλέσουμε μια συνάρτηση με κάποιες παραμέτρους (δια τιμής) τότε αυτές οι τιμές δεν αλλάζουν (Εξαίρεση: πίνακες οι οποίοι περνάνε δια-αναφοράς).

- Παράδειγμα: Να γράψετε μια συνάρτηση η οποία να έχει σαν αποτέλεσμα την εναλλαγή δύο στοιχείων.

```
void swap1(int x, int y){
  int temp;

  temp = x;
  x    = y;
  y    = temp;
}
```

Λάθος!
Καλώντας την συνάρτηση ως swap1(a,b) δεν θα έχει καμιά επίδραση στις τιμές των a και b.

Παράδειγμα

```
void swap(int *px, int *py){
    int temp;

    temp  = *px;
    *px   = *py;
    *py   = temp;
}
```

Σωστό!

- Η κλήση swap(&a, &b) θα έχει το επιθυμητό αποτέλεσμα:
  - Αφού ο τελεστής & δίνει τη διεύθυνση μιας μεταβλητής, οι &a και &b είναι δείκτες προς τις μεταβλητές a και b.
  - Οι παραμέτροι της swap δηλώνονται ως δείκτες και η προσπέλαση και ανταλλαγή των τιμών των a και b γίνονται μέσω των δεικτών αυτών.

Δείκτες και Πίνακες

- Στην C υπάρχει ισχυρός δεσμός ανάμεσα στους **δείκτες** και στους **πίνακες**. Οποιαδήποτε πράξη μπορεί να γίνει με **δείκτες πινάκων** (πχ a[i]) μπορεί να γίνει και με **δείκτες διευθύνσεων *pa**.
- Έστω πίνακας int a[10]. Η δήλωση αυτή ορίζει ένα μπλοκ 10 διαδοχικών αντικειμένων με ονόματα a[0], a[1], ..., a[9].
- Αν `int *pa;`

  τότε η ανάθεση `pa = &a[0];`

  κάνει τον pa να δείχνει το μηδενικό στοιχείο του πίνακα.
- ***Αριθμητική δεικτών*** : Αν ο pa είναι δείκτης σε κάποια θέση του πίνακα, τότε οι pa ± i, pa++, pa-- είναι επίσης δείκτες
  - `pa + 1, pa++` δείκτης στο επόμενο στοιχείο του πίνακα από αυτό που δείχνει ο pa.
  - `pa + i , (pa – i)` δείκτης στο σημείο του πίνακα που βρίσκεται i θέσεις μετά (πριν) από αυτό που δείχνει ο pa.

Δείκτες και Πίνακες (συν.)

- Έτσι, αν pa = &a[0], τότε για παράδειγμα,

  pa + 2 δείχνει στη τρίτη θέση του πίνακα, δηλ. στο a[2].

  *(pa + 2) έχει τιμή την τιμή της 3ης θέσης του πίνακα, δηλ. το a[2].

- Έξ'ορισμού, η τιμή μιας μεταβλητής τύπου πίνακα (int a[]) είναι η διεύθυνση του μηδενικού στοιχείου του πίνακα.
- Επομένως αντί pa = &a[0] μπορούμε να γράψουμε pa = a.
- Τότε οι pa και a έχουν τις ίδιες τιμές και μπορούν να χρησιμοποιούνται η μια στη θέση της άλλης:
  - Το a[i] είναι ταυτόσημο με τα *(pa +i) ή *(a + i) ή pa[i].
  - Το &a[i] είναι ταυτόσημο με τα a+i, pa+i.

Δείκτες και Πίνακες (συν.)

- Προσοχή: ο δείκτης είναι μεταβλητή και έτσι pa = a και pa++ είναι έγκυρες εκφράσεις. Το **όνομα ενός πίνακα όμως δεν είναι μεταβλητή**, επομένως κατασκευές όπως a = pa και a++ δεν είναι έγκυρες!
- Ποιες από τις πιο κάτω εντολές είναι έγκυρες;

```
int t[10] = {}, i = 5, *p;
p     = t;
p[2] = 3;
++p;
*p    = 14;
*(t+i) = 33;
++t;            => compile error
```

value	0	14	3	0	0	33	0	0	0	0
index	t[0]	t[1]	t[2]	t[3]	t[4]	t[5]	t[6]	t[7]	t[8]	t[9]

Παράδειγμα - strcopy

- Ζητούμενο: να γράψετε συνάρτηση που αντιγράφει ένα αλφαριθμητικό t στο αλφαριθμητικό s.
- Γιατί δεν είναι αρκετό να γράψουμε s=t ;;;
- Εκδοχή με πίνακες:

```
void strcopy (char *s, char *t) {
  int i;

  i=0;
  while ((s[i] = t[i]) != '\0')
        i++
}
```

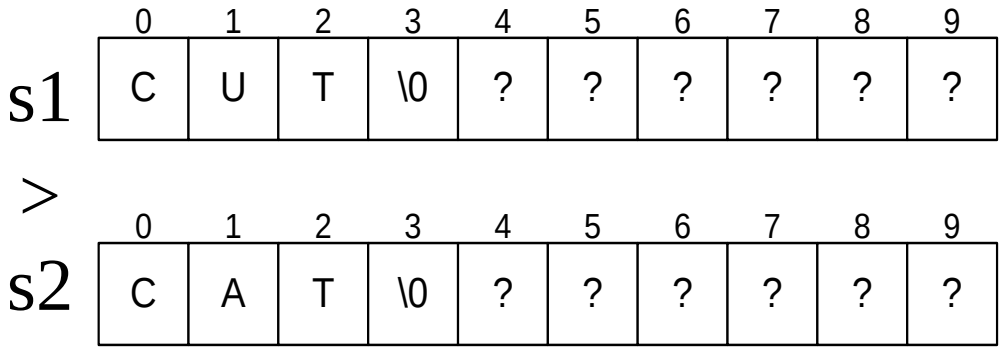
Παράδειγμα - strcopy

- Εκδοχή με δείκτες:

```
void strcopy1 (char *s, char *t)
{
   while ((*s = *t) != '\0'){
        s++;
        t++;
    }
}
```

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* (P.J. Plauger, Standard C Library):**

```
int mystrcmp(const char *s1, const char *s2) {
    int i;

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

    return (s1[i] < s2[i]) ? -1 : 1;
}
```

Δεν επιτρέπει την αλλαγή του s1 ή s2

- **Σημειώστε ότι επιστρέφεται «0» όταν η μια συμβολοσειρά είναι υπο-ακολουθία της άλλης**

Παράδειγμα 2 - strcmp

- **Β) Εκδοχή με δείκτες**

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

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

    return (*s1 < *s2) ? -1 : 1;
}
```

Δεν επιτρέπει την αλλαγή του s1 ή s2