

# Γλώσσες Προγραμματισμού

- Διδάσκων: Ανδρέας Παπασαλούρος
- Ιστοσελίδα:  
<http://www.samos.aegean.gr/math/andrapapas/courses/pl/default.htm>
- Πλατφόρμα ηλεκτρονικής μάθησης:  
<http://myria.math.aegean.gr/moodle/>

# Πλατφόρμα ηλεκτρονικής μάθησης

- Υλικό του μαθήματος
- Υποβολή εργασιών
- Επικοινωνία
- Συζητήσεις για το μάθημα

# Η πλατφόρμα ηλεκτρονικής μάθησης moodle

Department of Mathematics Course Management System - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://myria.math.aegean.gr/moodle/

Department of Mathematics Cou... Τμήμα Μαθηματικών, Πανεπιστήμιο Αι...

## Department of Mathematics Course Management System

Δεν έχετε εισέλθει. (Είσοδος)

Ελληνικά (el)

### Διαθέσιμα Μαθήματα

**LaTeX και Postscript**  
Teacher: [Αντώνης Τσολομύτης](#)  
Εισαγωγή στο σύστημα στοιχειοθεσίας LaTeX με έμφαση στο μαθηματικό κείμενο και στη γλώσσα περιγραφής σελίδας Postscript.

**Γλώσσες Προγραμματισμού**  
Teacher: [Ανδρέας Παπασαλούρος](#)  
Μάθημα του δευτέρου εξαμήνου. Διδάσκεται η γλώσσα προγραμματισμού C. Αποτελεί συνέχεια του μαθήματος "Εισαγωγή στην Πληροφορική" του πρώτου εξαμήνου.

**Εισαγωγή στην Πληροφορική**  
Administrator: [Αντώνης Τσολομύτης](#)  
Administrator: [Ανδρέας Παπασαλούρος](#)  
Εισαγωγή στην Πληροφορική

### Ημερολόγιο

Μάρτιος 2007

Κυρ	Δευ	Τρι	Τετ	Πεμ	Παρ	Σαβ
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

Δεν έχετε εισέλθει. (Είσοδος)



# Εγγραφή στην πλατφόρμα

Department of Mathematics Course Management System: Είσοδος στο δικτυακό τόπο - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://myria.math.aegean.gr/moodle/login/index.php

Department of Mathematics Cou... Τμήμα Μαθηματικών, Πανεπιστήμιο Αι...

## Department of Mathematics Course Management System

Δεν έχετε εισέλθει. (Είσοδος)

► Είσοδος στο δικτυακό τόπο Ελληνικά (el)

Επιστρέφετε σε αυτό το δικτυακό τόπο;	Είναι η πρώτη σας φορά εδώ;
<p>Εισέλθετε εδώ χρησιμοποιώντας όνομα χρήστη και κωδικό πρόσβασης: (Τα cookies πρέπει να είναι ενεργοποιημένα στο φυλλομετρητή σας) ?</p> <p>Όνομα χρήστη: <input type="text" value="andrapas"/> <input type="button" value="Είσοδος"/></p> <p>Κωδικός πρόσβασης: <input type="password" value="xxxxxxxx"/></p> <hr/> <p>Μερικά μαθήματα μπορεί να επιτρέπουν πρόσβαση επισκεπτών:</p> <p><input type="button" value="Είσοδος ως επισκέπτης"/></p> <hr/> <p>Ξεχάσατε το όνομα χρήστη ή τον κωδικό πρόσβασης;</p> <p><input type="button" value="Ναι, βοήθησέ με να εισέλθω"/></p>	<p>Γεια σας! Για πλήρη πρόσβαση στα μαθήματα θα χρειαστείτε ένα λεπτό για να δημιουργήσετε νέο λογαριασμό για τον εαυτό σας σε αυτό το δικτυακό τόπο. Κάθε ξεχωριστό μάθημα μπορεί επιπλέον να έχει ένα "κλειδί εγγραφής" μιας χρήσης, το οποίο δε χρειάζεστε ακόμη. Αυτά είναι τα βήματα:</p> <ol style="list-style-type: none"><li>1. Συμπληρώστε τη φόρμα <u>Νέου λογαριασμού</u> με τα δεδομένα σας.</li><li>2. Ένα μήνυμα ηλεκτρονικού ταχυδρομείου θα αποσταλεί στη διεύθυνσή σας.</li><li>3. Διαβάστε το μήνυμα και επιλέξτε τη διεύθυνση που περιέχει.</li><li>4. Ο λογαριασμός σας θα επιβεβαιωθεί και θα συνδεθείτε.</li><li>5. Τώρα, επιλέξτε το μάθημα στο οποίο θέλετε να συμμετέχετε.</li><li>6. Αν σας ζητηθεί ένα "κλειδί εγγραφής" - χρησιμοποιήστε αυτό που σας έδωσε ο εκπαιδευτής σας. Έτσι θα "εγγραφείτε" στο μάθημα.</li><li>7. Τώρα μπορείτε να έχετε πρόσβαση στο μάθημα. Από εδώ και στο εξής θα χρειάζεστε να δώσετε μόνο το όνομα χρήστη και τον κωδικό πρόσβασης (όπως είναι σε αυτή τη σελίδα) για να εισέλθετε και να προσπελάσετε τα μαθήματα στα οποία έχετε εγγραφεί.</li></ol> <p><input type="button" value="Ξεκινήστε τώρα δημιουργώντας νέο λογαριασμό!"/></p>

Δεν έχετε εισέλθει. (Είσοδος)

[Αρχή](#)

# Η κεντρική σελίδα του μαθήματος

The screenshot shows a Mozilla Firefox browser window with the address bar containing `http://myria.math.aegean.gr/moodle/course/view.php?id=5`. The page title is "Μάθημα: Γλώσσες Προγραμματισμού". The course name is "Γλώσσες Προγραμματισμού" (PL2007). The user is logged in as "Ανδρέας Παπασαλούρος (Έξοδος)".

**Navigation and Tools:**

- Switch role to... (dropdown)
- Επεξεργασία
- Μετάβαση σε... (dropdown)

**Left Sidebar:**

- Άτομα
  - Συμμετέχοντες
- Δραστηριότητες
- Search Forums
- Διαχείριση
- Μαθήματα

**Main Content:**

- Περιγραφή θέματος
  - Ομάδα συζητήσεων ειδήσεων
- 1 Γενικά στοιχεία για το μάθημα

**Right Sidebar:**

- Τελευταία νέα
  - Προσθήκη νέου θέματος... (Δεν έχουν σταλεί ακόμα ειδήσεις)
- Επικείμενα γεγονότα
  - Δεν υπάρχουν γεγονότα στο άμεσο μέλλον
  - Ημερολόγιο...
  - Νέο γεγονός...
- Πρόσφατη δραστηριότητα
  - Δραστηριότητα από Τετάρτη, 28 Μάρτιος 2007, 11:11
  - [Full report of recent activity...](#)
  - Τίποτε νέο από την τελευταία σύνδεσή σας

**Footer:**

- Moodle Docs for this page
- Έχετε εισέλθει ως Ανδρέας Παπασαλούρος (Έξοδος)

Αναλυτικός σχεδιασμός προγραμμάτων

# Αρθρωτή (δομημένη) ανάπτυξη προγραμμάτων

- Ενότητα 5.6 του βιβλίου του Roberts
- Ένα πρόγραμμα υλοποιεί τη λύση ενός (συνήθως σύνθετου) **προβλήματος**.
- Παράδειγμα: Ένα ταξίδι στο Λονδίνο
  - Έκδοση φθηνού αεροπορικού εισιτηρίου
    - Συμβουλή πράκτορα
    - Έκδοση του εισιτηρίου
  - Κλείσιμο ξενοδοχείου
  - Ταξίδι
  - Διαμονή
  - Επιστροφή

# Αναλυτικός σχεδιασμός προγραμμάτων

- Μια **μέθοδος** για την επίλυση σύνθετων προβλημάτων είναι η **ανάλυσή** (analysis) ή αποσύνθεση (decomposition) τους σε επιμέρους (“μικρότερα”) **υποπροβλήματα**.
- Κάθε υποπρόβλημα αναλύεται περαιτέρω μέχρι να φτάσουμε σε απλά ή ήδη λυμένα προβλήματα τα οποία επιλύονται άμεσα.
- Η σχεδίαση ενός προγράμματος με την ανάλυση ενός προβλήματος από το πιο γενικό στο πιο ειδικό ονομάζεται **αναλυτικός σχεδιασμός** ή **βηματική εκλέπτυνση**.



# Αναλυτικός σχεδιασμός

- Αναλύουμε ένα πρόβλημα σε υποπροβλήματα
  - Ένα υποπρόβλημα είναι δυνατόν να αναλύεται σε περαιτέρω υποπροβλήματα.
- Ορίζουμε μια κύρια συνάρτηση για την επίλυση του προβλήματος.
- Ορίζουμε μια συνάρτηση ή διαδικασία για κάθε υποπρόβλημα.
- Τα υποπροβλήματα στα οποία αναλύεται ένα πρόβλημα είναι συναρτήσεις που καλούνται από τη συνάρτηση που επιλύει το πρόβλημα

# Παράδειγμα

```
main ()
{
    findTickets ();
    findHotel ();
    travel ();
    comeBack ();
}

void findTickets () {
    consultAgent ();
    bookTickets ();
}

...
```

Πίνακες στην C

# Πίνακες στην C

- Δήλωση πίνακα
  - `#define SIZE 20`
  - `int values[SIZE];`
- Προσπέλαση στοιχείου πίνακα
- `int v;`
- `v = values[10];`

# Πίνακες στην C (συνέχ.)

- Προσπέλαση όλων των στοιχείων ενός πίνακα

```
int i;
```

```
for (i = 0; i < SIZE; i++) {
```

```
    printf("size[%d] = %d", i, values[i]);
```

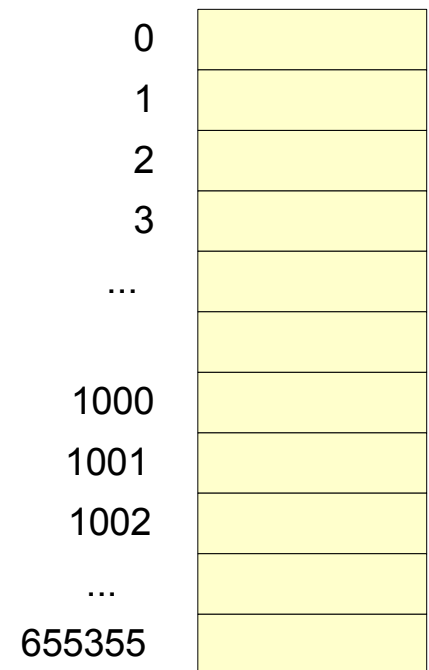
```
}
```

# Εσωτερική αναπαράσταση δεδομένων

- Κάθε τιμή δεδομένων αποθηκεύεται εσωτερικά με έναν αριθμό bits.
- Η μικρότερη μονάδα αποθήκευσης είναι το byte
  - 1 byte = 8 bits
  - 1 word = 2 bytes
- Η χωρητικότητα της μνήμης ενός υπολογιστή μετρείται σε bytes
  - 1 KB =  $2^{10}$  bytes = 1024 bytes
  - 1 MB =  $2^{20}$  bytes = 1.048.576 bytes

# Διευθύνσεις μνήμης

- Κάθε byte μνήμης προσδιορίζεται από μια **αριθμητική διεύθυνση**
- Κάθε byte μνήμης μπορεί να αποθηκεύσει ένα χαρακτήρα πληροφορίας
- Κατά τη δήλωση μιας μεταβλητής δεσμεύεται κατάλληλος χώρος στη μνήμη.

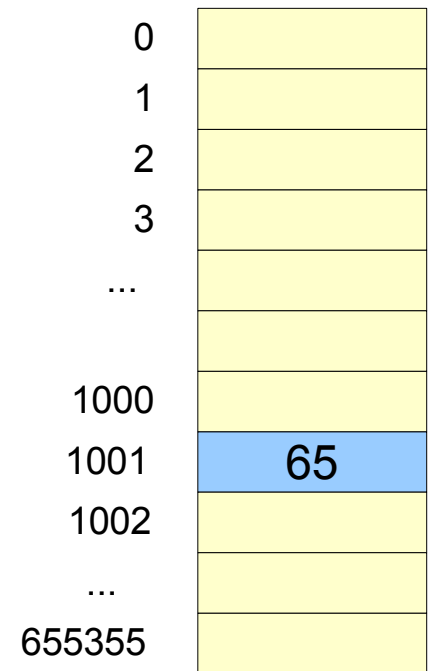


# Διευθύνσεις μνήμης (2)

- Η δήλωση

```
char ch = 'A' ;
```

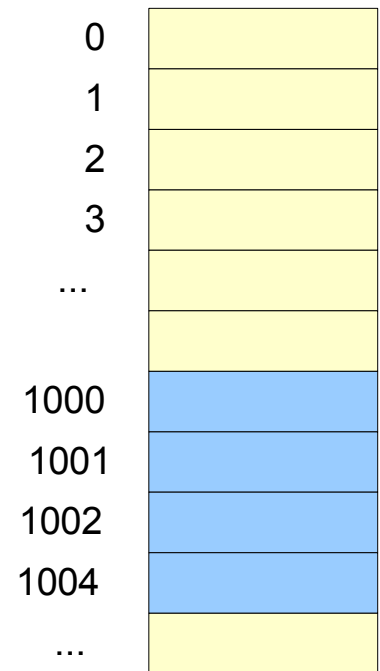
δεσμεύει **ένα** byte σε μια συγκεκριμένη διεύθυνση της μνήμης και αποθηκεύει σε αυτή τον κωδικό **ASCII** του χαρακτήρα A (65).





# Διευθύνσεις μνήμης (3)

- Άλλες τιμές δεδομένων διαφορετικών τύπων απαιτούν διαφορετικό αριθμό bytes για την αποθήκευσή τους.
- Για παράδειγμα, ένας ακαίρεος (τύπος `int`) δεσμεύει 2 ή ακόμη και byte της μνήμης, ανάλογα με την υλοποίηση.
- Ένας αριθμός τύπου `double` δεσμεύει 8 bytes μνήμης.



# Ο τελεστής `sizeof`

- Ο τελεστής `sizeof` της C επιστρέφει τον αριθμό των bytes που δεσμεύει μια μεταβλητή, σταθερά ή τύπος δεδομένων.
- Παράδειγμα

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

```
main()
```

```
{
```

```
    char c = 'b';
```

```
    printf("%d\n", sizeof(c));        /* 1 */
```

```
    printf("%d\n", sizeof(12));       /* 2 */
```

```
    printf("%d\n", sizeof(double));  /* 8 */
```

```
    printf("%d\n", sizeof(12.5));    /* 8 */
```

```
}
```

# Κατανομή μνήμης σε πίνακες

- Κατά την δήλωση ενός πίνακα δεσμεύεται χώρος στη μνήμη ίσος με το μέγεθος του πίνακα επί τον χώρο που δεσμεύει ο τύπος του πίνακα.

- Για παράδειγμα, η δήλωση

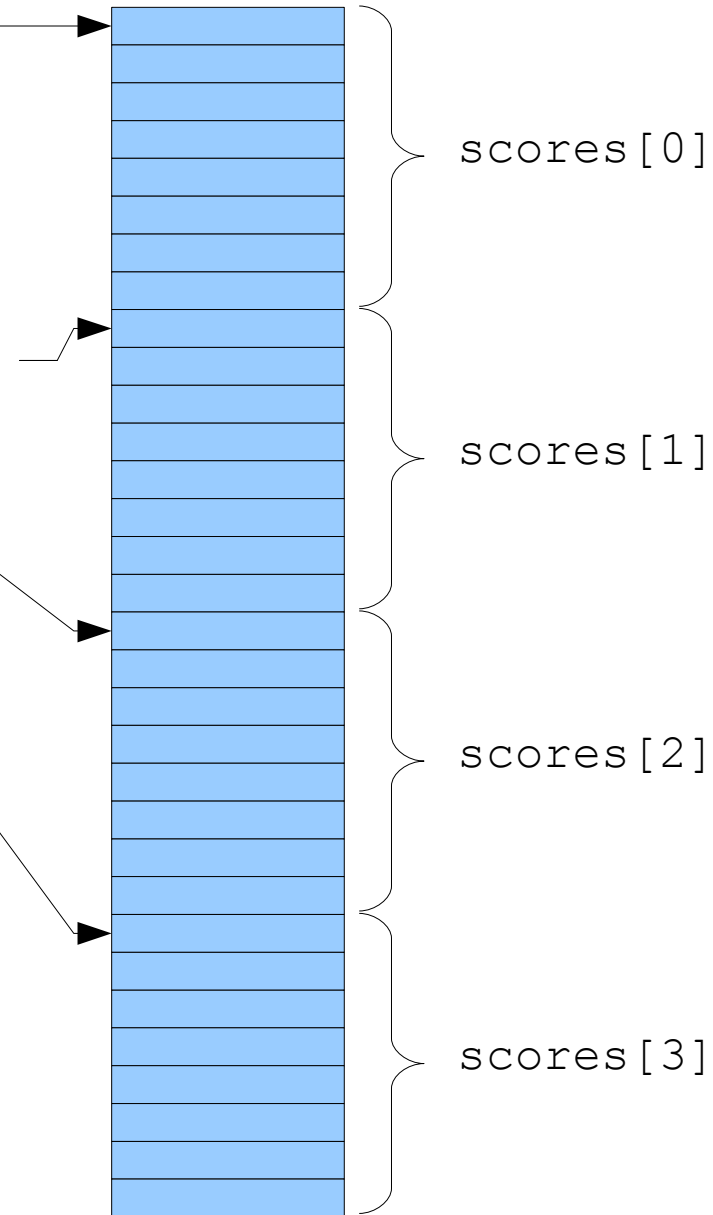
**double scores[4];**

δεσμεύει  $4 \times \text{sizeof}(\text{double})$   
 $= 4 \times 8 = 32$  bytes

- Το όνομα του πίνακα περιέχει τη διεύθυνση βάσης του πίνακα.

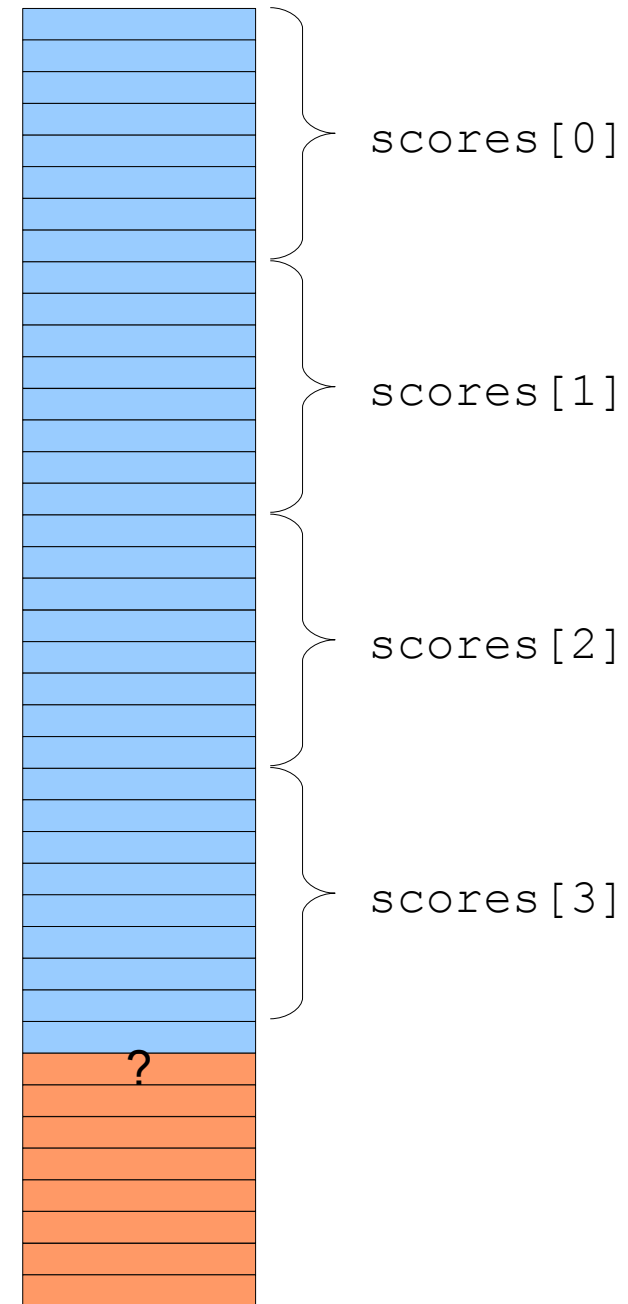
Διεύθυνση βάσης

Σχετική απόσταση



# Αναφορά σε στοιχεία έξω από τα όρια πίνακα

- Η αναφορά σε στοιχεία έξω από τα όρια ενός πίνακα αποτελεί **σφάλμα** που δεν ανιχνεύεται από όλους τους μεταγλωττιστές.
- Για παράδειγμα η αναφορά **scores[4]**
- βρίσκεται έξω από τα όρια του πίνακα scores (μεγέθους 4) και η τιμή της είναι απροσδιόριστη.



# Πέρασμα πίνακα ως παραμέτρου σε συνάρτηση

- Ένας πίνακας είναι δυνατόν να αποτελεί όρισμα μιας συνάρτησης.
- Η μορφή του πρωτοτύπου μιας συνάρτησης με όρισμα έναν μονοδιάστατο πίνακα είναι η εξής:

```
int f(int a[], int n);
```

Ενώ η κλήση της συνάρτησης γίνεται ως εξής:

```
int myArray[10];
```

```
f(myArray, 10);
```

- Στην πραγματικότητα, περνιέται ως παράμετρος η **διεύθυνση βάσης** του πίνακα.

# Πέρασμα πίνακα ως παραμέτρου σε συνάρτηση (2)

- Προσοχή: οι πίνακες περνιούνται ως παράμετροι **με αναφορά**, δηλαδή είναι δυνατή η αλλαγή των στοιχείων του πίνακα που αποτελεί όρισμα στην κλήση μιας συνάρτησης.

# Αρχικοποίηση πίνακα κατά τη δήλωσή του

Κατά τη δήλωση του παρακάτω πίνακα `digits` δίνονται αρχικές τιμές στα στοιχεία του.

```
int digits[10] =  
    {1,2,3,4,5,6,7,8,9,10};
```

ή

```
int digits[] =  
    {1,2,3,4,5,6,7,8,9,10};
```

# Παράδειγμα (Εν. 11.3)

- Αντιστροφή Πίνακα

Ζητείται η κατασκευή ενός προγράμματος το οποίο

1. Διαβάζει μια λίστα ακεραίων μέχρι ο χρήστης να καταχωρήσει την **τιμή-φρουρό (sentinel)** 0.
2. Αντιστρέφει τα στοιχεία της λίστας
3. Εμφανίζει την αντεστραμμένη λίστα .



# Αναλυτική σχεδίαση του προγράμματος

- Το πρόγραμμα αυτό αναφέρεται σε ένα “πρόβλημα” το οποίο αναλύεται στα εξής υποπρόβλήματα:
  1. Ανάγνωση των στοιχείων της λίστας
  2. Αντιστροφή της λίστας
  3. Εκτύπωση των στοιχείων της λίστας.

Για κάθε υποπρόβλημα ορίζουμε μια αντίστοιχη διαδικασία ή συνάρτηση και προκύπτει ο ακόλουθος σκελετός προγράμματος:

# Ένας σκελετός προγράμματος (μπορεί να αλλάξει)

```
main()  
{  
    int list[NElements];  
  
    GetIntegerArray(list);  
    ReverseIntegerArray(list);  
    PrintIntegerArray(list);  
}
```

# Συναρτήσεις με ορίσματα πίνακες

- Έστω η διαδικασία

```
void PrintIntegerArray(int list[]);
```

- Η παραπάνω διαδικασία δεν είναι δυνατόν να χρησιμοποιηθεί για πίνακα του οποίου ο αριθμός των στοιχείων δεν είναι γνωστός.
- Μια πιο γενική μορφή της είναι η παρακάτω:

```
void PrintIntegerArray(int list[], int n);
```
- Το δεύτερο όρισμα, *n*, ορίζει τον αριθμό των στοιχείων του πίνακα που θα χρησιμοποιηθούν (**τρέχον μέγεθος** του πίνακα).
- Το τρέχον μέγεθος πρέπει να είναι μικρότερο από το μέγεθος που καθορίζεται στη δήλωσή του (**κατανεμημένο μέγεθος**)

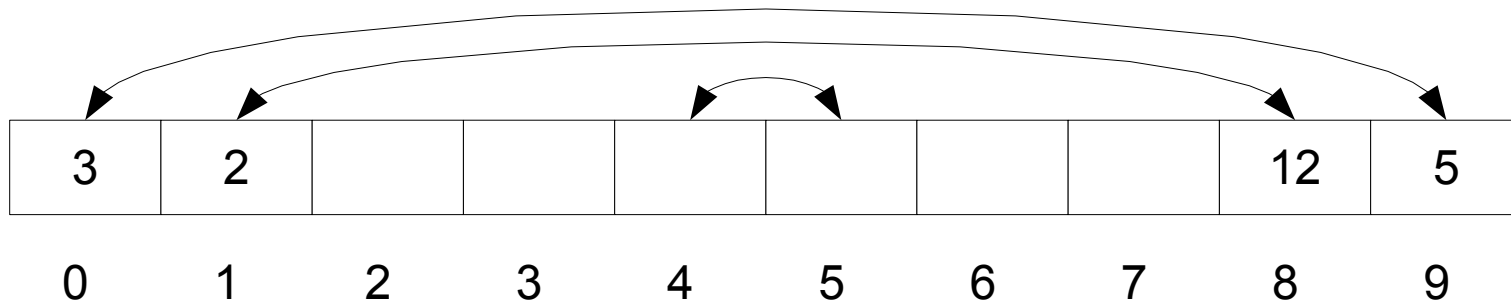
- Τα παραπάνω ισχύουν και για την συνάρτηση `ReverseIntegerArray`, της οποίας το πρωτότυπο γράφεται

```
void ReverseIntegerArray(int list[],  
    int n);
```

- Αντίθετα, η συνάρτηση `GetIntegerArray` δεν ακολουθεί τον παραπάνω “κανόνα”, γιατί όταν καλείται δεν είναι γνωστό το τρέχον μέγεθος του πίνακα. Έτσι, γράφεται

```
int GetIntegerArray(int array[], int  
    max, int sentinel);
```

# Η συνάρτηση αντιστροφής



```
static void ReverseIntegerArray(int array[],
    int n)
{
    int i;

    for (i = 0; i < n / 2; i++) {
        SwapIntegerElements(array, i, n - i - 1);
    }
}
```

# Το πλήρες πρόγραμμα αντιστροφής

- Η συνάρτηση SwapIntegerElements

```
static void SwapIntegerElements (int
    array[], int p1, int p2)
{
    int tmp;

    tmp = array[p1];
    array[p1] = array[p2];
    array[p2] = tmp;
}
```

- Το **πλήρες πρόγραμμα** αντιστροφής πίνακα.

# Παράδειγμα μέτρησης γραμμάτων

- Το πρόγραμμα `countlet.c`.
- Για το χειρισμό χαρακτήρων και συμβολοσειρών βλ. κεφ. 9.

# Λύση του προβλήματος

- Μέτρηση των γραμμάτων σε γραμμές.
  - Ανάγνωση γραμμής
  - Καταγραφή των γραμμάτων της γραμμής
- Εμφάνιση της συχνότητας εμφάνισης των γραμμάτων



```
main()
{
    int letterCounts[NLetters];

    printf("This program counts letter frequencies.\n");
    printf("Enter a blank line to signal end of
input.\n");

    ClearIntegerArray(letterCounts, NLetters);
    CountLetters(letterCounts);
    DisplayLetterCounts(letterCounts);
}
```

# Η συνάρτηση `ClearIntegerArray`

```
void ClearIntegerArray(int array[], int n)
{
    int i;

    for (i = 0; i < n; i++) {
        array[i] = 0;
    }
}
```

# Η συνάρτηση CountLetters

```
static void CountLetters(int letterCounts[])
{
    string line;

    while (TRUE) {
        line = GetLine();
        if (StringLength(line) == 0) break;
        CountLettersInString(line,
letterCounts);
    }
}
```

# Η συνάρτηση CountLettersInString

```
static void CountLettersInString(string str,
    int letterCounts[])
{
    int i;

    for (i = 0; i < StringLength(str); i++) {
        RecordLetter(IthChar(str, i),
            letterCounts);
    }
}
```

# Η συνάρτηση RecordLetter

```
void RecordLetter(char ch, int
    letterCounts[])
{
    int index;

    index = LetterIndex(ch);
    if (index != -1) letterCounts[index]++;
}
```

# Η συνάρτηση LetterIndex

```
int LetterIndex(char ch)
{
    if (isalpha(ch)) {
        return (toupper(ch) - 'A');
    } else {
        return (-1);
    }
}
```