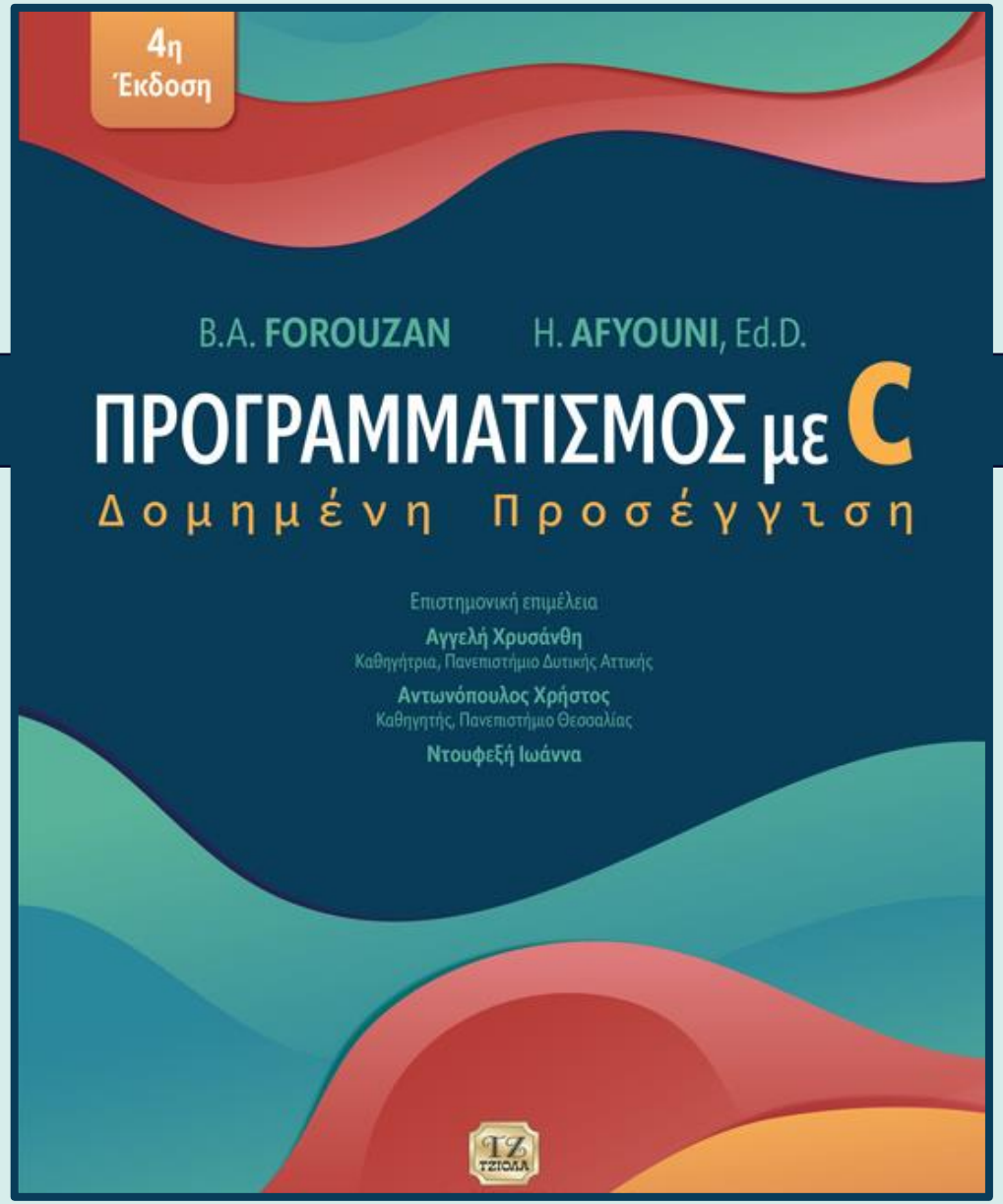


Επιστημονικές Εκδόσεις
ΤΖΙΟΛΑ

ΚΕΦΑΛΑΙΟ 11

Τύποι Απαρίθμησης, Δομής και Ένωσης

ΘΕΜΑΤΑ ΚΕΦΑΛΑΙΟΥ

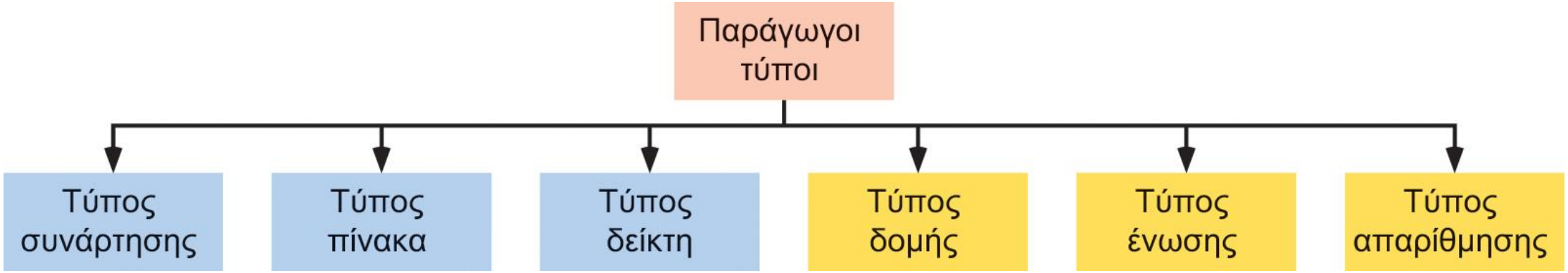
- Ορισμός τύπων σε προγράμματα
- Χρήση τύπων απαρίθμησης, συμπεριλαμβανομένων των ανώνυμων τύπων, σε προγράμματα
- Δημιουργία και χρήση δομών σε προγράμματα
- Δημιουργία και χρήση ενώσεων σε προγράμματα

11.1
Ορισμός τύπου (`typedef`)

Ορισμός τύπου (`typedef`) (1/3)

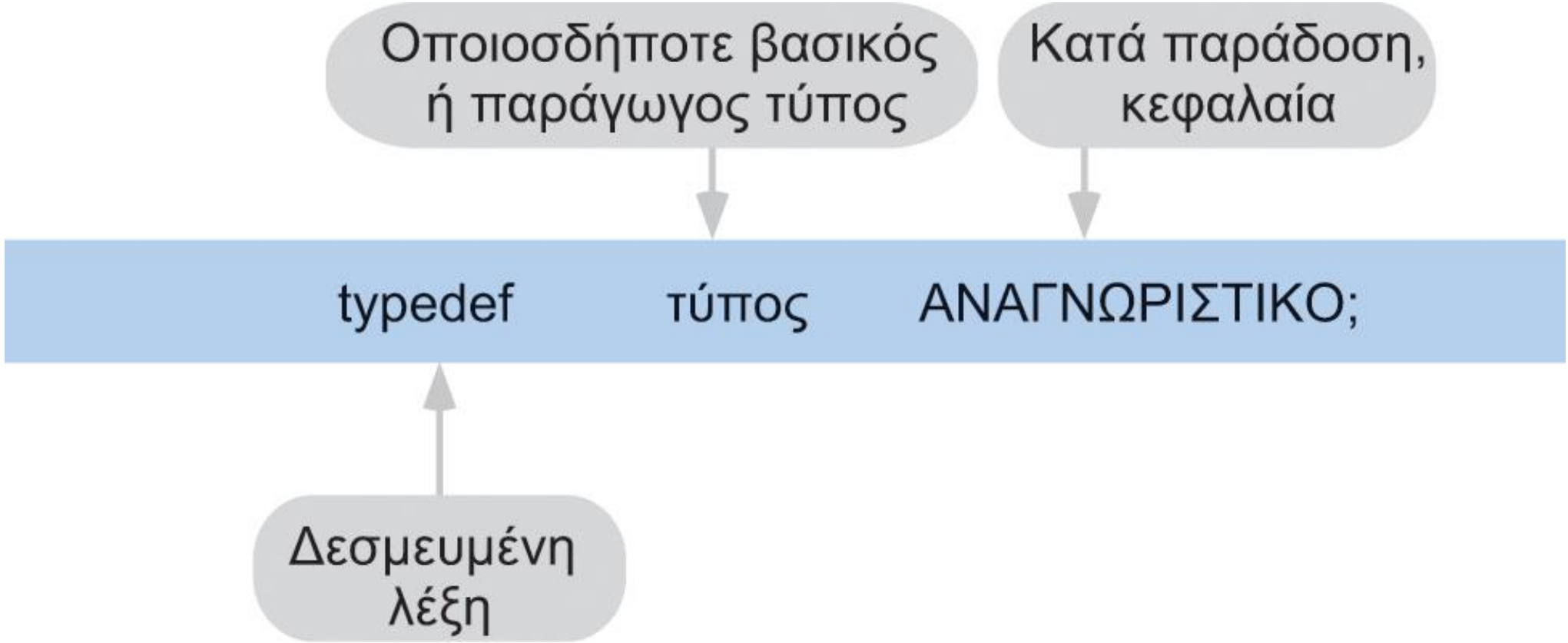
- Ένας ορισμός τύπου μέσω της εντολής typedef «ονοματίζει» και ουσιαστικά δημιουργεί ένα συνώνυμο ενός τύπου δεδομένων που μπορεί στη συνέχεια να χρησιμοποιηθεί οπουδήποτε επιτρέπεται/απαιτείται ένας τύπος δεδομένων
  - Μπορούμε να χρησιμοποιήσουμε τη δυνατότητα ορισμού τύπου με οποιονδήποτε εγγενή τύπο δεδομένων
- Σημειώστε ότι το αναγνωριστικό που ορίζεται στην typedef γράφεται κατά παράδοση με κεφαλαία γράμματα
  - Αυτό προειδοποιεί τον αναγνώστη ότι υπάρχει κάτι ασυνήθιστο σχετικά με τον συγκεκριμένο τύπο

11.1 Ορισμός τύπου (typedef) (2/3)



ΣΧΗΜΑ 11-1 Παράγωγοι τύποι.

11.1 Ορισμός τύπου (`typedef`) (3/3)



ΣΧΗΜΑ 11-2 Ορισμός τύπου με την typedef.

11.2
Τύποι απαρίθμησης

11.2 Τύποι απαρίθμησης (1/2)

- Ένας τύπος απαρίθμησης (enumerated type) ορίζεται από τον χρήστη με βάση τον εγγενή τύπο ακεραίων (int) της C
- Σε έναν τύπο απαρίθμησης, κάθε ακέραια τιμή αντιστοιχίζεται σε ένα αναγνωριστικό που ονομάζεται σταθερά απαρίθμησης (enumeration constant)
  - Μπορούμε να χρησιμοποιούμε τις σταθερές απαρίθμησης ως συμβολικά ονόματα, πράγμα το οποίο μπορεί να κάνει τα προγράμματά μας πολύ πιο ευανάγνωστα

11.2 Τύποι απαρίθμησης (2/2)

- Ενώ τα ονόματα, οι τιμές και οι πράξεις για τους εγγενείς τύπους ορίζονται από το σύστημα και τη γλώσσα C, για τους τύπους που δημιουργούμε πρέπει να τα ορίσουμε εμείς
  - Όταν ένας καθοριζόμενος από τον χρήστη τύπος «μεταφράζεται» απευθείας σε έναν εγγενή τύπο της C, όπως ισχύει για τους τύπους απαρίθμησης, ο μεταγλωττιστής μπορεί να τον αντιστοιχίζει αυτόματα στον εγγενή τύπο

Δήλωση ενός τύπου απαρίθμησης

- Για να δηλώσουμε έναν τύπο απαρίθμησης, πρέπει να δηλώσουμε το αναγνωριστικό του και τις τιμές του
  - Επειδή προέρχεται από τον εγγενή τύπο ακεραίων της C, οι πράξεις του είναι ίδιες με αυτές του τύπου int

`enum` όνομαΤύπου { λίστα απαρίθμησης };

Πράξεις σε τύπους απαρίθμησης

- Ανάθεση τιμών σε τύπους απαρίθμησης
  - Αφού δηλωθεί μία μεταβλητή με τύπο απαρίθμησης, μπορούμε να αποθηκεύσουμε τιμές σε αυτήν
- Σύγκριση τύπων απαρίθμησης
  - Μπορούμε να συγκρίνουμε μεταβλητές ενός τύπου απαρίθμησης, χρησιμοποιώντας τους γνωστούς τελεστές σύγκρισης
- Άλλες πράξεις σε τύπους απαρίθμησης
  - Γενικά, κάθε πράξη που ορίζεται για τους ακέραιους αριθμούς μπορεί να χρησιμοποιηθεί με τύπους απαρίθμησης

Μετατροπή τύπων απαρίθμησης

- Η C επιτρέπει τη μετατροπή τύπων απαρίθμησης, εμμέσως και ρητά
  - Ο μεταγλωττιστής μετατρέπει εμμέσως έναν τύπο απαρίθμησης σε ακέραιο, όταν απαιτείται
    - Ωστόσο, όταν επιχειρούμε το αντίστροφο—έμμεση μετατροπή ακεραίου σε τύπο απαρίθμησης—λαμβάνουμε ένα μήνυμα προειδοποίησης ή σφάλματος, ανάλογα με τον μεταγλωττιστή
  - Μπορούμε να μετατρέψουμε ρητά έναν ακέραιο σε έναν τύπο απαρίθμησης χωρίς προβλήματα

Αρχικοποίηση σταθερών απαρίθμησης

- Ο μεταγλωττιστής αναθέτει αυτόματα ακέραιες τιμές στις σταθερές των τύπων απαρίθμησης, ξεκινώντας από το 0
  - Μπορούμε να παρακάμψουμε αυτή τη συμπεριφορά και να αναθέσουμε διαφορετικές τιμές
  - Δεν χρειάζεται να αναθέσουμε τιμές αρχικοποίησης σε κάθε αναγνωριστικό του τύπου απαρίθμησης
  - Εάν παραλείψουμε κάποιες τιμές αρχικοποίησης, ο μεταγλωττιστής αναθέτει σε αυτές τον επόμενο ακέραιο, προσθέτοντας 1

```
enum mycolor {RED, ROSE = 0, CRIMSON = 0, SCARLET = 0,
              BLUE, AQUA = 1, NAVY = 1,
              GREEN, JADE = 2, WHITE};
```

Ανώνυμοι τύποι απαρίθμησης

- Μπορούμε να δημιουργήσουμε έναν ανώνυμο τύπο απαρίθμησης, παραλείποντας το όνομα στον ορισμό του
  - Επειδή τα αναγνωριστικά στους τύπους απαρίθμησης είναι σταθερές, οι τύποι απαρίθμησης εξυπηρετούν ως ένας βολικός τρόπος δήλωσης σταθερών

```
enum {space = ' ', comma = ',', colon = ':', ...};
```

Λειτουργίες εισόδου/εξόδου

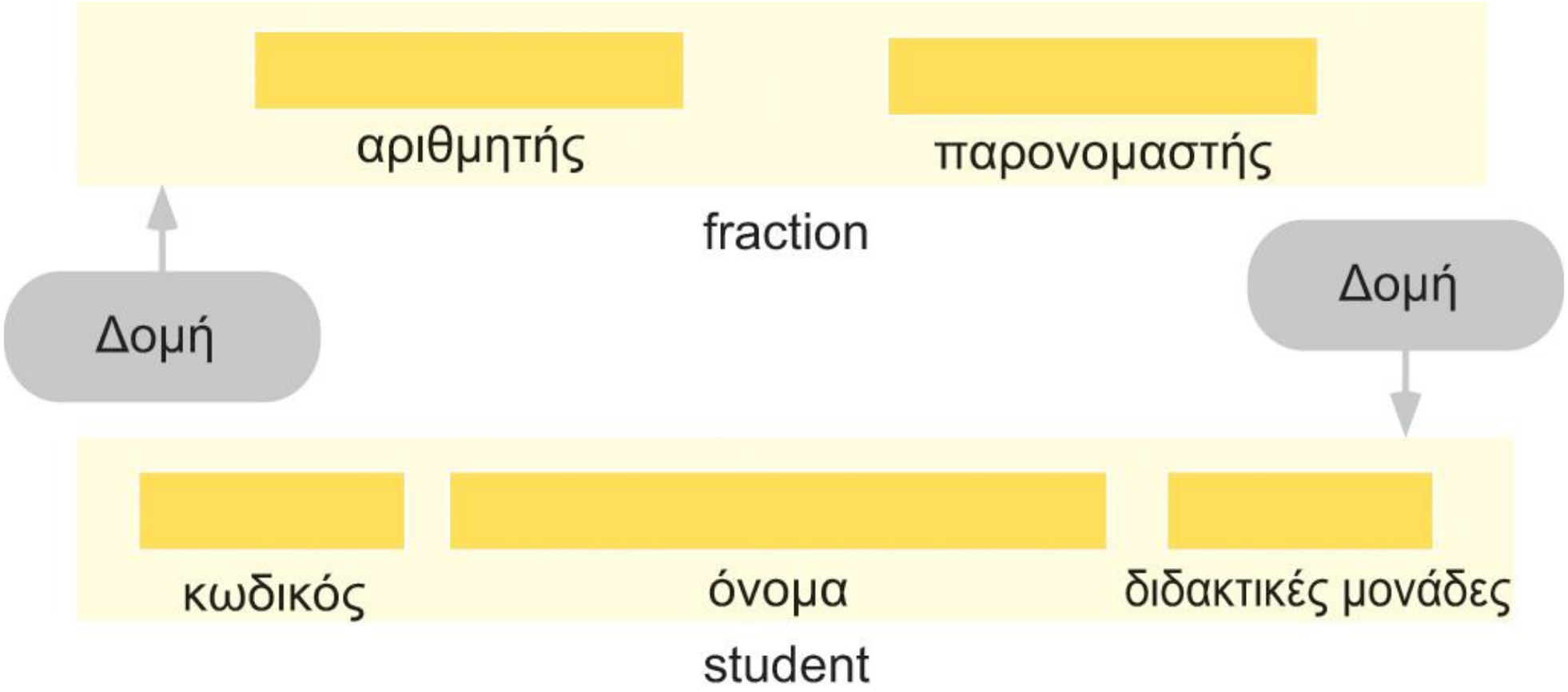
- Επειδή οι τύποι απαρίθμησης είναι παράγωγοι τύποι, δεν είναι δυνατή η ανάγνωση και εγγραφή δεδομένων σε αυτούς με τις συναρτήσεις μορφοποιημένης εισόδου/εξόδου που διαθέτει η C
  - Όταν διαβάζουμε τα στοιχεία ενός τύπου απαρίθμησης, θα πρέπει να τα διαβάζουμε ως ακεραίους
  - Όταν γράφουμε σε έναν τύπο απαρίθμησης, τα δεδομένα εμφανίζονται ως ακέραιοι
  - Όταν είναι απαραίτητο, η C μετατρέπει εμμέσως τους τύπους των μεταβλητών

11.3
Δομές

11.3 Δομές (1/2)

- Μία δομή (structure) είναι μία συλλογή σχετιζόμενων στοιχείων, ενδεχομένως διαφορετικών τύπων δεδομένων, τα οποία αναπαρίστανται συλλογικά με ένα μεμονωμένο όνομα
- Κάθε στοιχείο ή μέλος μιας δομής ονομάζεται πεδίο (field)
  - Ένα πεδίο είναι το μικρότερο στοιχείο «επώνυμων» δεδομένων που έχει νόημα ως οντότητα
  - Διαθέτει πολλά από τα χαρακτηριστικά των μεταβλητών
  - Έχει έναν τύπο δεδομένων και υπάρχει στη μνήμη
  - Μπορεί να λάβει τιμές, οι οποίες, με τη σειρά τους, μπορούν να προσπελάζονται για ανάγνωση ή αλλαγή

11.3 Δομές (2/2)



ΣΧΗΜΑ 11-3 Παραδείγματα δομών.

Δήλωση τύπου δομής (1/3)

- Όμοια με όλους τους τύπους δεδομένων, οι δομές πρέπει να δηλώνονται και να ορίζονται στα προγράμματα που τις χρησιμοποιούν
- Ο πρώτος τρόπος ορισμού μιας δομής συνίσταται στη χρήση μιας χαρακτηρισμένης δομής (tagged structure)
  - Μία χαρακτηρισμένη δομή μπορεί να χρησιμοποιηθεί για τον ορισμό μεταβλητών, παραμέτρων και τύπων επιστροφής σε εντολές return

ΣΧΗΜΑ 11-4 Ορισμός χαρακτηρισμένης δομής.

```
struct TAG
{
    λίστα πεδίων
};
```

Σύνταξη

```
struct STUDENT
{
    char id[10];
    char name[26];
    int  gradePts;
}; // δομή STUDENT
```

Παράδειγμα

Δήλωση τύπου δομής (2/3)

- Ο πιο ισχυρός τρόπος για να δηλώσετε μία δομή είναι να χρησιμοποιήσετε έναν ορισμό τύπου, δηλαδή την εντολή typedef

```
typedef struct
{

    λίστα πεδίων

} TYPE;
```

Σύνταξη

```
typedef struct
{
    char id[10];
    char name[26];
    int  gradePts;
} STUDENT;
```

Παράδειγμα

ΣΧΗΜΑ 11-5 Δήλωση δομής με την εντολή typedef.

Δήλωση τύπου δομής (3/3)

- Αφού οριστεί μία δομή, μπορούμε να δηλώσουμε μεταβλητές που να τη χρησιμοποιούν ως τύπο

```
// Καθολικές δηλώσεις τύπων
struct STUDENT
{
    char id[10];
    char name[26];
    int  gradePts;
};

// Τοπικές δηλώσεις
struct STUDENT aStudent;
```

```
// Καθολικές δηλώσεις τύπων
typedef struct
{
    char id[10];
    char name[26];
    int  gradePts;
} STUDENT;

// Τοπικές δηλώσεις
STUDENT aStudent;
```

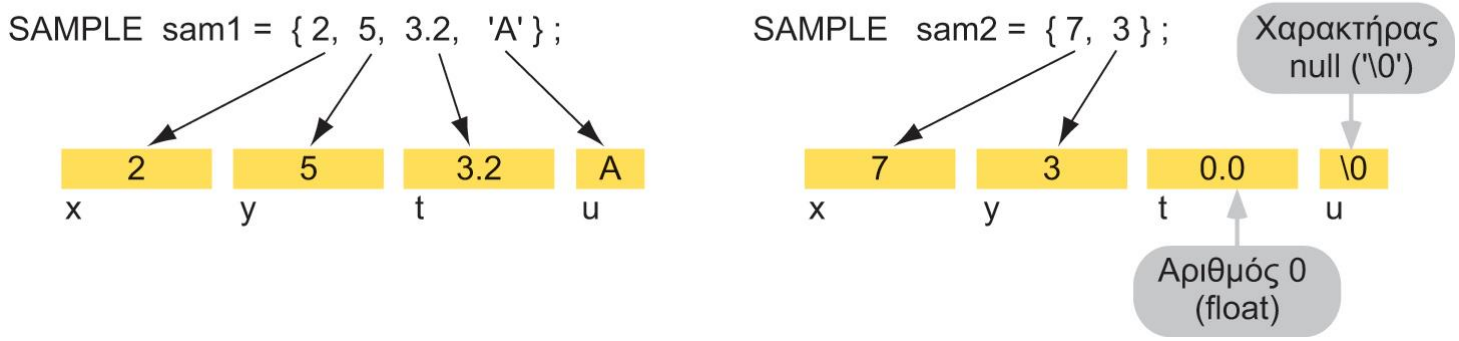
ΣΧΗΜΑ 11-6 Ορισμός και χρήση μιας δομής.

Αρχικοποίηση δομής

- Μπορούμε να αρχικοποιήσουμε τα μέλη μιας δομής με τιμές
  - Οι κανόνες για την αρχικοποίηση δομών είναι παρόμοιοι με αυτούς που ισχύουν για την αρχικοποίηση πινάκων

```
typedef struct
{
    int   x;
    int   y;
    float t;
    char  u;
} SAMPLE;
```

ΣΧΗΜΑ 11-7
Αρχικοποίηση δομών.

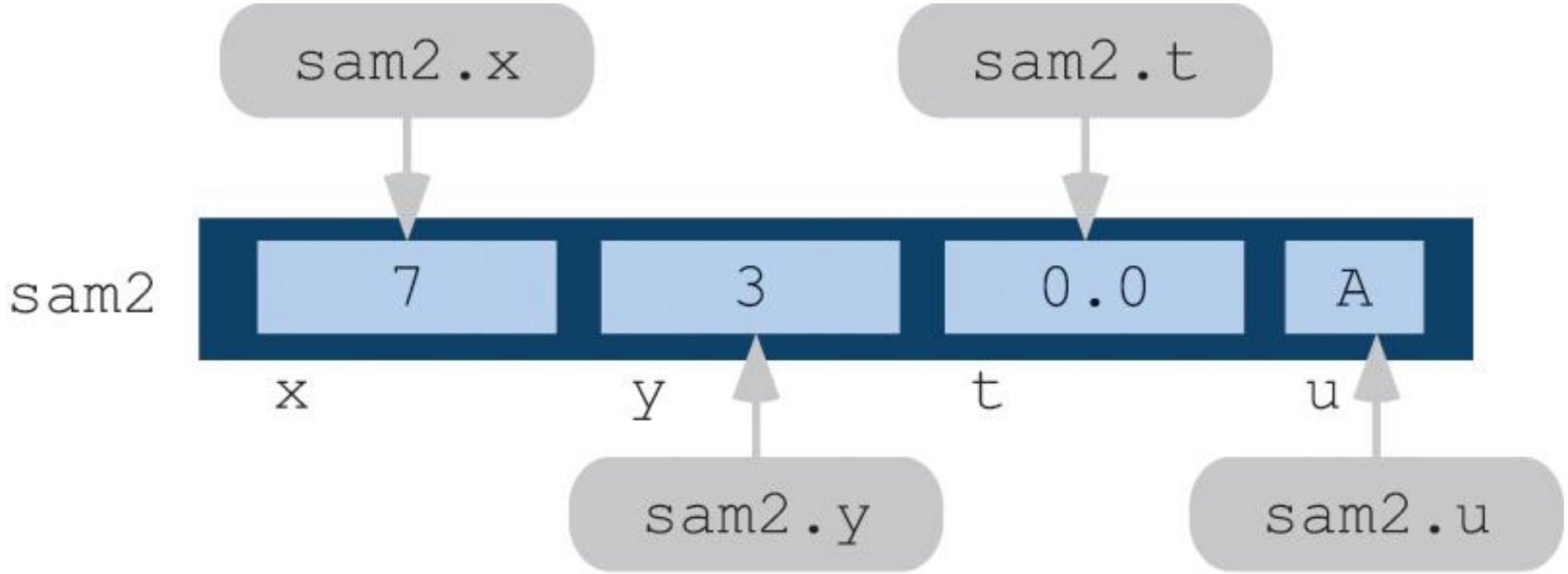


Προσπέλαση δομών και μεμονωμένων μελών τους (1/2)

- Αναφορά σε μεμονωμένα πεδία
  - Για να προσπελάσουμε ένα πεδίο μιας δομής πρέπει να αναφέρουμε τόσο τη δομή όσο και το συγκεκριμένο πεδίο
  - Η C χρησιμοποιεί μία σύμβαση ιεραρχικής ονοματοδοσίας, βάσει της οποίας αναφέρεται πρώτα το αναγνωριστικό της μεταβλητής τύπου δομής και κατόπιν το αναγνωριστικό του πεδίου
  - Το αναγνωριστικό μιας μεταβλητής τύπου δομής (structure variable) διαχωρίζεται από το αναγνωριστικό του πεδίου με μία τελεία

Προσπέλαση δομών και μεμονωμένων μελών τους (2/2)

- Προτεραιότητα του τελεστή άμεσης επιλογής
  - Ο τελεστής άμεσης επιλογής (.) είναι επιθεματικός και έχει προτεραιότητα 16

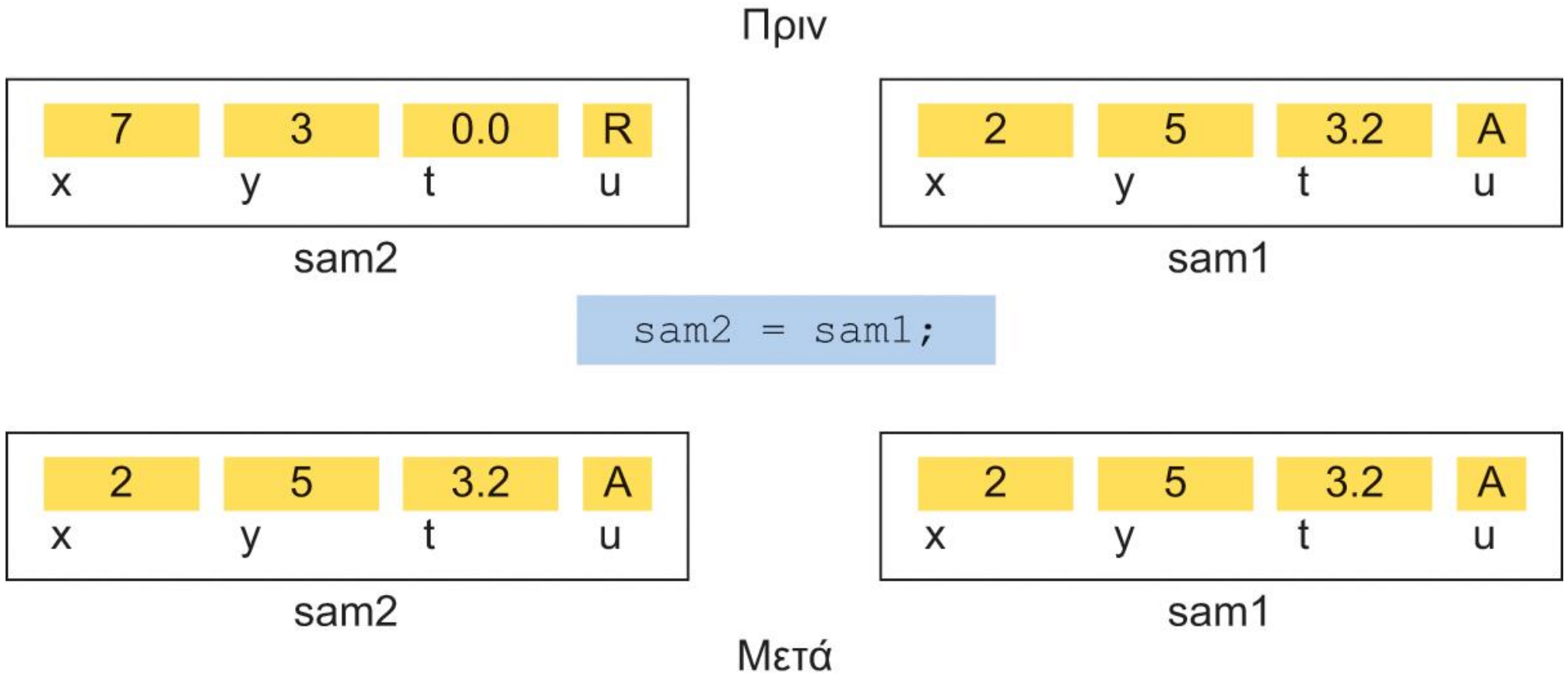


ΣΧΗΜΑ 11-8 Χρήση του τελεστή άμεσης επιλογής για την προσπέλαση μελών μιας δομής.

Πράξεις με δομές (1/5)

- Η δομή είναι μία οντότητα που μπορεί να αντιμετωπιστεί ως όλον
  - Όταν η δομή αντιμετωπίζεται ως όλον, επιτρέπεται μόνο μία πράξη, η ανάθεση
  - Μία δομή μπορεί μόνο να αντιγραφεί σε μία άλλη δομή του ίδιου τύπου χρησιμοποιώντας τον τελεστή ανάθεσης
  - Αντί να αναθέτουμε τιμές σε μεμονωμένα μέλη, όταν θέλουμε να αντιγράψουμε μία δομή σε μία άλλη μπορούμε απλώς να εφαρμόσουμε τον τελεστή ανάθεσης σε ολόκληρες δομές

Πράξεις με δομές (2/5)

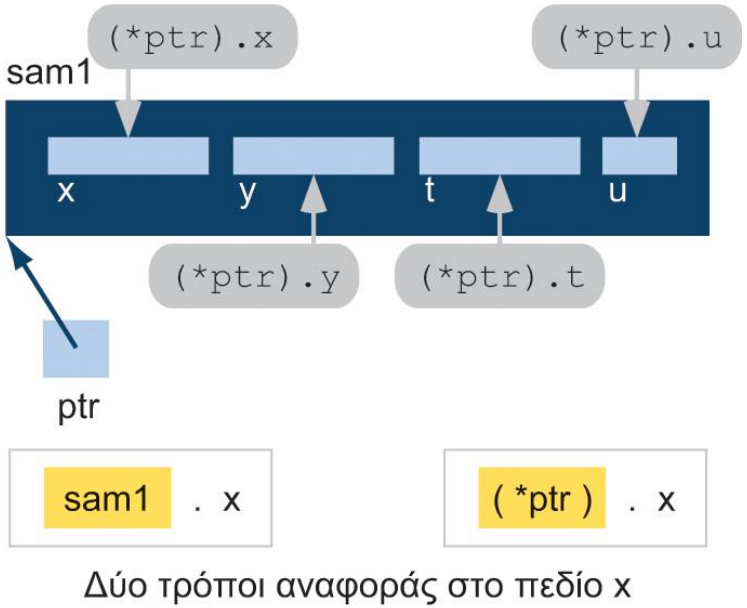


ΣΧΗΜΑ 11-9 Αντιγραφή δομής με ανάθεση.

Πράξεις με δομές (3/5)

- Δείκτες σε δομές
  - Οι δομές, όπως και άλλοι τύποι δεδομένων, μπορούν επίσης να προσπελάζονται μέσω δεικτών
    - Είναι μία από τις πιο κοινές μεθόδους που χρησιμοποιούνται για την αναφορά σε δομές

```
typedef struct
{
    int   x;
    int   y;
    float t;
    char  u;
} SAMPLE;
...
SAMPLE  sam1;
SAMPLE* ptr;
...
ptr = &sam1;
...
```

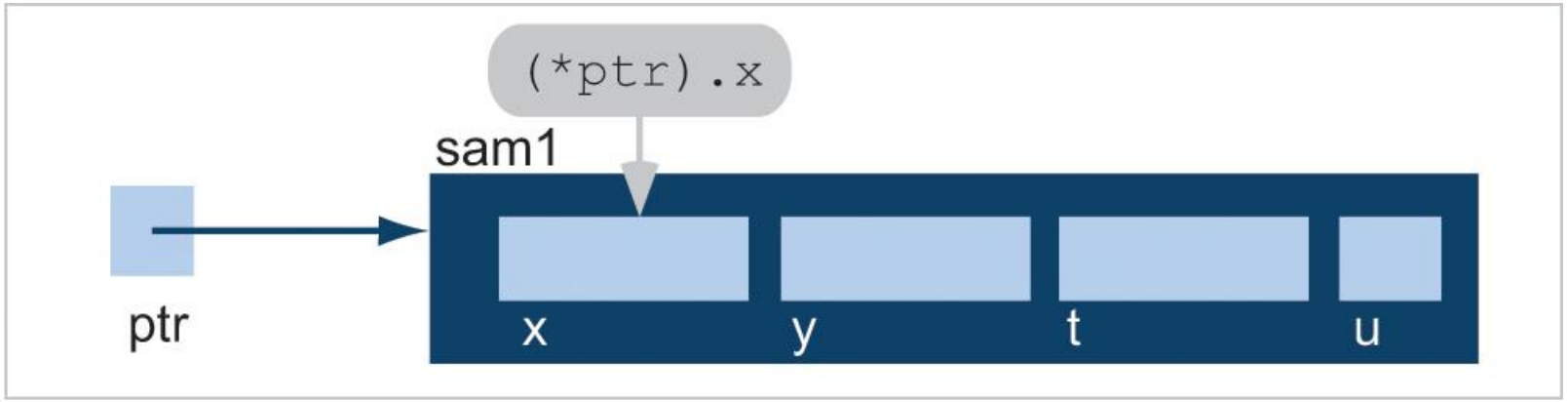


Δύο τρόποι αναφοράς στο πεδίο x

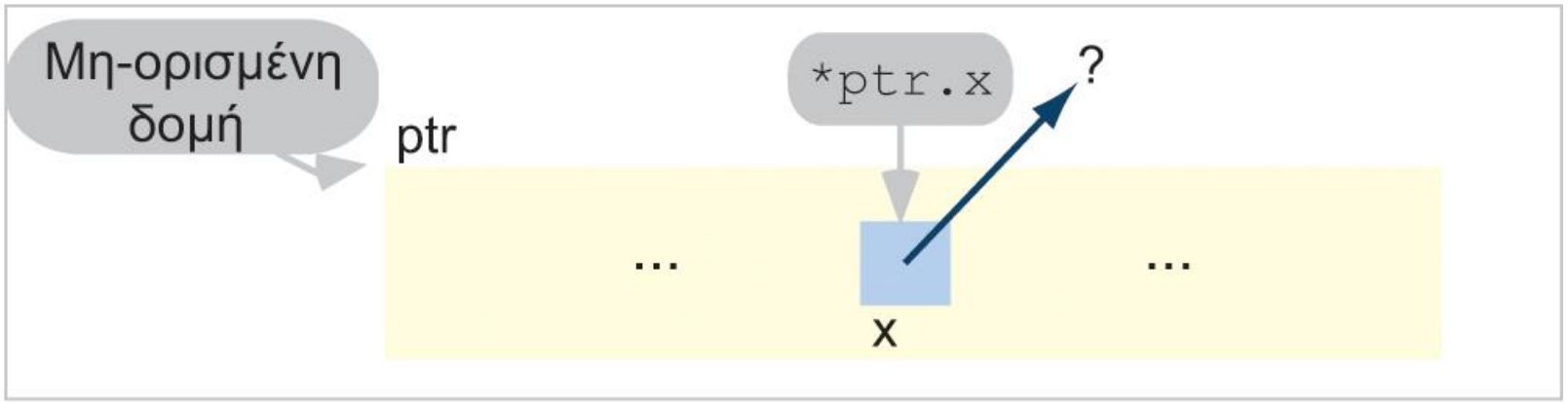
ΣΧΗΜΑ 11-10 Δείκτες σε δομές.

Πράξεις με δομές (4/5)

ΣΧΗΜΑ 11-11
Ερμηνεία της εσφαλμένης χρήσης δείκτη σε δομή.



Ορθός τρόπος αναφοράς



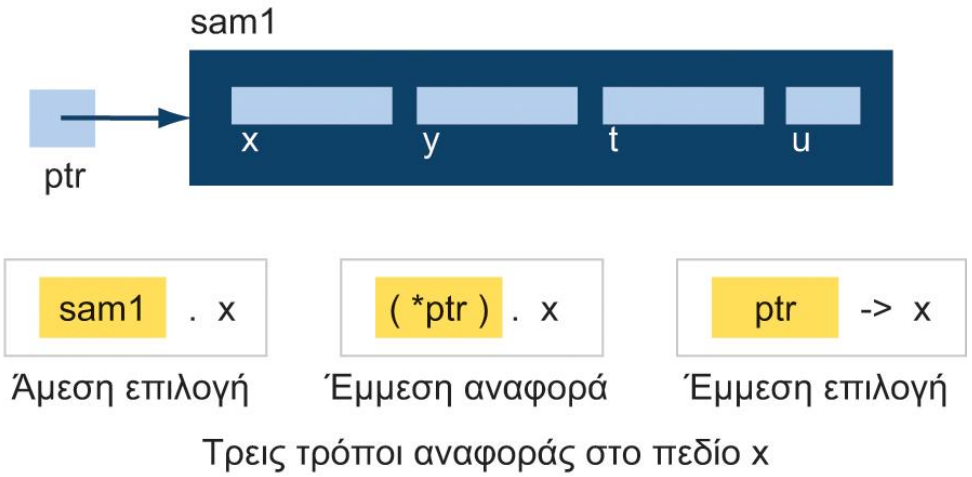
Εσφαλμένος τρόπος αναφοράς στο στοιχείο

Πράξεις με δομές (5/5)

- Τελεστής έμμεσης επιλογής
  - Ένας άλλος τελεστής της C, ο τελεστής έμμεσης επιλογής, εξαλείφει τα προβλήματα που μπορεί να προκαλέσει η χρήση δεικτών σε δομές

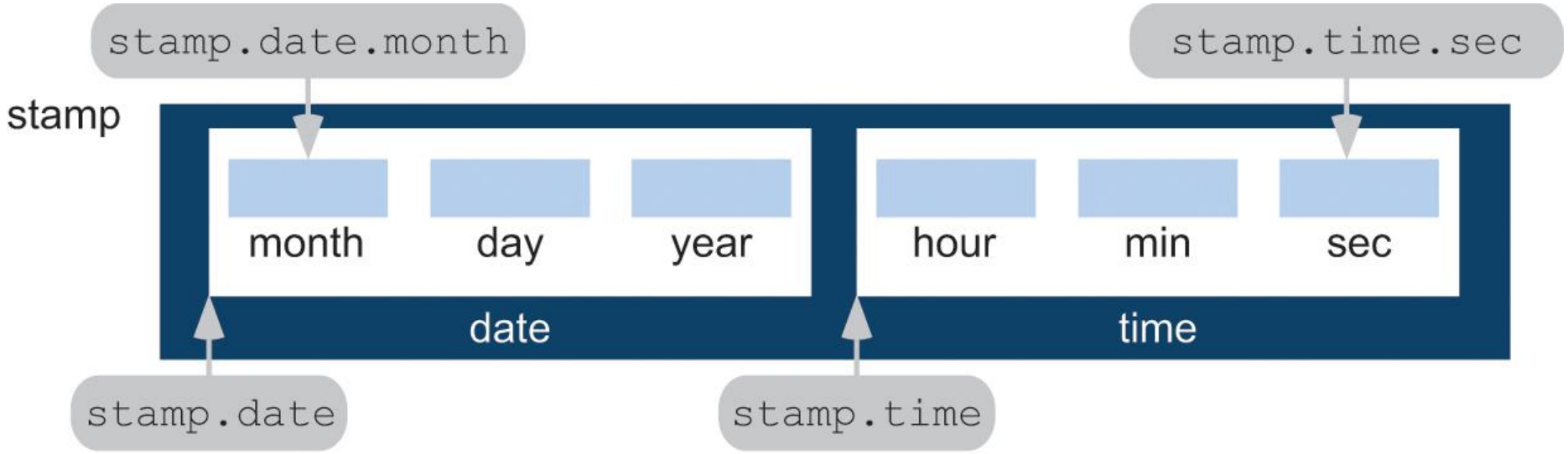
ΣΧΗΜΑ 11-12
Τελεστής έμμεσης επιλογής.

```
typedef struct
{
    int    x;
    int    y;
    float  t;
    char   u;
} SAMPLE;
...
SAMPLE  sam1;
SAMPLE* ptr;
...
ptr = &sam1;
...
```



Σύνθετες δομές (1/4)

- Ένθετες δομές
  - Μπορούμε να χρησιμοποιούμε δομές ως μέλη μιας άλλης δομής
  - Κατ' αυτό τον τρόπο δημιουργείται μία ένθετη δομή (nested structure)
  - Δεν υπάρχει όριο όσον αφορά στον αριθμό των επιπέδων ένθεσης δομών, αλλά σπάνια ξεπερνάμε τα τρία

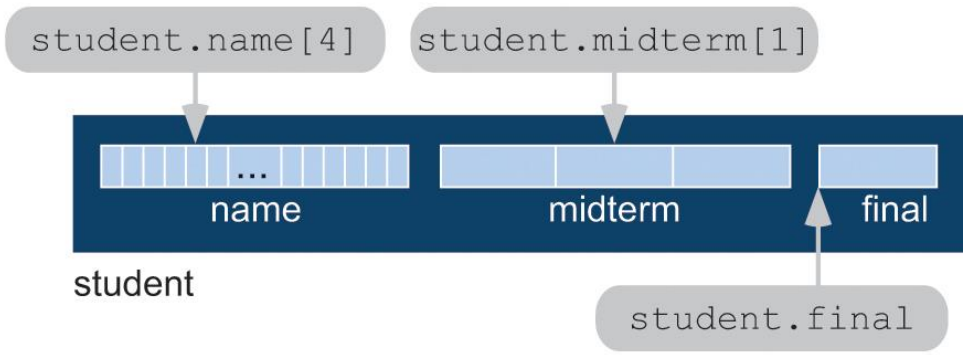


ΣΧΗΜΑ 11-13
Ένθετη δομή.

Σύνθετες δομές (2/4)

- Δομές που περιέχουν πίνακες
  - Οι δομές μπορούν να έχουν έναν ή περισσότερους πίνακες ως μέλη
  - Οι πίνακες μπορούν να προσπελάζονται είτε μέσω ενδεικτών θέσης (indexes), είτε μέσω δεικτών (pointers), αρκεί να είναι κατάλληλα χαρακτηρισμένοι με τον τελεστή άμεσης επιλογής

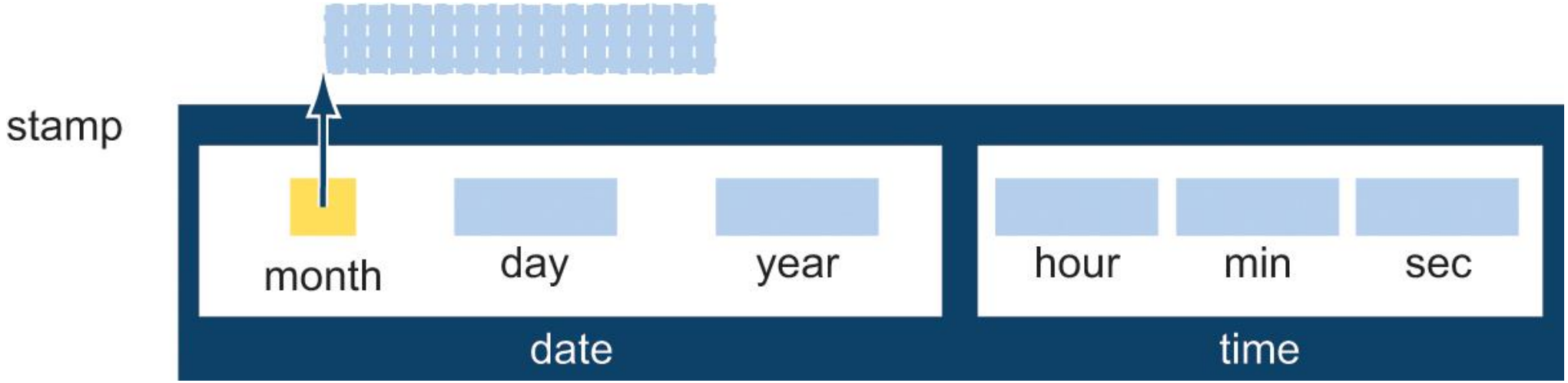
```
// Καθολικές δηλώσεις
typedef struct
{
    char name[26];
    int  midterm[3];
    int  final;
} STUDENT ;
// Τοπικές δηλώσεις
STUDENT student;
```



ΣΧΗΜΑ 11-14 Πίνακες σε δομές.

Σύνθετες δομές (3/4)

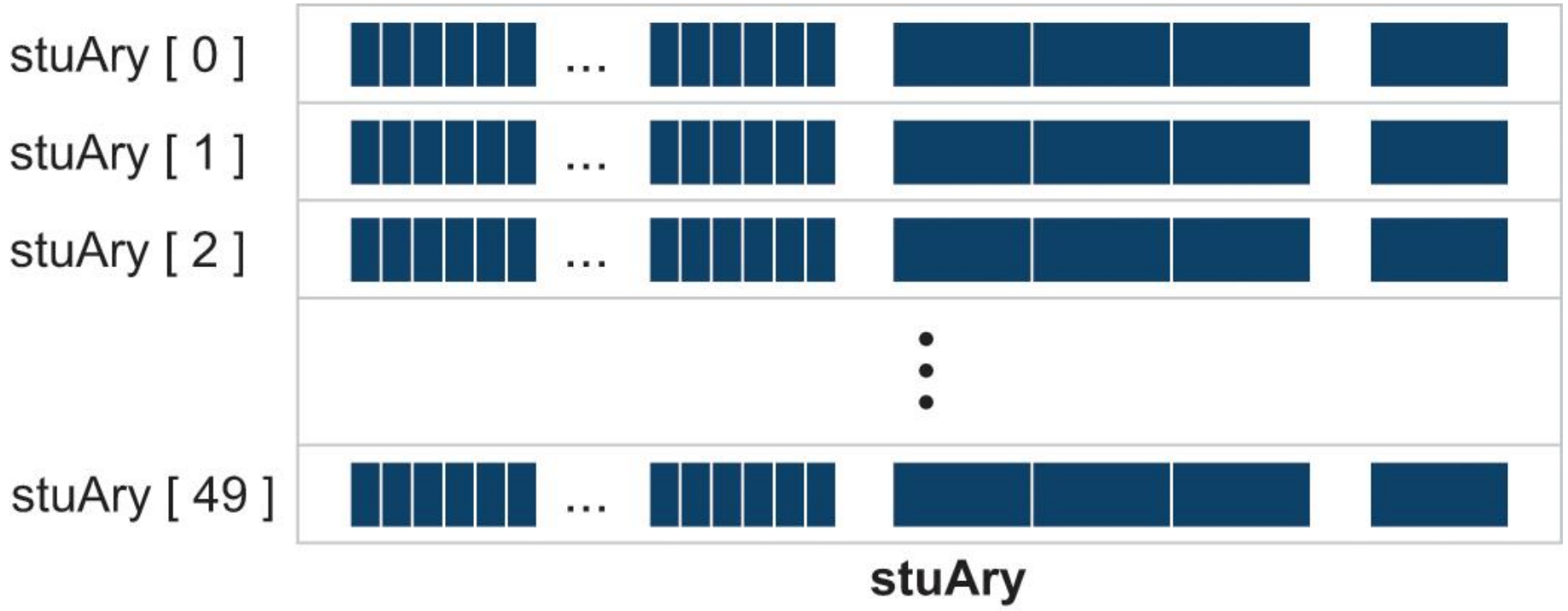
- Δομές που περιέχουν δείκτες
  - Μία δομή μπορεί να έχει δείκτες ως μέλη· η χρήση δεικτών είναι πολύ συνηθισμένη στις δομές
  - Η χρήση δεικτών μπορεί να εξοικονομήσει μνήμη



ΣΧΗΜΑ 11-15 Δείκτες σε δομές.

Σύνθετες δομές (4/4)

- Πίνακες αποτελούμενοι από δομές (πίνακες δομών)
  - Σε ορισμένες περιπτώσεις, χρειάζεται να «ομαδοποιήσουμε» πολλαπλές δομές σε έναν πίνακα



ΣΧΗΜΑ 11-16
Πίνακας δομών.

Δομές και συναρτήσεις (1/2)

- Μία συνάρτηση μπορεί να προσπελάζει τα μέλη μιας δομής με τρεις τρόπους:

    1. Πέρασμα μεμονωμένων μελών της δομής στη συνάρτηση

    2. Πέρασμα ολόκληρης της δομής στη συνάρτηση· σε αυτή την περίπτωση, τα μέλη της δομής μπορούν να προσπελάζονται χρησιμοποιώντας τον μηχανισμό περάσματος διά τιμής

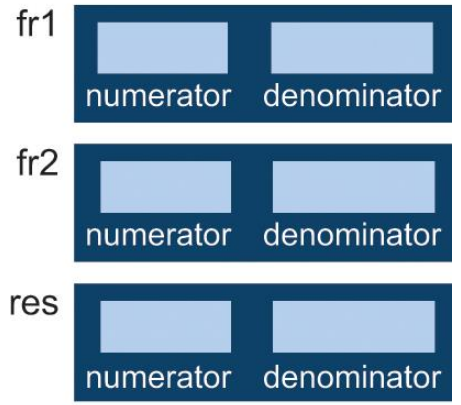
    3. Πέρασμα της διεύθυνσης μιας δομής ή ενός μέλους δομής στη συνάρτηση· η πρόσβαση στα μέλη της δομής γίνεται μέσω τελεστών έμμεσης αναφοράς και έμμεσης επιλογής

Δομές και συναρτήσεις (2/2)

- Το πέρασμα μεμονωμένων μελών μιας δομής σε μία συνάρτηση δεν διαφέρει απ' ό,τι γνωρίζουμε και εφαρμόζουμε ήδη

```
...
res.numerator =
   multiply(frl.numerator, fr2.numerator)
res.denominator =
   multiply(frl.d.denominator, fr2.denominator);
...
```

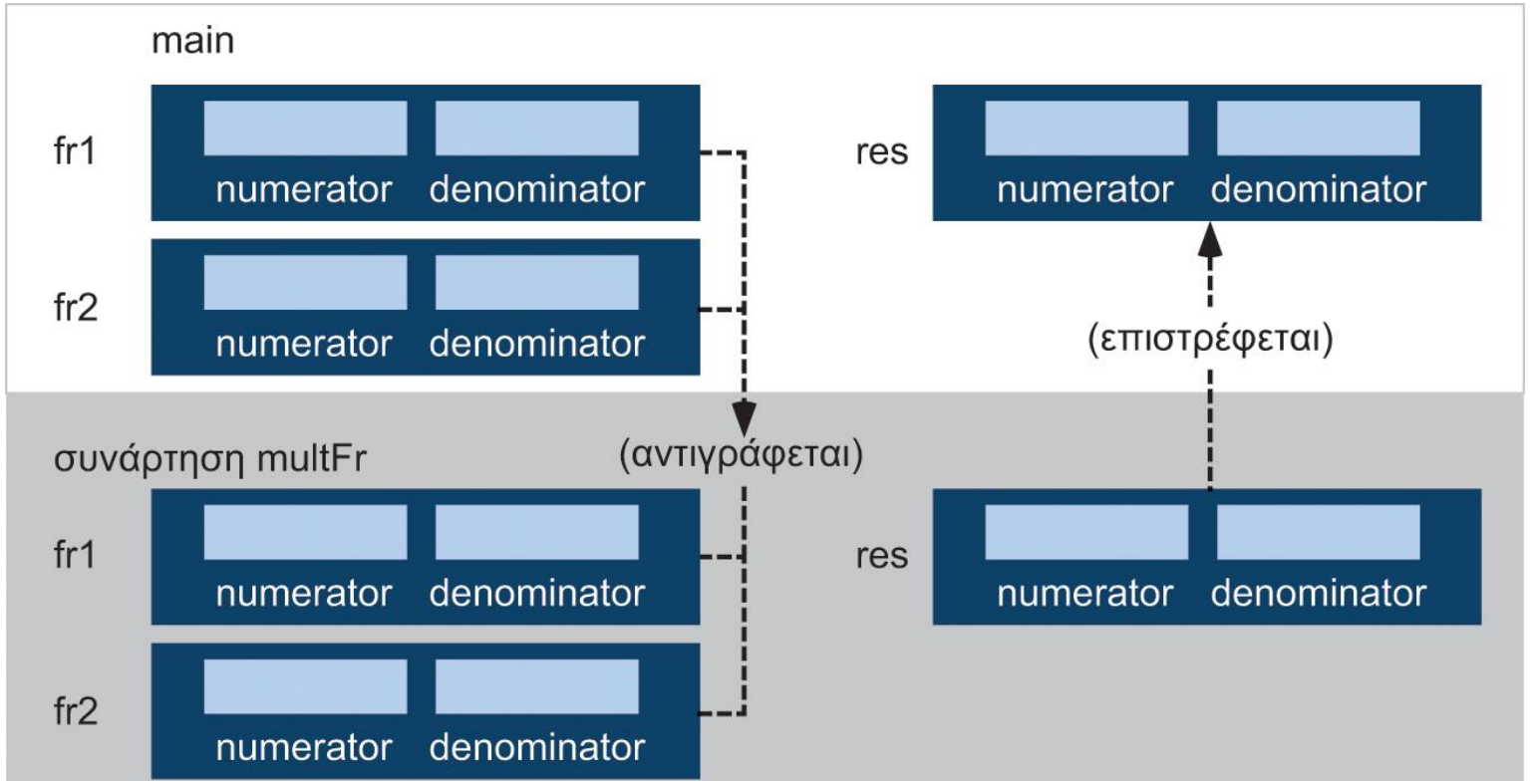
```
// ========== συνάρτηση multiply ==========
int multiply (int x, int y)
{
  return x * y;
}// τέλος multiply
```



ΣΧΗΜΑ 11-17 Πέρασμα μελών μιας δομής σε συναρτήσεις.

Πέρασμα ολόκληρων δομών

- Ουσιαστικά, το πέρασμα μιας ολόκληρης δομής δεν διαφέρει από το πέρασμα μεμονωμένων στοιχείων
  - Εφόσον η δομή είναι ένας τύπος δεδομένων, απλά χρησιμοποιούμε αυτό τον τύπο στις τυπικές παραμέτρους της καλούμενης συνάρτησης

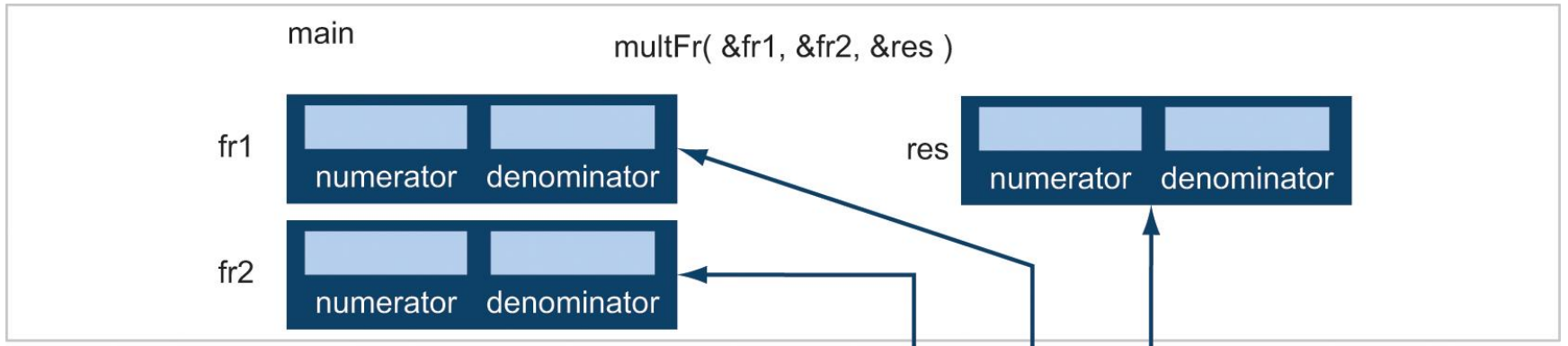


ΣΧΗΜΑ 11-18 Πέρασμα και επιστροφή δομών.

Πέρασμα δομών με χρήση δεικτών (1/2)

- Όταν η δομή είναι πολύ μεγάλη, η αποδοτικότητα μπορεί να υποβαθμιστεί, ειδικά σε περιπτώσεις συναρτήσεων που χρησιμοποιούνται πολλές φορές κατά τη διάρκεια του προγράμματος
  - Για το λόγο αυτό, χρησιμοποιούμε συχνά δείκτες για το πέρασμα δομών σε συναρτήσεις
  - Είναι επίσης σύνηθες να περνάμε δομές μέσω δεικτών όταν η δομή βρίσκεται στη δυναμική μνήμη
  - Σε αυτές τις περιπτώσεις, το μόνο που πρέπει να περάσουμε είναι ο δείκτης

Πέρασμα δομών με χρήση δεικτών (2/2)




```
void multFr(FRACTION *pFrl,
            FRACTION *pFr2,
            FRACTION *pRes)
{
   ...
   pRes->numerator   = pFrl->numerator   * pFr2->numerator;
   pRes->denominator = pFrl->denominator * pFr2->denominator;
   return;
} // τέλος συνάρτησης multFr
```

ΣΧΗΜΑ 11-19 Πέρασμα δομών με χρήση δεικτών.

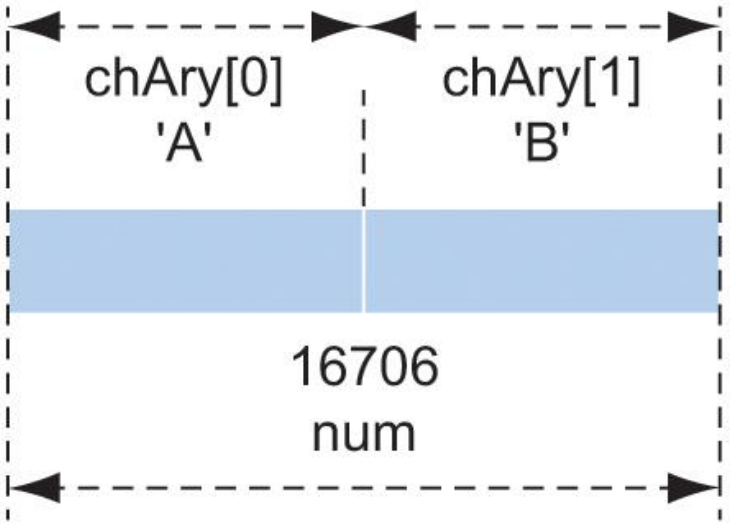
11.4
Ενώσεις

11.4 Ενώσεις

- Στη C, η ένωση (union) είναι ένας καθοριζόμενος από τον χρήστη τύπος, ο οποίος μπορεί να περιέχει στοιχεία (μέλη) διαφορετικών τύπων δεδομένων, όμοια με μία δομή
  - Ανόμοια με μία δομή όλα τα μέλη μιας ένωσης «μοιράζονται» την ίδια θέση μνήμης.
  - Αυτό σημαίνει ότι η ένωση μας δίνει τη δυνατότητα να χρησιμοποιούμε την ίδια θέση μνήμης για πολλαπλούς σκοπούς

11.4 Ενώσεις

```
union shareData
{
   char chAry[2];
   short num;
}shareData;
```



Τα `num` και `chAry` ξεκινούν στην ίδια διεύθυνση μνήμης.
Το στοιχείο `chAry[0]` καταλαμβάνει την ίδια μνήμη με το πιο σημαντικό byte του `num`.

ΣΧΗΜΑ 11-20
Δήλωση ένωσης και αναπαράστασή της στη μνήμη.

Αναφορές σε ενώσεις και μεμονωμένα πεδία τους

- Οι κανόνες για την αναφορά σε μία ένωση είναι ίδιοι με αυτούς που ισχύουν για τις δομές
  - Για να αναφερθούμε σε μεμονωμένα πεδία της ένωσης χρησιμοποιούμε τον τελεστή άμεσης επιλογής (.)
  - Κάθε αναφορά πρέπει να προσδιορίζεται πλήρως—δηλαδή πρέπει να περιλαμβάνει το όνομα της ένωσης και το όνομα του στοιχείου που μας ενδιαφέρει

```
shareData.num
shareData.chAry[0]
```

Αρχικοποίηση ενώσεων

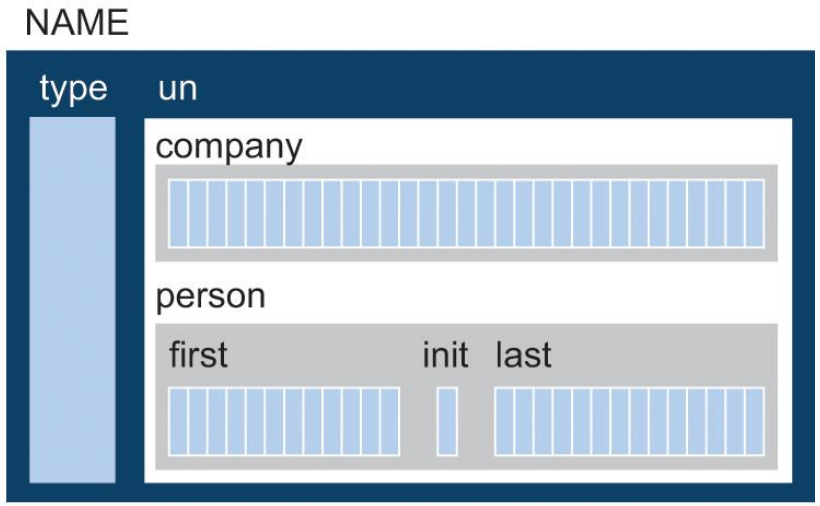
- Αν και η C επιτρέπει την αρχικοποίηση μιας ένωσης, μόνο το πρώτο στοιχείο της ένωσης μπορεί να αρχικοποιηθεί όταν ορίζεται μία μεταβλητή του τύπου της ένωσης
- Τα υπόλοιπα στοιχεία μιας ένωσης μπορούν να αρχικοποιηθούν μόνο με ανάθεση ή ανάγνωση τιμών
- Η τιμή για την αρχικοποίηση μιας ένωσης πρέπει να περικλείεται σε άγκιστρα

Ενώσεις και δομές

- Όλα τα μέλη μιας ένωσης «μοιράζονται» την ίδια θέση στη μνήμη
  - Λόγω αυτού, μόνο ένα μέλος της ένωσης μπορεί να αποθηκεύει δεδομένα ανά πάσα στιγμή

```
typedef struct
{
    char first[20];
    char init;
    char last[30];
} PERSON;

typedef struct
{
    char type;
    union
    {
        char company[40];
        PERSON person;
    } un;
} NAME;
```



ΣΧΗΜΑ 11-21 Ορισμός ένωσης για αποθήκευση ονομάτων εταιρειών ή ατόμων.

Χειρισμός διευθύνσεων Internet

- Οι διευθύνσεις που χρησιμοποιούνται από το πρωτόκολλο Διαδικτύου (Internet Protocol, IP) ορίζονται ως μία ομάδα τεσσάρων αριθμών (με ένα έως τρία ψηφία), οι οποίοι χωρίζονται με τελείες, ως εξής

```
153.18.8.105
```

- Αυτή η μορφή διευθύνσεων χρησιμοποιείται κυρίως από τους διαχειριστές συστημάτων
- Για λόγους που σχετίζονται με την εξοικονόμηση χρόνου μετάδοσης και τη διευκόλυνση της επεξεργασίας, οι διευθύνσεις IP πρέπει να μετατρέπονται σε ακέραιους αριθμούς των 4 bytes για μετάδοση στο Internet
  - Κάθε τμήμα της διεύθυνσης IP έχει μέγιστη τιμή 255
  - Αυτό σημαίνει ότι μπορεί να αποθηκευτεί σε 1 byte

Σύνοψη

- Ένας τύπος απαρίθμησης βασίζεται στον εγγενή τύπο δεδομένων int
- Σε έναν τύπο απαρίθμησης, κάθε αναγνωριστικό λαμβάνει μία ακέραια τιμή
- Μία δομή είναι μία συλλογή συναφών στοιχείων, ενδεχομένως διαφορετικών τύπων δεδομένων, τα οποία μπορούν να προσπελάζονται με αναφορά στο όνομα της δομής
- Κάθε στοιχείο μιας δομής ονομάζεται πεδίο
- Μία διαφορά μεταξύ των πινάκων και των δομών είναι ότι, ενώ όλα τα στοιχεία ενός πίνακα πρέπει να είναι ίδιου τύπου δεδομένων, τα στοιχεία μιας δομής μπορούν να είναι του ίδιου ή διαφορετικού τύπου
- Εξετάσαμε δύο διαφορετικούς τρόπους δήλωσης ή/και ορισμού μιας δομής: χαρακτηρισμένη δομή και δομή οριζόμενου τύπου

Σύνοψη

- Μία δομή μπορεί να αρχικοποιηθεί όταν ορίζεται. Οι κανόνες για την αρχικοποίηση δομών είναι αυτοί που ισχύουν για την αρχικοποίηση πινάκων

- Μπορούμε να προσπελάζουμε τα μέλη μιας δομής χρησιμοποιώντας τον τελεστή άμεσης επιλογής (.)

- Οι δομές μπορούν να προσπελάζονται μέσω ενός δείκτη. Ο καλύτερος τρόπος γι' αυτό είναι χρησιμοποιώντας τον τελεστή έμμεσης επιλογής (->)

- Οι ακόλουθες δύο εκφράσεις είναι ταυτόσημες, εάν ο ptr είναι δείκτης σε δομή:

    (* p). x p -> x

Σύνοψη

- Οι πληροφορίες που περιέχονται σε μία δομή μπορούν να περαστούν σε μία συνάρτηση χρησιμοποιώντας μία από τις ακόλουθες μεθόδους:

1. Πέρασμα μεμονωμένων μελών

2. Πέρασμα ολόκληρης της δομής

3. Πέρασμα δείκτη στη δομή

- Η ένωση είναι μία δομή η οποία επιτρέπει σε ένα τμήμα της μνήμης να χρησιμοποιείται από διαφορετικούς τύπους δεδομένων

- Στο πλαίσιο της μηχανικής λογισμικού, η σύζευξη αποτελεί ένα μέτρο του πόσο στενά σχετίζονται δύο συναρτήσεις μεταξύ τους

Σύνοψη

- Στο πλαίσιο της επιστήμης των υπολογιστών έχουν οριστεί πέντε τύποι σύζευξης: σύζευξη δεδομένων, σύζευξη αντιγράφου, σύζευξη ελέγχου, σύζευξη μέσω κοινών δεδομένων και σύζευξη περιεχομένου

- Σε ένα καλά δομημένο πρόγραμμα, οι συναρτήσεις είναι χαλαρά συζευγμένες

- Να αποφεύγετε τη σύζευξη μέσω κοινών δεδομένων και μην χρησιμοποιείτε ποτέ τη σύζευξη περιεχομένου

- Η ποιότητα της σχεδίασης ενός προγράμματος μπορεί να μετρηθεί με βάση τρεις αρχές: (1) οι μονάδες κώδικα πρέπει να είναι ανεξάρτητες· (2) οι μονάδες κώδικα πρέπει να είναι χαλαρά συζευγμένες· (3) κάθε μονάδα κώδικα πρέπει να εκτελεί μόνο μία εργασία

Το παρόν συνοδευτικό έργο **παρέχεται** αποκλειστικά και μόνο στους διδάσκοντες που έχουν επιλέξει το αντίστοιχο βιβλίο μέσω του συστήματος του **Ευδόξου** ως διδακτικό σύγγραμμα για τη διδασκαλία των μαθημάτων τους και την αξιολόγηση της εκμάθησης των φοιτητών και για όσο χρονικό διάστημα διατηρείται η επιλογή του συγκεκριμένου συγγράμματος. Η **διάδοση, δημοσίευση, αναπαραγωγή** ή **πώληση** οπουδήποτε μέρους αυτού του έργου με οποιοδήποτε μέσο καταστρέφει την ακεραιότητα του έργου **και για σκοπούς διαφορετικούς από αυτούς για τους οποίους έχει χορηγηθεί η σχετική άδεια δεν επιτρέπεται**. Το περιεχόμενο του έργου **δεν θα πρέπει να διατίθεται** στους φοιτητές παρά μόνο όταν αυτό αναφέρεται ρητά ότι μπορεί να γίνει. Η **χρήση** του παρόντος έργου γίνεται αποκλειστικά από τον διδάσκοντα στα πλαίσια των εκπαιδευτικών καθηκόντων του και με τη χρήση των μέσων που αυτός διαχειρίζεται και έχει στη διάθεσή του από τις **επίσημες υπηρεσίες του εκπαιδευτικού ιδρύματος** όπου ανήκει, διασφαλίζοντας τη μη περαιτέρω διάδοση, δημοσίευση και αναπαραγωγή του έργου προς τρίτους. Ο διδάσκων δύναται να **προσαρμόζει** μέρος του υλικού των διαφανειών σύμφωνα με τις διδακτικές του ανάγκες, αναφερόμενος όμως πάντοτε στην αρχική πηγή αναφοράς. Το παρόν έργο προστατεύεται από την ελληνική και ευρωπαϊκή νομοθεσία περί πνευματικής ιδιοκτησίας καθώς και από τη νομοθεσία του κράτους όπου ανήκει η ξενόγλωσση πρωτότυπη έκδοση του έργου.