Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κεντρικής Μακεδονίας - Σέρρες
Τμήμα Μηχανικών Πληροφορικής

Προγραμματισμός ΙΙ (Θ)

Δρ. Δημήτρης Βαρσάμης
Επίκουρος Καθηγητής

Μάρτιος 2017

ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΙΙ (Θ)

Δομές - Εισαγωγή

Το κλειδί για την αποτελεσματική χρήση της γλώσσας προγραμματισμού C είναι να προσδιορίσουμε και να χρησιμοποιήσουμε, με σωστό τρόπο, τα βοηθητικά εργαλεία που μας προσφέρει.

Ένα από αυτά τα εργαλεία είναι οι δομές (structures).

Η πιο σημαντική λειτουργία των δομών, είναι το ότι μας επιτρέπουν να ορίσουμε και να χρησιμοποιήσουμε δικούς μας τύπους δεδομένων.

Αυτή η λειτουργία διευκολύνει τον προγραμματιστή να διαχειριστεί πιο δομημένα και πιο εύκολα μεγάλο όγκο δεδομένων.

Δομές - Εισαγωγή

Μια **δομή** (**`struct`**) είναι μια συλλογή στοιχείων.

Τα στοιχεία που επιλέγουμε να αποτελέσουν μία δομή είναι ανεξαρτήτου τύπου δεδομένων και αναφερόμαστε σε αυτά ως **μέλη** της δομής.

Αυτός ο δομημένος τρόπος σύζευξης στοιχείων εκτός του ότι μας παρέχει την δυνατότητα να διαχειριστούμε μεγάλο όγκο δεδομένων, μας παρέχει επίσης την δυνατότητα να εφαρμόσουμε μία από τις αρχές του προγραμματισμού.

Κάνοντας σωστή χρήση των δομών δεν επιβαρύνουμε το πρόγραμμα μας με επαναλαμβανόμενο κώδικα.

Δομές - Εισαγωγή

Συνεπώς, μια **δομή** (**struct**) είναι ένας σύνθετος τύπος δεδομένων ή μια σύνθετη μεταβλητή στην C.

Για να χρησιμοποιήσουμε μια **δομή** (**struct**) στην C πρέπει

- Να δηλώσουμε-ορίσουμε το πρότυπο της δομής που θέλουμε να χρησιμοποιήσουμε στην αρχή του προγράμματος, δηλαδή, πριν την `main()`.
- Στην `main()` ή σε οποιαδήποτε συνάρτηση του προγράμματος, να δηλώσουμε μια μεταβλητή να είναι τύπου της δομής που δημιουργήσαμε.
- Στην `main()` ή σε οποιαδήποτε συνάρτηση του προγράμματος, να αναθέσουμε τιμές στην σύνθετη μεταβλητή και να την χρησιμοποιήσουμε σε πράξεις.

Πρότυπο Δομής

Ας δούμε όμως αναλυτικά πως μπορούμε να κατασκευάσουμε μια δομή και συγκεκριμένα ένα πρότυπο δομής.
Τα στοιχεία από τα οποία υλοποιείται το πρότυπο μιας δομής είναι:

- το **όνομα** της, με το οποίο μπορούμε να αναφερόμαστε σε αυτήν όπως αναφερόμαστε σε μία απλή μεταβλητή
- το **σώμα** της, στο οποίο βρίσκονται τα μέλη που την αποτελούν.

Για παράδειγμα:

```
struct name_of_struct
{
    type member1;
    type member2;

};
```

Πρότυπο Δομής I

Τι σημαίνουν όμως οι παραπάνω γραμμές κώδικα?

- Το **`struct`** είναι μια από τις δεσμευμένες λέξεις της γλώσσας C και γνωστοποιεί στο πρόγραμμα μας ότι επρόκειτο να ακολουθήσει η δήλωση ενός προτύπου δομής ή η δήλωση μιας μεταβλητής ως δομή.
- Το `name_of_struct` είναι το όνομα της δομής το οποίο το επιλέγει ο προγραμματιστής.
  Προσοχή στην ονομασία του ονόματος της δομής, ακολουθεί την ονοματολογία των μεταβλητών.
- Τα **μέλη** της δομής (`member1`, `member2`) τα δηλώνουμε όπως τις απλές μεταβλητές. Δηλαδή, δηλώνουμε τον τύπο και το όνομα του κάθε μέλους.

Πρότυπο Δομής II

- **Προσέχουμε** πάντα να μην ξεχάσουμε το σημείο στίξης (`;`) - ερωτηματικό στο τέλος της δήλωσης της δομής.
  Εάν το παραλείψουμε ή το ξεχάσουμε το πρόγραμμα μας θα έχει **σφάλματα μεταγλώττισης** τα οποία ενώ είναι εύκολα να διορθωθούν, σε ορισμένες περιπτώσεις μπορεί να είναι δύσκολο να εντοπιστούν.

Δημιουργία και Χρήση Δομής

Παραπάνω είδαμε πώς μπορούμε να κατασκευάσουμε και ταυτόχρονα να δηλώσουμε ένα πρότυπο δομής

Η δήλωση ενός προτύπου μιας δομής δεν δεσμεύει μνήμη στο σύστημα.

Αυτό συμβαίνει διότι κάθε φορά που κατασκευάζουμε ένα πρότυπο μιας δομής δημιουργούμε έναν νέο τύπο δεδομένων ο οποίος αρχικά δεν περιέχει τίποτα. Ο καινούργιος πλέον τύπος δεδομένων, ισχύει μόνο για το συγκεκριμένο πρόγραμμα και για να χρησιμοποιήσουμε μια δομή την οποία κατασκευάσαμε θα πρέπει πρώτα να δηλώσουμε μεταβλητές αυτού του τύπου.

Δημιουργία και Χρήση Δομής

Ας κατασκευάσουμε λοιπόν μία καινούργια δομή με πραγματικά δεδομένα, η οποία θα δημιουργηθεί για να αποθηκεύονται στοιχεία από βιβλία:

```
struct Books
{
    char title[50];
    char author[50];
    int isbn;
};
```

Παρατηρούμε λοιπόν ότι έχοντας κάνει χρήση των κατάλληλων ονομάτων καταλαβαίνουμε απευθείας ποιος ήταν ο σκοπός δημιουργίας αυτού του προτύπου δομής. Καθώς επίσης και τι στοιχεία πρόκειται να αποθηκευτούν στις μεταβλητές-μέλη της. Οι μεταβλητές που βρίσκονται σε αυτό το στάδιο, δεν περιέχουν καμία τιμή.

Δομές - Ανάθεση τιμών I

Κατασκευάσαμε και δηλώσαμε το πρότυπο της δομής μας.
Ας παρουσιάσουμε κάποιους τρόπους με τους οποίους μπορούμε να δηλώσουμε μεταβλητές και να τους αποδώσουμε τιμές:

Κατά την δήλωση της δομής:

```
struct Books
{
    char title[50];
    char author[50];
    int isbn;
}Programmatismos={"C PROGRAMMATISMOS","DEITEL",
9789605125905};
```


Δομές - Ανάθεση τιμών II

Μέσα στην `main()`:
Για να κάνουμε απόδοση τιμών μέσα στην `main()` θα πρέπει πρώτα να δηλώσουμε ότι οι μεταβλητές που θα χρησιμοποιήσουμε είναι τύπου **`struct`** και το ανάλογο όνομα της δομής.
Με δήλωση και ανάθεση ξεχωριστά

```
struct Books Programmatismos;
Programmatismos = {"C PROGRAMMATISMOS", "DEITEL",
    9789605125905};
```

ή με δήλωση και ανάθεση ταυτόχρονα

```
struct Books Programmatismos = {"C PROGRAMMATISMOS",
"DEITEL", 9789605125905};
```

Δομές - Ανάθεση τιμών III

Κάνοντας χρήση του typedef:
Η λέξη **typedef** είναι μία από τις δεσμευμένες λέξεις κλειδιά του προγραμματισμού και την χρησιμοποιούμε για να ορίσουμε **συνώνυμα** σε έναν τύπο δεδομένων. Αυτός ο τρόπος δήλωσης και απόδοσης τιμών λειτουργεί ακριβώς όπως και ο δεύτερος. Η μόνη διαφορά σημειώνεται στο ότι δηλώνουμε μια λέξη κλειδί για την δομή, κάνοντας χρήση της δεσμευμένης λέξης **typedef**. Μετά από αυτό δεν χρειάζεται να ξανακάνουμε χρήση ούτε της λέξης **struct**, καθώς ούτε και του ανάλογου ονόματος για να αναφερθούμε την δομή μας. Για παράδειγμα:

```
typedef struct Books book;

book Programmatismos = {"C Programmatismos",
"DEITEL", 9789605125905};
```


Δομές - Ανάθεση τιμών

Συμπεραίνουμε λοιπόν ότι ο δεύτερος και ο τρίτος τρόπος μας παρέχουν την δυνατότητα για ορθή χρήση της δομής. Με αυτούς τους δύο τρόπους δεν ταυτίζουμε την δομή με συγκεκριμένα δεδομένα με αποτέλεσμα να είναι πιο ευέλικτες. Δηλώνουμε μία φορά την δομή και μετά μπορούμε να την χρησιμοποιήσουμε όποτε θέλουμε και όσες φορές θέλουμε κάνοντας αναφορά μόνο στο όνομα της.

Δομές - Αναφορά και επεξεργασία I

Ακόμη μία σημαντική δυνατότητα που μας παρέχουν οι δομές είναι ότι η αναφορά και η επεξεργασία κάθε στοιχείου γίνεται ξεχωριστά και ανεξάρτητα από τα υπόλοιπα μέλη της.

Αυτό μπορούμε να το επιτύχουμε κάνοντας χρήση της **τελείας** (.), με τον ακόλουθο τρόπο:

όνομα_μεταβλητής.όνομα_στοιχείου

Δηλαδή εάν στο παραπάνω παράδειγμά μας θα θέλαμε να αλλάξουμε την τιμή κάποιου στοιχείου από το βιβλίο `Programmatismos`, αυτό θα γινόταν ως εξής:

```
Programmatismos.isbn = 978960;
```

και με αυτόν τον τρόπο πλέον το στοιχείο `isbn` του συγκεκριμένου βιβλίου έχει την τιμή `978960`.

Δομές - Αναφορά και επεξεργασία II

Ακολουθώντας αυτόν τον τρόπο μπορούμε να μεταβάλλουμε και να τυπώσουμε στην οθόνη κάθε στοιχείο του βιβλίου ξεχωριστά.

Προσέχουμε όμως, η αναφορά σε κάποιο στοιχείο γίνεται μόνο με την χρήση της **τελείας** (.).

Παράδειγμα 1: Απλή Δομή

Να δημιουργηθεί πρόγραμμα το οποίο θα κατασκευάζει μία δομή με όνομα `Person` η οποία θα αποτελείται από τα παρακάτω στοιχεία:

Όνομα,

Ύψος,

Βάρος,

Δείκτης Μάζας Σώματος

Θα πρέπει να καταχωρηθούν τιμές για 2 άτομα και έπειτα να εκτυπωθούν τα στοιχεία του ατόμου με τον μεγαλύτερο δείκτη μάζας σώματος.

Ο Δείκτης Μάζας Σώματος υπολογίζεται από τον τύπο

$$\Delta M\Sigma = \frac{B}{Y^2}$$

Παράδειγμα 1: Απλή Δομή I

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

struct Person{
    char name[15];
    float height;
    int weight;
    float DMS;
};

int main(){
    struct Person pers1, pers2;
    printf("Give Person 1 data:\n");
    printf("Name: ");
    scanf("%s", pers1.name);
    printf("Height: ");
    scanf("%f", &pers1.height);
```

Παράδειγμα 1: Απλή Δομή II

```
    printf("Weight: ");
    scanf("%d", &pers1.weight);
    pers1.DMS=pers1.weight/(pers1.height*pers1.height);
    printf("Give Person 2 data:\n");
    printf("Name: ");
    scanf("%s", pers2.name);
    printf("Height: ");
    scanf("%f", &pers2.height);
    printf("Weight: ");
    scanf("%d", &pers2.weight);
    pers2.DMS=pers2.weight/(pers2.height*pers2.height);
    if(pers1.DMS>=pers2.DMS){
        printf("Name: %s\n", pers1.name);
        printf("Heigth: %.2f\n", pers1.height);
        printf("Weigth: %d\n", pers1.weight);
    }
    else{
```


Παράδειγμα 1: Απλή Δομή III

```
        printf("Name: %s\n", pers2.name);
        printf("Heigth: %.2f\n", pers2.height);
        printf("Weigth: %d\n", pers2.weight);
    }
    return 0;
}
```

Παράδειγμα 2: Δομή Πίνακας

Να δημιουργηθεί πρόγραμμα το οποίο θα κατασκευάζει μία δομή με όνομα Student η οποία θα αποτελείται από τα παρακάτω στοιχεία:

Όνομα,

Επώνυμο,

Α.Ε.Μ,

Μέσο όρο μαθημάτων

Θα πρέπει να καταχωρηθούν τιμές για 5 φοιτητές και έπειτα να εκτυπωθούν τα στοιχεία του φοιτητή με τον μεγαλύτερο μέσο όρο μαθημάτων (θεωρούμε ότι η τιμή είναι μοναδική).

Παράδειγμα 2: Δομή Πίνακας I

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

struct Student{
    char name[15];
    char surname[15];
    int aem;
    float average;
};

int main(){
    struct Student stud[5];
    int i, position;
    float maxAvg;
    for(i=0;i<5;i=i+1){
        printf("Give students %d data:\n", i+1);
        printf("First name: ");
```

Παράδειγμα 2: Δομή Πίνακας II

```
        scanf("%s", stud[i].name);
        printf("Last name: ");
        scanf("%s", stud[i].surname);
        printf("Students personal number: ");
        scanf("%d", &stud[i].aem);
        printf("Average Grade: ");
        scanf("%f", &stud[i].average);
    }
    maxAvg=stud[0].average;
    for(i=1;i<5;i=i+1){
        if(stud[i].average > maxAvg){
            maxAvg = stud[i].average;
            position=i;
        }
    }
    printf("Student with the maximum average is:\n");
```


Παράδειγμα 2: Δομή Πίνακας III

```
    printf("%s %s\n", stud[position].name, stud[position
        ].surname);
    printf("with personal number: %d\n", stud[position].
        aem);
    printf("and average: %.2f", stud[position].average);
    return 0;
}
```

Άσκηση

Να δημιουργηθεί πρόγραμμα το οποίο θα δέχεται τα εξής στοιχεία από τους πελάτες μιας εταιρίας αυτοκινήτων:

Όνομα,

Επίθετο,

Τηλέφωνο,

Διεύθυνση,

Υπόλοιπο οφειλών

Θα τα αποθηκεύει σε έναν πίνακα και έπειτα (α) εάν το **Υπόλοιπο οφειλών** είναι μεγαλύτερο από €500 θα εμφανίζει τα στοιχεία του πελάτη και (β) θα εμφανίζει το συνολικό υπόλοιπο οφειλών.