

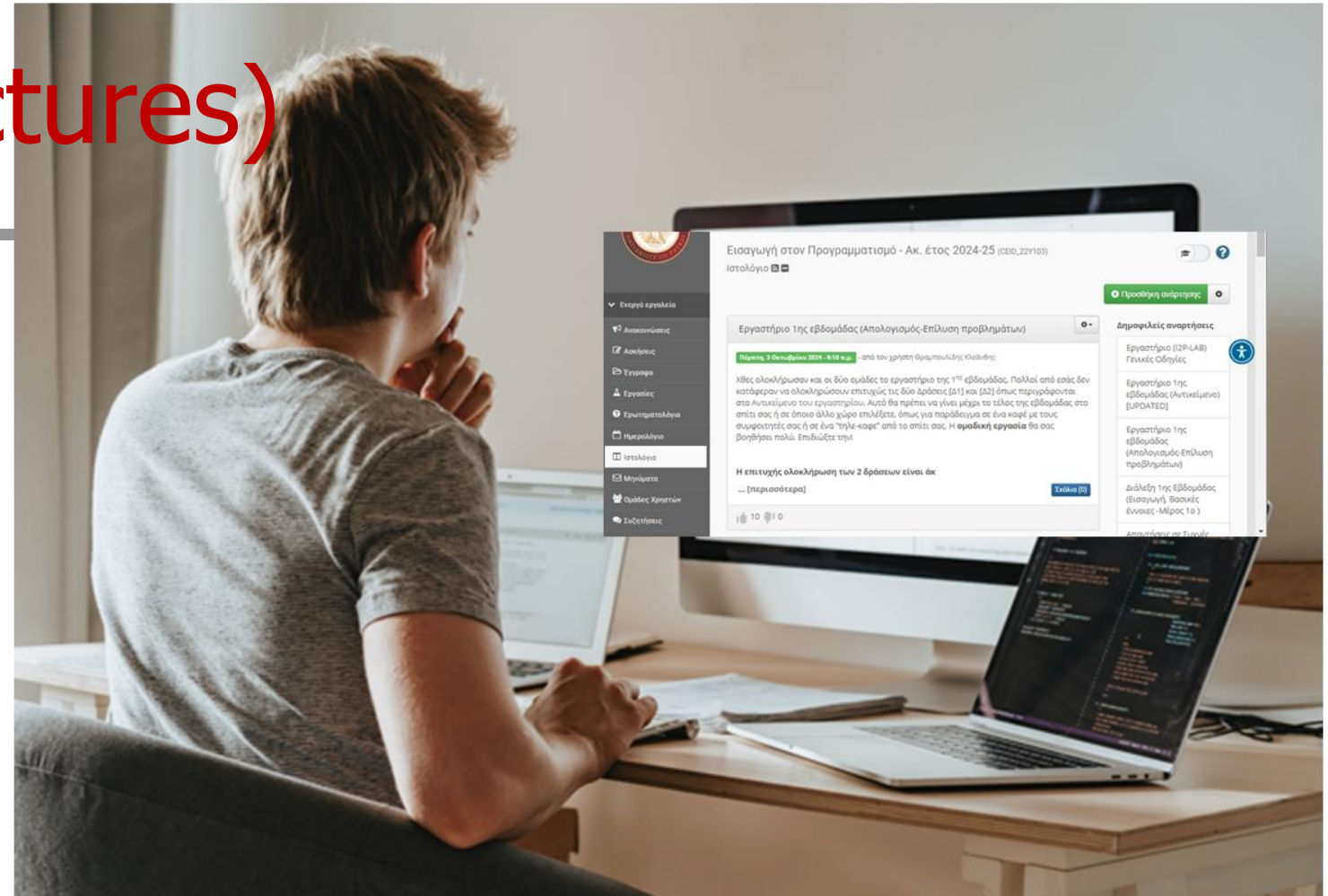
Εισαγωγή στον Προγραμματισμό

(CEID_NY131)

Δομές (Structures)

```
struct student {  
    char fname[20];  
    char lname[30];  
    int am;  
};
```

Kleanthis Thramboulidis
Prof. of Software and System Engineering
University of Patras
<https://sites.google.com/site/thramboulidiskleanthis/>



Δομές

- Στην C μπορούμε να προσθέσουμε τους δικούς μας τύπους (π.χ. FILE)
- Βασικά εργαλεία: *struct*, *typedef*.

struct: ορίζει ένα νέο τύπο δεδομένων.

typedef: συσχετίζει ένα όνομα με αυτόν.

- Οι μεταβλητές δομών μπορούν να αποδίδονται σαν τιμές, να μεταβιβάζονται σε συναρτήσεις και να επιστρέφονται από συναρτήσεις. Επίσης, μπορεί να εφαρμόζεται σε αυτές ο τελεστής *sizeof*).
- Αρχικοποιούνται με λίστα τιμών όπως οι πίνακες.
- ΔΕΝ συγκρίνονται με χρήση τελεστή `==` ή `!=`
- Η αναφορά σε ένα μέλος γίνεται: *όνομα_μεταβλητής_δομής.μέλος*
- Ειδικός συμβολισμός για προσπέλαση μέλους μεταβλητής δείκτη σε δομή.

Αφαιρετικότητα στα Δεδομένα

Δώσε μου τη
διεύθυνση σου



Χρησιμοποιούμε τον όρο **Διεύθυνση** για να αναφερθούμε σε ένα σύνολο από στοιχεία (όνομα, επώνυμο, οδός, αριθμός, ταχυδρομικό κώδικα, πόλη). Αποφεύγουμε με τον τρόπο αυτό τη λεπτομερή αναφορά σε καθένα από αυτά.

«An **address book** (...) is a book, (...) used for storing entries, called **contacts**. Each **contact entry** usually consists of a few standard fields (for example: first name, last name, company name, address, telephone number, email address, fax number, mobile phone number).» wikipedia

Άσκηση [gov.gr wallet]

- Αναπτύξτε ένα πρόγραμμα για τη διαχείριση N αριθμού αστυνομικών ταυτοτήτων
 - Ονοματεπώνυμο
 - Αριθμός ταυτότητας
 - Ημερομηνία έκδοσης
 - Αρχή Έκδοσης
 - Ύψος
 - Χρώμα ματιών
 - Εθνικότητα
 -

Πως θα οργανώσετε τα δεδομένα N ταυτοτήτων;

Η κατασκευή της Δομής (struct) μας δίνει τη λύση.

Στοιχεία Ταυτότητας

Αριθμός Ταυτότητας:

Ημερομηνία Έκδοσης:

Αρχή Έκδοσης:

Αριθμός Προηγούμενης Ταυτότητας:

Όνομα:

Όνομα (Λατινικά):

Επώνυμο:

Επώνυμο (Λατινικά):

Όνομα Πατρός:

Όνομα Πατρός (Λατινικά):

Όνομα Μητέρας:

Όνομα Μητέρας (Λατινικά):

Ημερομηνία Γέννησης:

Τόπος Γέννησης:

Φωτογραφία Ταυτότητας:

gov.gr wallet

Δομή - Struct

- **επιτρέπει** την ομαδοποίηση μεταβλητών (πιθανόν διαφορετικού τύπου) που αναπαριστούν ένα σύνολο χαρακτηριστικών μιας οντότητας.

- **έχει**

- ένα **όνομα** και
- ένα σύνολο από **μέλη**

Κάθε μέλος προσδιορίζεται από το όνομα και τον τύπο του.

```
struct student {  
    char fname[20];  
    char lname[30];  
    int am;  
};
```

Οργάνωση Διάλεξης

- Δήλωση δομής
- Ένθεση δομών
- Χρήση δομών
 - Δήλωση μεταβλητών τύπου δομής
 - Απόδοση αρχικών τιμών
 - Αναφορά σε μέλη δομής
- Πίνακες δομών
- Δομή ως όρισμα και επιστρεφόμενη τιμή
 - Παραδείγματα
- Δείκτες σε δομές

Δήλωση δομής

```
struct student {  
    char fname[20];  
    char lname[30];  
    int am;  
};
```

Δηλώνει μια δομή με όνομα **student** η οποία απαρτίζεται από 3 μέλη, δύο αλφαριθμητικά (fname και lname) και ένα int (am).

```
struct rectangle {  
    int x1,y1;  
    int x2,y2;  
};
```

Δηλώνει μια δομή με όνομα **rectangle** η οποία απαρτίζεται από 4 μέλη τα οποία είναι όλα ίδιου τύπου.

Δήλωση δομής με χρήση της **typedef**

```
typedef struct student {  
    char fname[20];  
    char lname[30];  
    int am;  
} Student;
```

Δηλώνει μια δομή με όνομα **student** και επιπλέον ορίζει το όνομα **Student** για πιο εύκολη αναφορά στην δομή student.

```
typedef struct rectangle {  
    int x1,y1;  
    int x2,y2;  
} Rectangle;
```

Δηλώνει μια δομή με όνομα **rectangle** και επιπλέον ορίζει το όνομα **Rectangle** για πιο εύκολη αναφορά στην δομή rectangle.

typedef είναι δεσμευμένη λέξη στην C και χρησιμοποιείται για **να ορίσει ένα νέο όνομα** σε ένα τύπο δεδομένων. **Δεν δημιουργεί τύπο.**

Δήλωση δομής με βάση άλλη δομή

```
struct point {
```

```
    int x;
```

```
    int y;
```

```
};
```

```
struct rectangle2 {
```

```
    struct point p1;
```

```
    struct point p2;
```

```
};
```

Δηλώνει τη δομή **point** η οποία απαρτίζεται από δύο μέλη, τα x και y τα οποία είναι τύπου int.

Δηλώνει τη δομή **rectangle** η οποία απαρτίζεται από δύο μέλη, τα p1 και p2 τα οποία είναι τύπου **struct point**.

Δήλωση μεταβλητής τύπου δομής

- Ακολουθεί τον γενικό κανόνα. Ορίζει τον τύπο και στη συνέχεια το όνομα της μεταβλητής.

`struct rectangle rect1;` ή

`Rectangle rect1;`

Η πρόταση δήλωσης

- **δηλώνει** την `rect1` ως μεταβλητή τύπου `struct rectangle`, που έχει τη δομή που ορίζει ο τύπος `struct rectangle` ή αλλιώς `Rectangle`.
- **έχει** ως αποτέλεσμα τη δέσμευση τόσης μνήμης όσης απαιτείται για την αποθήκευση των μελών της δομής που δηλώνει ο τύπος `struct rectangle`.

Απόδοση αρχικής τιμής

```
Rectangle rect1 = { 10,10,30,20 };
```

Είναι μια πρόταση δήλωσης που αποδίδει και αρχικές τιμές.

Αποδίδει αρχικές τιμές στα μέλη της μεταβλητής **rect1** τα οποία είναι τα x1,y1, x2 και y2 όπως ακριβώς τα ορίζει η δομή **struct rectangle**.

```
struct rectangle2 rect2 = { {1,2},{3,4} };
```

Η πρόταση αποδίδει αρχικές τιμές στα μέλη της μεταβλητής **rect2** τα οποία είναι τα p1 και p2 (με επιμέρους μέλη p1.x, p1.y, και p2.x, p2.y αντίστοιχα) όπως ακριβώς τα ορίζει η δομή **struct rectangle2**.

Αναφορά στα μέλη της δομής

- Για αναφορά σε μέλος της δομής χρησιμοποιείται ο τελεστής τελεία (.) ανάμεσα στο όνομα της μεταβλητής και το όνομα του μέλους, όπως το ορίζει ο αντίστοιχος τύπος της μεταβλητής.

`rect1.x1`

Αναφέρεται στο μέλος `x1` της δομής `struct rectangle` καθώς η `rect1` είναι τύπου `struct rectangle`.

`rect2.p1.x`

Αναφέρεται στο μέλος `x` της `p1` (τύπου `point`) η οποία `p1` είναι μέλος της μεταβλητής `rect2`, η οποία είναι τύπου `struct rectangle2`.

Σημείωση: Η `rect2` έχει δηλωθεί προηγουμένως ως τύπου `struct rectangle2`.

Αντιγραφή δομής

- Ο τελεστής `=` μπορεί να χρησιμοποιηθεί για να αναθέσει σε μία μεταβλητή δομής την τιμή μιας άλλης μεταβλητής του ίδιου τύπου.

```
struct rectangle2 rect21 = { {1,2},{3,4} };
```

```
struct rectangle2 rect22;
```

```
rect22 = rect21;
```

Τα μέλη της μεταβλητής `rect22` θα πάρουν τις τιμές των αντίστοιχων μελών της `rect21`.

Δομή ως όρισμα και επιστρεφόμενη τιμή συνάρτησης

- Μια δομή **μπορεί να προσδιορίζει τον τύπο**
 - Ορίσματος συνάρτησης
 - Επιστρεφόμενης τιμής συνάρτησης

```
11 void printRectangle(struct rectangle2 rect);  
12 struct rectangle2 doubleRect(struct rectangle2);
```

Η συνάρτηση **printRectangle** δέχεται ένα όρισμα τύπου **struct rectangle2**.

Η συνάρτηση **doubleRect** δέχεται ένα όρισμα τύπου **struct rectangle2** και επιστρέφει τιμή τύπου **struct rectangle2**.

Πίνακες δομών

Δήλωση

```
struct rectangle rect[10];
```

Δηλώνεται η **rect** ως ένας πίνακας 10 στοιχείων όπου το κάθε στοιχείο είναι τύπου **struct rectangle**.

Αρχικοποίηση

```
17 struct rectangle2 rectArr[10] = {  
18     { {1,2},{3,4} },  
19     { {5,6},{7,8} }  
20 };
```

Δηλώνεται η **rectArr** ως ένας πίνακας 10 στοιχείων τύπου **struct rectangle2** και αποδίδονται αρχικές τιμές στα δύο πρώτα στοιχεία του.

Αναφορά σε μέλη

rect[0].x1

rectArr[0].p1.x

Αναφέρεται στο μέλος **x1** (όπως το ορίζει η δομή τύπου **struct rectangle**) του πρώτου στοιχείου του πίνακα **rect**.

Αναφέρεται στο μέλος **x** (όπως το ορίζει η δομή τύπου **struct point**) του μέλους **p1** (όπως το ορίζει η δομή τύπου **struct rectangle2**) του πρώτου στοιχείου του πίνακα **rectArr**.

Παράδειγμα 1^ο Δομή ως Όρισμα 1/2

TestStruct1.c

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 struct point {
5     int x;
6     int y;
7 };
8 struct rectangle2 {
9     struct point p1;
10    struct point p2;
11 };
12
13 void printRectangle(struct rectangle2 rect);
14 /* run this program using the console pauser or add y
15
16 struct rectangle2 rect2 = { {1,2},{3,4} };
17 struct rectangle2 rectArr[2] = {
18     { {1,2},{3,4} },
19     { {5,6},{7,8} }
20 };
```

Δήλωση δομών

Δήλωση συνάρτησης
(Function prototype)

Η συνάρτηση δέχεται το όρισμα rect το οποίο είναι τύπου δομής struct rectangle2

Απόδοση αρχικών τιμών σε

α) μεταβλητή τύπου δομής, και,
β) σε πίνακα 2 στοιχείων τύπου δομής rectangle2.

Παράδειγμα 1^ο Δομή ως Όρισμα 2/2

```
TestStruct1.c
22 int main(int argc, char *argv[]) {
23     int i=0;
24     printRectangle(rect2);
25     // printf("%d\t%d\n",rect2.p1.x, rect2.p1.y);
26     // printf("%d\t%d\n",rect2.p2.x, rect2.p2.y);
27
28     for(i=0;i<2;i++)
29     // {
30     //     printf("%d\t%d\n",rectArr[i].p1.x, rectArr[i].p1.y);
31     //     printf("%d\t%d\n",rectArr[i].p2.x, rectArr[i].p2.y);
32     // }
33     printRectangle(rectArr[i]);
34     system("pause");
35     return 0;
36 }
37
38 void printRectangle(struct rectangle2 rect){
39     printf("p1.x=%d\tp1.y=%d\n",rect.p1.x, rect.p1.y);
40     printf("p2.x=%d\tp2.y=%d\n\n",rect.p2.x, rect.p2.y);
41 }
```

Παράδειγμα 2^ο Δομή ως επιστρεφόμενη τιμή 1/3

StructRectangleTest2.c

```
1 struct point {
2     int x;
3     int y;
4 };
5 struct rectangle2 {
6     char color[10];
7     struct point p1;
8     struct point p2;
9 };
10
11 void printRectangle(struct rectangle2 rect);
12 struct rectangle2 doubleRect(struct rectangle2);
13
14 struct rectangle2 rect2 = { "Green", {1,2}, {3,4} };
15 struct rectangle2 rect3;
16 struct rectangle2 rectArr[2];
```

Δήλωση δομών

Function prototypes

Η συνάρτηση doubleRect δέχεται ένα όρισμα το οποίο είναι τύπου δομής struct rectangle2 και έχει επιστρεφόμενη τιμή του ίδιου τύπου.

Απόδοση αρχικών τιμών σε μεταβλητή τύπου δομής rectangle2.

Παράδειγμα 2^ο Δομή ως επιστρεφόμενη τιμή 2/3

```
18 int main(int argc, char *argv[]) {
19     int i;
20     printRectangle(rect2);
21     rect3=rect2;
22     strcpy(rect3.color, "Yellow");
23     printRectangle(rect3);
24
25     rectArr[0]=doubleRect(rect2);
26     rectArr[1]=doubleRect(rectArr[0]);
27     for(i=0;i<2;i++)
28         printRectangle(rectArr[i]);
29     return 0;
30 }
```

Τα μέλη της rect3 παίρνουν τις τιμές των μελών της rect2.

Αποδίδεται η τιμή Yellow στην μεταβλητή color της rect3.

Καλείται η συνάρτηση doubleRect με όρισμα την rect2 και η επιστρεφόμενη τιμή της αποδίδεται στο πρώτο στοιχείο του πίνακα rectArr.

```
32 struct rectangle2 doubleRect(struct rectangle2 r){
33     r.p2.x*=2;
34     r.p2.y*=2;
35     return r;
36 }
```

Οι μεταβλητές τύπου δομής, όπως η r, περνιούνται ως ορίσματα by value (με τιμή) έτσι η αλλαγή των τιμών των μελών τους δεν επηρεάζει τα πραγματικά ορίσματα (Actual Parameter).

Παράδειγμα 2^ο Δομή ως επιστρεφόμενη τιμή 3/3

Έξοδος Προγράμματος

C:\Code\courses\I2P22-23\StructRectangleTest2\StructRectangleTest2.exe

```
Rect color: Green  
p1.x=1 p1.y=2  
p2.x=3 p2.y=4
```

Αποδόθηκε η τιμή Yellow στην μεταβλητή color της rect3.

```
Rect color: Yellow  
p1.x=1 p1.y=2  
p2.x=3 p2.y=4
```

Τα μέλη της rect3 πήραν τις τιμές των μελών της rect2.

```
Rect color: Green  
p1.x=1 p1.y=2  
p2.x=6 p2.y=8
```

Κλήθηκε η συνάρτηση doubleRect με όρισμα την rect2 και η επιστρεφόμενη τιμή της αποδόθηκε στο πρώτο στοιχείο του πίνακα rectArr.

```
Rect color: Green  
p1.x=1 p1.y=2  
p2.x=12 p2.y=16
```

Οι τιμές των μελών του πρώτου στοιχείου δεν άλλαξαν από την χρήση του ως πραγματικού ορίσματος στην κλήση της συνάρτησης doubleRect γιατί η μεταβλητή τύπου δομής περνιέται by value (με τιμή).

Process exited after 0.04876 seconds with return value 0
Press any key to continue . . .

Ένα εργοστάσιο παρασκευάζει μικρά και μεγάλα μπουκάλια εμφιάλωσης. Τα μικρά κοστίζουν 0.008€ και τα μεγάλα 0.02€. Για παραγγελίες μεγαλύτερες από 200€ ή 3000 μπουκάλια, υπάρχει έκπτωση 8%. Για παραγγελίες μεγαλύτερες από 600€, η έκπτωση είναι 20%.

Να γραφεί ένα πρόγραμμα το οποίο να δέχεται το πλήθος των μικρών και μεγάλων μπουκαλιών μίας παραγγελίας και να εμφανίζει το συνολικό κόστος υπολογίζοντας την έκπτωση.

Δώστε στον χρήστη τη δυνατότητα επιλογής της προς εκτέλεση λειτουργίας

Δες Άσκηση 4, Κεφάλαιο 11
Για ορισμό κύριας ροής του προγράμματος

Set2Exer3V2.c

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #define SBPRICE 8
4 #define LBPRICE 20
5
6 struct order{
7     char name[20];
8     int sbnum;
9     int lbnum;
10    float cost;
11    float finalCost;
12 };
13
14 struct order orders[10]={
15     {"Nikos",200,300},
16     {"Kostas",1000,3000},
17     {"panos",2200,3100}
18 };
19
20 struct order getOrder(void);
21 void calculateCost(struct order o);
22 float calculateFinalCost(struct order o);
23 void displayOrder(struct order o);
```

Δήλωση δομής order

Απόδοση αρχικών τιμών σε 3 στοιχεία πίνακα 10 στοιχείων τύπου δομής order.

Function prototypes

Παράδειγμα 4^ο - Άσκηση Fractions

```
typedef struct fraction{
    int ar;
    int par;
}Fraction;

typedef struct expression{
    char oprtr; // for operator
    Fraction op1;
    Fraction op2;
}Expression;
```

Δήλωση των τύπων
Fraction και **Expression**

Function prototype της
συνάρτησης
getExpressionV2()

```
Expression getExpressionV2(void);
```

Παράδειγμα 5^ο - Άσκηση wordsHandling

```
136 typedef struct wordStatistic{
137     int max_len, min_len;
138     int total_len;
139     int numOfWords;
140 }WordStatistics;
```

Δήλωση του τύπου
WordStatistics

```
142 WordStatistics initializeWordStatsV2(char str[],int numOfWords);
143 void displayWordStatsV2(WordStatistics ws);
```

Η initializeWordStatsV2 έχει **επιστρεφόμενη τιμή** τύπου WordStatistics
Η displayWordStatsV2 **δέχεται όρισμα** τύπου WordStatistics

```
194 void displayWordStatsV2(WordStatistics ws){
195     printf("max len = %d\n",ws.max_len);
196     printf("min len = %d\n",ws.min_len);
197     printf("average len = %f\n",freeEntry==0?0:(float)ws.total_len/ws.numOfWords);
198 }
```


Δείκτες σε δομές

Δήλωση

```
struct rectangle *rect1Ptr;  
struct rectangle2 *rectPtr[10];
```

Η `rect1Ptr` είναι δείκτης σε μεταβλητή τύπου `rectangle`

Η `rectPtr` είναι πίνακας 10 στοιχείων που το καθένα είναι δείκτης σε δομή `rectangle2`.

Αρχικοποίηση

```
struct rectangle *rect1Ptr = &rect1;
```

Δηλώνεται η μεταβλητή `rect1Ptr` ως δείκτης σε δομή `rectangle` και της ανατίθεται ως τιμή η διεύθυνση της μεταβλητής `rect1`.
Η `rect1Ptr` δείχνει στην `rect1`.

Αναφορά σε μέλη

```
rect1Ptr->x1
```

```
rectPtr[0]->p1.x
```

Δες Άσκηση 6, Κεφάλαιο 11
Βιβλίο Διευθύνσεων