Εισαγωγή στην Πληροφορική & στον Προγραμματισμό

Αρχές Προγραμματισμού Η/Υ (με τη γλώσσα C)

Διάλεξη #6

17 Μαΐου 2024

Παναγιώτης Παύλου

c-programming-24@allos.gr

Αναπαράσταση χαρακτήρων

Αναπαράσταση μεμονωμένου χαρακτήρα

Κωδικοποίηση χαρακτήρων 1/2

Οι χαρακτήρες κειμένου στον υπολογιστή παριστάνονται ως αριθμοί βάσει κάποιας αντιστοίχισης η οποία ονομάζεται κωδικοποίηση (encoding) του κειμένου. Παραδοσιακά **κάθε χαρακτήρας αντιστοιχεί σε έναν αριθμό (και αντίστροφα)**.

Η πιο γνωστή κωδικοποίηση κειμένου χρησιμοποιεί κωδικούς αριθμούς, οι οποίοι στο δυαδικό παριστάνονται με 7bits (άρα τιμές 0 – 127). Ονομάζεται **ASCII** (δείτε και εδώ) και ουσιαστικά είναι ένας πίνακας αντιστοίχισης για τους χαρακτήρες του Αγγλικού αλφαβήτου, των αριθμητικών ψηφίων και κάποιων συμβόλων προς τους κωδικούς τους.

Επειδή το κάθε byte έχει 8bits οι υπόλοιπες 128 τιμές (128-255) που δεν ορίζονται από το ASCII μπορούν να περιέχουν διάφορους χαρακτήρες. Το ποιοι χαρακτήρες είναι αυτοί εξαρτάται από την παραλλαγή της κωδικοποίησης που θα επιλεγεί. Συνήθως υπάρχει μία παραλλαγή για κάθε γλώσσα.

Άρα για την αναπαράσταση κατά ASCII σε κάθε χαρακτήρα αρκούν 8bits = 1byte, γι'αυτό και ο τύπος δεδομένων του ακεραίου των 8bits στη C ονομάζεται **`char`**.

Κωδικοποίηση χαρακτήρων 1/2

Στις νέες πολυγλωσσικές κωδικοποιήσεις (encodings) υπάρχει δυνατότητα συνδυασμού περισσότερων bytes για την αναπαράσταση των χαρακτήρων. Έτσι μπορεί μία κωδικοποίηση να καλύψει περισσότερους από 256 διαφορετικούς χαρακτήρες, πράγμα απαραίτητο για πολυγλωσσικά κείμενα. Τέτοιες κωδικοποιήσεις είναι οι UTF-8, UTF-16, κλπ

Στο μάθημα θα περιοριστούμε σε κείμενα με Αγγλικούς χαρακτήρες και στην κωδικοποίηση ASCII με 7bits.

Η εμφάνιση στην οθόνη ενός μεμονωμένου χαρακτήρα γίνεται με τη χρήση της **%c** στην **printf**. Σε αυτή θα πρέπει να αντιστοιχίσουμε έναν 8bit (άρα τύπου δεδομένων char ή unsigned char) αριθμό που θα είναι ο κωδικός του χαρακτήρα. Π.χ.

```
printf("%c\n", 65);
```

Που όπως θα δούμε θα εμφανίσει τον χαρακτήρα **A**. Βέβαια εμείς δεν θα θυμόμαστε τους κωδικούς των χαρακτήρων, ούτε και θα ήταν σωστό να κάνουμε κάτι τέτοιο, καθώς σε σπάνιες περιπτώσεις μπορεί να είναι διαφορετικοί από το ένα σύστημα σε άλλο. Οπότε για να παραστήσουμε τον κωδικό του χαρακτήρα τον σημειώνουμε μέσα σε μονά εισαγωγικά:

```
printf("%c\n", 'A');
```

Πίνακας ASC-II (κατακόρυφα)

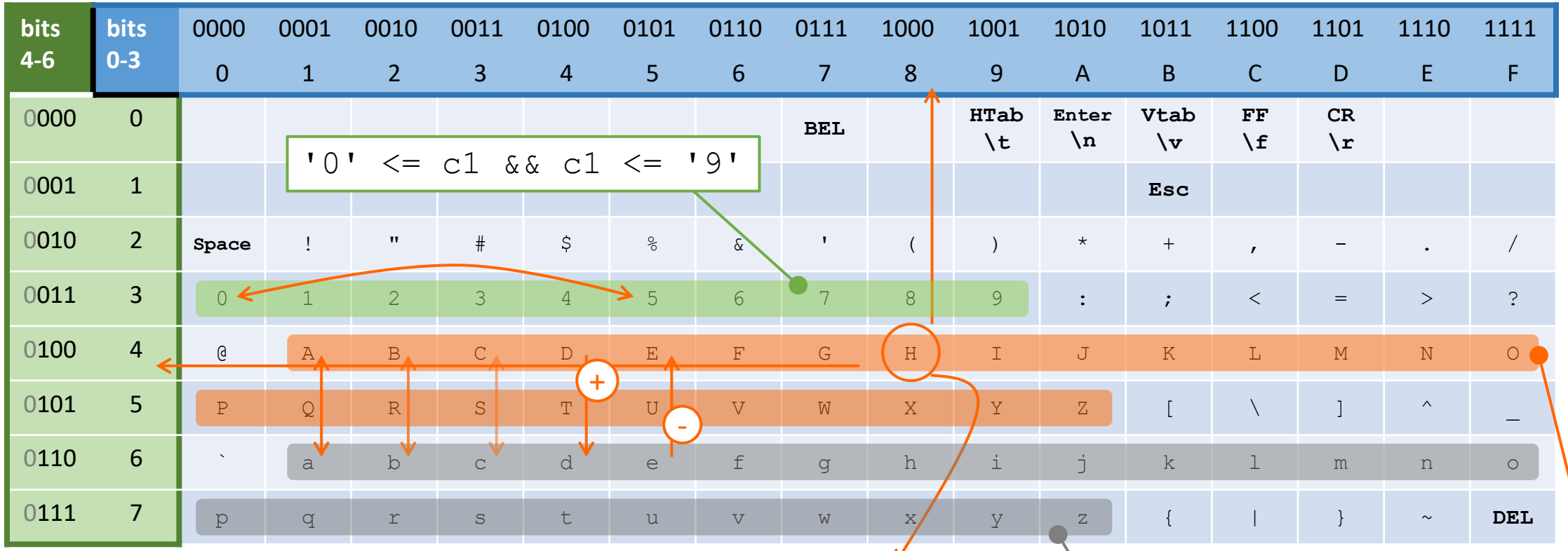
Κωδικός (10δικό)	Κωδικός (16αδικό)	Χαρακτήρας
32	20	space
33	21	!
34	22	"
35	23	#
36	24	$
37	25	%
38	26	&
39	27	'
40	28	(
41	29	)
42	2A	*
43	2B	+
44	2C	,
45	2D	-
46	2E	.
47	2F	/
48	30	0
49	31	1
50	32	2
51	33	3
52	34	4
53	35	5
54	36	6
55	37	7

Κωδικός (10δικό)	Κωδικός (16αδικό)	Χαρακτήρας
56	38	8
57	39	9
58	3A	:
59	3B	;
60	3C	<
61	3D	=
62	3E	>
63	3F	?
64	40	@
65	41	A
66	42	B
67	43	C
68	44	D
69	45	E
70	46	F
71	47	G
72	48	H
73	49	I
74	4A	J
75	4B	K
76	4C	L
77	4D	M
78	4E	N
79	4F	O

Κωδικός (10δικό)	Κωδικός (16αδικό)	Χαρακτήρας
80	50	P
81	51	Q
82	52	R
83	53	S
84	54	T
85	55	U
86	56	V
87	57	W
88	58	X
89	59	Y
90	5A	Z
91	5B	[
92	5C	\
93	5D	]
94	5E	^
95	5F	_
96	60	`
97	61	a
98	62	b
99	63	c
100	64	d
101	65	e
102	66	f
103	67	g

Κωδικός (10δικό)	Κωδικός (16αδικό)	Χαρακτήρας
104	68	h
105	69	i
106	6A	j
107	6B	k
108	6C	l
109	6D	m
110	6E	n
111	6F	o
112	70	p
113	71	q
114	72	r
115	73	s
116	74	t
117	75	u
118	76	v
119	77	w
120	78	x
121	79	y
122	7A	z
123	7B	{
124	7C	|
125	7D	}
126	7E	~

Πίνακας ASC-II




```
› char c1 = someDigitCharacter;
  int digitValue = c1 - '0';
```

```
› int offsetCaps2Small = 'a' - 'A';
  char c2 = someCapitalLetter;
  char c3 = c2 + offsetCaps2Small;
```

Ομάδες χαρακτήρων – ctype

- **Ψηφία / Digits**
  Τα αριθμητικά ψηφία 0 1 2 3 4 5 6 7 8 9

`int isdigit(int c)`

- **Δεκαεξαδικά ψηφία / Hexadecimal digits**
  Τα αριθμητικά ψηφία και οι χαρακτήρες a-f
  0 1 2 3 4 5 6 7 8 9 A a B b C c D d E e F f

`int isxdigit(int c)`

- **Πεζά γράμματα / Lowercase letters**
  Οι πεζοί χαρακτήρες του Αγγλικού αλφαβήτου
  a b c d e f g h i j k l m n o p q r s t u v w x y z

`int islower(int c)`

- **Κεφαλαία γράμματα / Uppercase letters**
  Οι κεφαλαίοι χαρακτήρες του Αγγλικού αλφαβήτου
  A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

`int isupper(int c)`

- **Γράμματα ή Αλφαβητικοί χαρακτήρες / Letters ή Alphabetic characters**
  Τα κεφαλαία και τα πεζά γράμματα μαζί

`int isalpha(int c)`

- **Αλφαριθμητικοί χαρακτήρες / Alphanumeric characters**
  Τα γράμματα μαζί με τα (αριθμητικά) ψηφία

`int isalnum(int c)`

- **Σημεία στίξης / Punctuation characters**
  Οι χαρακτήρες
  ! " # $ % & ' ( ) * + , - . / : ; < = > ? @ [\ ] ^ _ ` { | } ~

`int ispunct(int c)`

0 1 2 3 4 5 6 7 8 9

abcdefghijklmnopqrstuvwxyz

ABCDEFGHIJKLMNOPQRSTUVWXYZ

!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~

ΠΡΟΣΟΧΗ!
Χρησιμοποιήστε τις επιστρεφόμενες τιμές ως λογικές μεταβλητές.

- **Γραφικοί χαρακτήρες / Graphical characters**
  Οι αλφαριθμητικοί χαρακτήρες μαζί με τα σημεία στίξης. Δηλαδή όλοι οι χαρακτήρες οι οποίοι έχουν οπτική αναπαράσταση (εμφανίζουν κάτι στην οθόνη).

`int isgraph(int c)`

- **Κενοί χαρακτήρες / Space ή Whitespace characters**
  Χαρακτήρες που έχουν επίδραση στο αποτέλεσμα αλλά δεν έχουν ορατό περιεχόμενο. Πιο συγκεκριμένα είναι το κενό ( ), το οριζόντιο tab (\t), το κατακόρυφο tab (\v), αλλαγή γραμμής (\n), επιστροφή (\r) και form feed (\f).

`int isspace(int c)`

Αυτοί είναι και οι χαρακτήρες που χρησιμοποιούνται για την ελεύθερη σύνταξη ενός προγράμματος C

- **Εκτυπώσιμοι χαρακτήρες / Printable characters**
  Οι γραφικοί χαρακτήρες και οι κενοί μαζί.

`int isprint(int c)`

- **Χαρακτήρες ελέγχου / Control characters**
  Χαρακτήρες με κωδικό 0 (\000) ως και 31 (\037) και ο 127 (\177)

`int iscntrl(int c)`

Ερωτήσεις?

- Διαβάστε τις σημειώσεις, διαβάστε τις διαφάνειες και δείτε τα videos **πριν** ρωτήσετε
- **Συμβουλευτείτε** τη σελίδα ερωταποκρίσεων του μαθήματος

  https://qna.c-programming.allos.gr
- **Στείλτε** τις ερωτήσεις σας πριν και μετά το μάθημα στο

  c-programming-24@allos.gr

- Εάν έχετε **πρόβλημα** με κάποιο κώδικα στείλτε τον κώδικα ως κείμενο με copy/paste. Εάν θεωρείτε ότι επιπλέον βοηθά και ένα στιγμιότυπο οθόνης, είναι καλοδεχούμενο.
- Επαναλαμβάνουμε : Μην στείλετε ποτέ κώδικα ως εικόνα μας είναι παντελώς άχρηστος!

Πρακτική 1 – Σύγκριση χαρακτήρων

Γράψτε τη συνάρτηση:

```
bool isSameLetter(char a, char b, bool caseInsensitive)
```

η οποία θα πρέπει να απαντά στο ερώτημα:

«είναι οι χαρακτήρες `a` και `b` , χαρακτήρες κειμένου και ίδιοι»;

Για τη σύγκριση θα πρέπει να λαμβάνει υπόψη του εάν η παράμετρος είναι αληθής, οπότε η σύγκριση θα πρέπει να θεωρεί τα πεζά και τα κεφαλαία ίσα μεταξύ τους.

Αναπαράσταση κειμένων

Αναπαράσταση κειμένου ως ομάδα (πίνακας) χαρακτήρων

Κείμενα

Τα κείμενα στη C είναι απλά πίνακες χαρακτήρων. Η μόνη διαφορά είναι ότι το επόμενο στοιχείο από τον τελευταίο χαρακτήρα θα πρέπει να περιέχει τον κωδικό ASCII **0**, αντί να χρειάζεται να είναι γνωστό το μήκος τους!

Άρα το κείμενο Hello θα μπορούσε να γραφεί όπως ξέρουμε ως πίνακας:

```
char greeting[] = { 'H', 'e', 'l', 'l', 'o', 0 };
```

NULL terminated string

Ένωση σταθερών κειμένων κατά το compilation ως εξής:
`"one,""two"",three"`
που ταυτίζεται με το
`"one,two,three"`

Αλλά αυτό είναι και κάπως "κουραστικό". Γι'αυτό το ισοδύναμα γράφεται:

```
char greeting[] = "Hello";
```

το οποίο τοποθετεί από μόνο του το πρόσθετο στοιχείο με κωδικό 0. Δηλαδή ο πίνακας όπως και πριν είναι με 6 στοιχεία.

Η χρήση του μηδενικού στο τέλος του κειμένου δεν είναι σπατάλη μνήμης ενός χαρακτήρα, αλλά είναι οικονομία αφού για την αποθήκευση του μήκους ενός κειμένου εν γένει θα χρησιμοποιούνταν μία ακέραια μεταβλητή τουλάχιστον 2 ή 4 bytes.

Εάν βέβαια χρειαστεί να γνωρίζουμε το μήκος του κειμένου, τότε θα χρειαστεί να το υπολογίσουμε:

```
int calculateLength(char text[]) {
    int length;
    for (length = 0; text[length] != 0; length++) {
        ;
    }
    return length;
}
```

Πρακτική 2 – Αντιγραφή κειμένου

Γράψτε τη συνάρτηση:

```
bool copyText(char from[], char to[], int lengthOfTo)
```

η οποία θα πρέπει να αντιγράφει το κείμενο που έχει η `from` στον πίνακα/κείμενο `to`, το οποίο έχει πλήθος στοιχείων `lengthOfTo`.

Εάν ο πίνακας `to` έχει αρκετά στοιχεία για να χωρέσει το κείμενο, τότε γίνεται η αντιγραφή και επιστρέφεται αληθές. Εάν δεν έχει αρκετά στοιχεία, τότε αντιγράφονται όσοι χαρακτήρες χωρούν και επιστρέφεται ψευδές.

Ερωτήσεις?

- Διαβάστε τις σημειώσεις, διαβάστε τις διαφάνειες και δείτε τα videos **πριν** ρωτήσετε
- **Συμβουλευτείτε** τη σελίδα ερωταποκρίσεων του μαθήματος

  https://qna.c-programming.allos.gr
- **Στείλτε** τις ερωτήσεις σας πριν και μετά το μάθημα στο

  c-programming-24@allos.gr

- Εάν έχετε **πρόβλημα** με κάποιο κώδικα στείλτε τον κώδικα ως κείμενο με copy/paste. Εάν θεωρείτε ότι επιπλέον βοηθά και ένα στιγμιότυπο οθόνης, είναι καλοδεχούμενο.
- Επαναλαμβάνουμε : Μην στείλετε ποτέ κώδικα ως εικόνα μας είναι παντελώς άχρηστος!

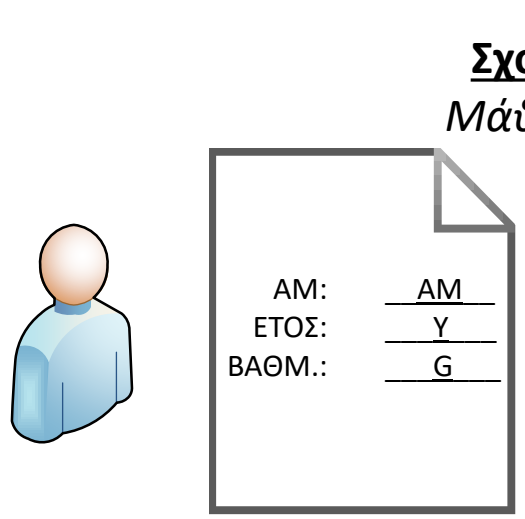
Δομές δεδομένων

Οργάνωση πληροφορίας

Συσχέτιση της πληροφορίας 1/2

Όταν περιγράφουμε ένα πρόβλημα με μαθηματικό τρόπο, δηλαδή όταν το μοντελοποιούμε, υπάρχουν ποσότητες οι οποίες σχετίζονται μεταξύ τους πιο στενά από τις υπόλοιπες. Για να τις παραστήσουμε όλες τις ποσότητες χρησιμοποιούνται απλές μεταβλητές, οπότε δεν γίνεται εμφανής αυτή η συσχέτηση. Χρειαζόμαστε λοιπόν έναν τρόπο που να την αναδεικνύει.

Για παράδειγμα για να περιγράψουμε έναν **φοιτητή** και τον βαθμό του σε ένα μάθημα, ας υποθέσουμε ότι αρκεί:

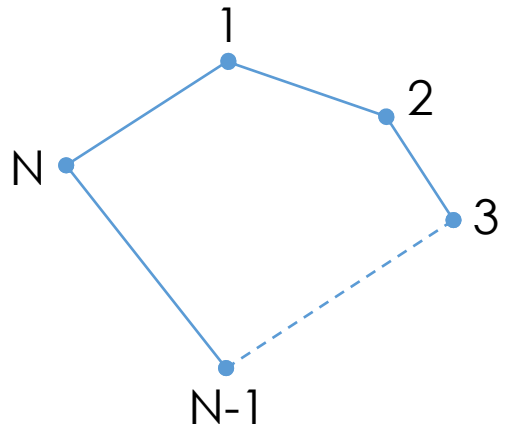


- Το όνομα της σχολής **S**
- Ο τίτλος του μαθήματος **T**
- Ο αριθμός μητρώου του φοιτητή **ΑΜ**
- Το έτος εξέτασης **Υ** και
- Ο βαθμός στο μάθημα **G**

Όμως τα ΑΜ, Υ και G έχουν στενότερη σχέση μεταξύ τους απ'ότι με τις υπόλοιπες μεταβλητές.

Συσχέτιση της πληροφορίας 2/2

Επίσης για να περιγράψουμε ένα πολύγωνο στο επίπεδο χρειαζόμαστε:



- Το πλήθος των κορυφών του (N) και
- Τις συντεταγμένες τους, (x1,y1), (x2,y2), (x3,y3), κ.ο.κ

Όμως η σχέση του κάθε x με το αντίστοιχο y (ιδιαίτερα αφού λογικά τα x θα έχουν δηλωθεί ως πίνακας x[0], x[1], x[2],... και τα y ομοίως y[0], y[1], y[2],...) **δεν γίνεται εμφανής**.

Η κατάσταση «επιδεινώνεται» σε αυτό το παράδειγμα επειδή, παρατηρώντας τις μεταβλητές του (δηλαδή τους δύο πίνακες x και y), δεν γίνεται καθόλου εμφανές ότι οι δύο πίνακες επιβάλλεται να έχουν το ίδιο πλήθος στοιχείων. Αυτό φαίνεται μόνο από τη συμπεριφορά του κώδικα προς αυτά!

Ορίζοντας τις έννοιες

Σε κάθε μία από αυτές τις περιπτώσεις δεν λύνουμε κάποιο πρόβλημα, μόνο περιγράφουμε/ορίζουμε κάποιες έννοιες που εμπλέκονται στο πρόβλημα που μας ενδιαφέρει.

Έτσι ο ορισμός στη πλήρη του μορφή γράφεται όπως φαίνεται δίπλα.

Ακολούθως θα τα δούμε αναλυτικά.

```
typedef
struct _ennoia {
// πεδία που περιγράφουν την έννοια πχ
    double x;
    double y;
} Ennoia ;
```

Δήλωση δομών δεδομένων (struct) 1/2

Η λύση που παρέχει η C για τέτοιες περιπτώσεις είναι η κατασκευή μίας ονοματισμένης ομάδας μεταβλητών με τη χρήση της λέξης κλειδί **`struct`** η οποία γράφεται για τα δύο παραδείγματα ως ακολούθως:

```
struct _student {
    int AM;
    int Y;
    double G;
};
struct _student someStudent;
```

Πεδία (fields) ή **Μέλη** (members) οποιουδήποτε τύπου δεδομένων

```
struct _point {
    double x;
    double y;
};
struct _point points[N];
```

Ισχύουν οι γενικοί κανόνες ονομασίας.

Λειτουργεί ως **τύπος δεδομένων**

ή

```
typedef struct _student {
    int AM;
    int Y;
 double G;
} Student;

Student someStudent;
```

```
typedef struct _point {
    double x;
    double y;
} Point;


Point points[N];
```

Δήλωση δομών δεδομένων (struct) 2/2

Κάθε δομή δεδομένων μπορεί να έχει οσαδήποτε ονοματισμένα μέλη (πεδία) οποιουδήποτε τύπου δεδομένων.

Για παράδειγμα μπορεί εκτός από αριθμητικά που είδαμε, να έχει ως πεδία:

- πίνακες & κείμενα
- άλλες δομές δεδομένων

Δεν επιτρέπεται όμως να περιέχει δομές του ίδιου τύπου.

Μια τέτοια αναδρομική δήλωση θα δημιουργούσε αυτόματα απαίτηση για άπειρη μνήμη με την δήλωση της πρώτης μεταβλητής αυτού του τύπου!

```
struct myDate {
    int monthDay;
    int monthNum;
    int year;
}


struct Person {
    char firstName[100];
    char lastName[100];
    struct myDate birthday;
    struct Person mother;
    int heightInCm;
    double weightInKgr;
}
```

Δήλωση νέων τύπων δεδομένων - typedef

Όταν χρησιμοποιείται η γραφή με το typedef πρέπει να είναι ξεκάθαρα τα ακόλουθα:

- Απαιτείται η δήλωση του ονόματος του νέου τύπου δεδομένων μετά το κλείσιμο του άγκιστρου και πριν το ; αλλιώς πρόκειται για λάθος που ο compiler το επισημαίνει απλώς ως προειδοποίηση (warning)

```
typedef struct _point {
    double x;
    double y;
} Point ;

Point points1[N];
```

- Η χρήση του δεν αναιρεί τη **παράλληλη** και **ισοδύναμη** χρήση του struct Point,

```
struct _point points2[N];
```

- Εάν (εκ παραδρομής) παραληφθεί η λέξη κλειδί `typdef`, τότε το λεκτικό που ακολουθεί το κλείσιμο του αγκίστρου της struct (εδώ το `planePoint1`) θεωρείται απλά ως μεταβλητή του τύπου `struct PointOnPlane`, η οποία «έτυχε» να δηλώνεται ταυτόχρονα με τη δομή δεδομένων.

```
struct PointOnPlane {
    double x;
    double y;
} planePoint1 ;
```

> Άρα συντακτικά είναι αποδεκτό, ενώ λογικά είναι λάθος!

Χρήση δομών δεδομένων

Η χρήση μιας δομής δεδομένων γίνεται όπως παρουσιάζεται στον διπλανό κώδικα.

Κάθε πεδίο ή μέλος δεν χρησιμοποιείται αυτόνομα,

...αλλά μόνο σε συνδυασμό με το όνομα της **μεταβλητής** του τύπου struct (όχι του ονόματος της δομής), ενωμένα με τον **τελεστή «τελεία»** .

...και ο συνδυασμός αυτός λειτουργεί όπως μία **οποιαδήποτε** μεταβλητή.

Όμως η μεταβλητή του τύπου struct (εδώ η p) μπορεί να χρησιμοποιηθεί αυτόνομα.

Επίσης πριν από τη δήλωση έστω και μίας μεταβλητής του εκάστοτε τύπου struct (π.χ. struct _point ή Point) δεν έχει δημιουργηθεί καμία μεταβλητή και κατά συνέπεια δεν καταλαμβάνεται καθόλου μνήμη.

Αντίστροφα με κάθε δήλωση μίας τέτοιας μεταβλητής (π.χ. εδώ η p), δημιουργούνται μία μεταβλητή για κάθε μέλος (πεδίο). Έτσι για παράδειγμα αφού κάθε double καταλαμβάνει 8bytes

```
typedef struct _point {
    double x;
    double y;
} Point;
```

0 bytes

```
Point A;  8+8 bytes
Point points[N]; (8+8)*N bytes
```

```
int quadrant(Point p) {
    if (p.x > 0) {
        if (p.y > 0) {
            return 1;
        } else {
            return 4;
        }
    } else {
        if (p.y > 0) {
            return 2;
        } else {
            return 3;
        }
    }
}
```

p.y

Αρχικοποίηση μεταβλητών τύπου struct

Σε πίνακα μπορώ να δίνω μαζικά τιμές στα στοιχεία, βάζοντάς τες στη σειρά μέσα σε άγκιστρα.

Με τις δομές πως μπορώ να κάνω κάτι ανάλογο;

```
typedef struct _point {
    double x;
    double y;
} Point;

Point center = {
   .x = 1.1,
   .y = 2.2
};
```

Παράδειγμα

Η παρακάτω συνάρτηση επιστρέφει το κέντρο βάρους ενός πολυγώνου με N πλευρές.

```
typedef
struct _point {
    double x;
    double y;
}
Point;

Point calcCenterPoint(int N, Point points[N]) {
    Point center;
    center.x = 0.0;
    center.y = 0.0;
    for (int i = 0; i < N; ++i) {
        center.x += points[i].x;
        center.y += points[i].y;
    }
    center.x /= N;
    center.y /= N;
    return center;
}
```

Δήλωση της δομής «σημείο» με συντεταγμένες x και y τύπου double

Ορισμός τύπου δεδομένων Point (ώστε να μην γράφουμε συνέχεια "struct _point")

Πλήθος κορυφών του πολυγώνου

Πίνακας με τα σημεία των κορυφών του πολυγώνου

Δήλωση της μεταβλητής που θα κρατάει το κέντρο βάρους του πολυγώνου. Και αυτή σημείο είναι.

Αρχική τιμή των συντεταγμένων, ώστε να αποθηκευθεί το άθροισμα.

Άθροιση των αντίστοιχων συντεταγμένων του κάθε **`points[i]`** σημείου

Διαίρεση με το πλήθος των πλευρών/κορυφών/σημείων ώστε να προκύψει η μέση τιμή

Επιστροφή του αποτελέσματος (ολόκληρο το σημείο)

Πρακτική 3 – Πρώτη λέξη

Γράψτε τη συνάρτηση:

```
Word getFirstWord(char text[], int startIndex)
```

η οποία θα πρέπει να επιστρέφει μια δομή Word η οποία περιγράφει:

- είτε τη θέση και το μήκος της 1ης λέξης μέσα στο κείμενο text (ξεκινώντας από τη θέση `startIndex`)
- είτε μία τέτοια δομή με θέση και μήκος -1, εάν δεν βρεθεί λέξη.

Η δομή θα πρέπει να έχει δύο ακέραια πεδία:

- το **`index`** που θα είναι η θέση της 1ης λέξης και
- το **`length`** που θα είναι το μήκος της λέξης σε χαρακτήρες.

Ερωτήσεις?

- Διαβάστε τις σημειώσεις, διαβάστε τις διαφάνειες και δείτε τα videos **πριν** ρωτήσετε
- **Συμβουλευτείτε** τη σελίδα ερωταποκρίσεων του μαθήματος

  https://qna.c-programming.allos.gr
- **Στείλτε** τις ερωτήσεις σας πριν και μετά το μάθημα στο

  c-programming-24@allos.gr

- Εάν έχετε **πρόβλημα** με κάποιο κώδικα στείλτε τον κώδικα ως κείμενο με copy/paste. Εάν θεωρείτε ότι επιπλέον βοηθά και ένα στιγμιότυπο οθόνης, είναι καλοδεχούμενο.
- Επαναλαμβάνουμε : Μην στείλετε ποτέ κώδικα ως εικόνα μας είναι παντελώς άχρηστος!