Προγραμματισμός Ι (ΗΥ120)

Διάλεξη 3:

Είσοδος / Έξοδος, Βασικοί Τύποι, Δήλωση Μεταβλητών

Χρήστος Δ. Αντωνόπουλος
26/9/2016

Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, Πανεπιστήμιο Θεσσαλίας

Βασικοί τύποι της C

Όνομα	Τύπος / Κωδικοποίηση	Μέγεθος (bytes)
`char`	Χαρακτήρας	1
`int`	Ακέραιος	2 ή 4 (*)
`float`	Πραγματικός απλής ακρίβειας	4
`double`	Πραγματικός διπλής ακρίβειας	8
`void`	‘Τίποτα’	

(*) Εξαρτάται από την αρχιτεκτονική του επεξεργαστή

Προσδιοριστές μεγέθους / πεδίου τιμών

Όνομα	Τύποι / Ερμηνεία	Μέγεθος (bytes)
`short`	«Μικρός» `int`	1, 2 ή 4 (*)
`long`	«Μεγάλος» `int`	2 ή 8 (*)
`long`	«Μεγάλος» `double`	12 ή 16 (*)

(*) Εξαρτάται από την αρχιτεκτονική του επεξεργαστή

- `short / long` χωρίς τύπο: υπονοείται `int`

Όνομα	Τύποι	Πεδίο Τιμών
`signed`	`int / char` με πρόσημο	$[-2^{N-1}...2^{N-1}-1]$
`unsigned`	`int / char` χωρίς πρόσημο	$[0...2^{N}-1]$

N: αριθμός bits

- Τύπος χωρίς `signed / unsigned:` υπονοείται `signed`

Τύποι / Μεγέθη / Πεδία τιμών (σε επεξεργαστές Intel x86)

`unsigned char`	1 byte	0 ... 255
`char`	1 byte	-128 ... 127
`unsigned int`	4 bytes	0 ... 4,294,967,295
`short int`	2 bytes	-32,768 ... 32,767
`int`	4 bytes	-2,147,483,648 ... 2,147,483,647
`unsigned long`	4 bytes	0 ... 4,294,967,295
`long`	4 bytes	-2,147,483,648 ... 2,147,483,647
`float`	4 bytes	1.17549435 * (10^-38) ... 3.40282347 * (10^+38)
`double`	8 bytes	2.2250738585072014 * (10^-308) ... 1.7976931348623157 * (10^+308)
`long double`	12 bytes	3.4 * (10^-4932) ... 1.1 * (10^4932)

Και πώς τα βρίσκω;;;

```
/* εκτύπωση μεγέθους βασικών τύπων */

#include <stdio.h>

int main(int argc, char *argv[]) {
  printf("sizeof(char)=%d bytes\n",sizeof(char));
  printf("sizeof(short)=%d bytes\n",sizeof(short);
  printf("sizeof(int)=%d bytes\n",sizeof(int));
  printf("sizeof(long)=%d bytes\n", sizeof(long));
  printf("sizeof(float)=%d bytes\n",sizeof(float));
  printf("sizeof(double)=%d bytes\n",sizeof(double));
  printf("sizeof(long double) = %d bytes\n", sizeof (long double));
  printf("\n");

  return(0);
}
```

Ερμηνεία Δυαδικών Δεδομένων

- Το υλικό του Η/Υ δεν γνωρίζει την σημασία των δεδομένων που αποθηκεύονται στην μνήμη.
  - Τα bits ερμηνεύονται με βάση την κωδικοποίηση που αντιστοιχεί στον τύπο που έχει η μεταβλητή μέσω της οποίας προσπελαύνεται η μνήμη.
  - Π.χ.:
    - `01100001: 'a' (char)` ή `193 (short)`
    - `11111111: -1 (short)` ή `255 (unsigned short)`
  - Η ερμηνεία των περιεχομένων της μνήμης πρέπει να γίνεται με πλήρη επίγνωση και ιδιαίτερη προσοχή…
    - … διαφορετικά: σημασιολογικά λάθη.

Δηλώσεις Μεταβλητών

- Οι δηλώσεις μεταβλητών δίνονται πριν από (σχεδόν) όλες τις υπόλοιπες εντολές ενός προγράμματος.
- Μορφή των εκφράσεων δήλωσης είναι:
  - `<τύπος> <όνομα> ;`
  - `<τύπος> <όνομα>,..., <όνομα> ;`
  - `<τύπος> <όνομα> = <τιμή> ;`
- Κάθε μεταβλητή (και συνάρτηση) πρέπει να έχει διαφορετικό όνομα.
- Κατά την δήλωση της, μια μεταβλητή μπορεί προαιρετικά να λάβει και μια αρχική τιμή.
  - Το πρόθεμα `const` σε συνδυασμό με την ανάθεση αρχικής τιμής υποδηλώνει ότι η τιμή της μεταβλητής δε μπορεί να αλλάξει κατά την διάρκεια της εκτέλεσης.

```
char c;        /* μεταβλητή χαρακτήρα με όνομα c */

short si=1;    /* μεταβλητή μικρού ακεραίου με
                  όνομα si και (literal) τιμή 1 */

const float pi=3.1415;    /* μεταβλητή πραγματικού
                             αριθμού απλής ακριβείας
                             με όνομα pi και (literal)
                             τιμή 3.1415              */

int c;         /* μεταβλητή ακεραίου με όνομα c */
```


ο μεταγλωττιστής θα διαμαρτυρηθεί, καθώς το όνομα c χρησιμοποιείται

```
$ gcc var_decl_mistake.c -o var_decl_mistake
var_decl_mistake.c: In function `main':
var_decl_mistake.c:11: error: conflicting types for 'c'
var_decl_mistake.c:2: error: previous declaration of 'c' was here
```

Μεταβλητές και Μνήμη

- Κατά την εκτέλεση του προγράμματος, η δήλωση μιας μεταβλητής οδηγεί στην δέσμευση αντίστοιχου χώρου μνήμης για την αποθήκευση των τιμών της.
  - Η δέσμευση μνήμης γίνεται όταν «ενεργοποιείται» η δήλωση της μεταβλητής
    - Λίγη υπομονή ...
- Οι μεταβλητές ενός προγράμματος καταλαμβάνουν (συνήθως) συνεχόμενες περιοχές μνήμης: εκεί που τελειώνει η περιοχή της μεταβλητής που δηλώθηκε πρώτη, αρχίζει η περιοχή της μεταβλητής που δηλώθηκε δεύτερη, κλπ.
  - Αυτό δεν ισχύει πάντα. Εξαρτάται από την υλοποίηση του μεταφραστή ή/και του περιβάλλοντος εκτέλεσης της γλώσσας

Είσοδος / Έξοδος (Input / Output)

- Το πρόγραμμα πρέπει να μπορεί να **εισάγει/εξάγει** δεδομένα από/προς το «περιβάλλον» του
  - Συνήθως από το πληκτρολόγιο και προς την οθόνη του Η/Υ.
  - Η από και προς το δίκτυο, το δίσκο κλπ.
- Πώς;
  - Ειδικές εντολές **εισόδου/εξόδου** (input/output commands).

Η Βιβλιοθήκη stdio

- Οι συναρτήσεις εισόδου/εξόδου υλοποιούνται στην βιβλιοθήκη **`stdio`**
  - Τα «πρωτότυπα» των συναρτήσεων της `stdio` υπάρχουν στο αρχείο κεφαλίδων `stdio.h`
  - η χρήση της οποίας πρέπει να δηλώνεται στην αρχή του προγράμματος με την εντολή **`#include <stdio.h>`**
- Περιέχει και πολλές άλλες συναρτήσεις
  - Καλό σας διάβασμα…
- Περισσότερα για το τι είναι και πως φτιάχνεται μια βιβλιοθήκη πολύ πολύ αργότερα ...

Εντολές Εισόδου / Εξόδου

- Οι εντολές εισόδου/εξόδου είναι ειδικές συναρτήσεις που βρίσκονται στην βιβλιοθήκη **`stdio`**
  - Οι δηλώσεις βρίσκονται στο αρχείο επικεφαλίδων **`stdio.h`** και πρέπει να συμπεριληφθούν στο πρόγραμμα
  - Εντολή: `#include <stdio.h>`
- **`getchar`**: διαβάζει ένα (τον επόμενο) χαρακτήρα από την είσοδο του προγράμματος (πληκτρολόγιο).
- **`putchar`**: γράφει ένα χαρακτήρα στην έξοδο του προγράμματος (οθόνη).
- **`scanf`**: διαβάζει από την είσοδο χαρακτήρες και αναθέτει τιμές σε συγκεκριμένες μεταβλητές
  - Διαβάζονται όσοι χαρακτήρες είναι απαραίτητοι για να ανατεθεί τιμή στις «συγκεκριμένες» μεταβλητές.
- **`printf`**: εκτυπώνει στην έξοδο κείμενο καθώς και τιμές από συγκεκριμένες μεταβλητές.

Ένα Παράδειγμα

```
#include <stdio.h>

int main(int argc, char* argv[]) {

  putchar('h');
  putchar('e');
  putchar('l');
  putchar('l');
  putchar('o');
  putchar(' ');
  putchar('w');
  putchar('o');
  putchar('r');
  putchar('l');
  putchar('d');
  putchar('\n');

  return(0);
}
```

```
>./myprog<enter>
hello world
>
```

Κι Άλλο Παράδειγμα

```
#include <stdio.h>

int main(int argc, char* argv[]) {

  putchar('5');
  putchar(' ');
  putchar('+');
  putchar(' ');
  putchar('2');
  putchar(' ');
  putchar('=');
  putchar('=');
  putchar(' ');
  putchar('7');
  putchar('\n');

  return(0);
}
```

```
>./myprog<enter>
5 + 2 == 7
>
```

```c
#include <stdio.h>

int main(int argc, char* argv[]) {
  char c1,c2,c3;

  putchar('3');
  putchar(' ');
  putchar('c');
  putchar('h');
  putchar('a');
  putchar('r');
  putchar('s');
  putchar(':');
  putchar('\n');

  c1=getchar();
  c2=getchar();
  c3=getchar();

  putchar(c1);
  putchar(c2);
  putchar(c3);
  putchar('\n');
  return(0);
}
```

```
>./myprog<enter>
3 chars:
a2$<enter>
a2$
>
>./myprog<enter>
3 chars:
a bcdefg<enter>
a b
>
```

το πρόγραμμα δεν διαβάζει τα «επιπλέον» δεδομένα αφού δεν υπάρχουν αντίστοιχες εντολές στον κώδικα

Η printf()

- Δέχεται μια παράμετρο σε μορφή συμβολοσειράς από εκτυπώσιμους χαρακτήρες ASCII
  - Προαιρετικά και έναν **απεριόριστο** αριθμό «εκφράσεων αποτίμησης».
- Η πρώτη παράμετρος περιέχει
  - Το κυριολεκτικό κείμενο προς εκτύπωση, και
  - Τους προσδιορισμούς εκτύπωσης (format specifiers) για τις τιμές κάθε μιας έκφρασης που δίνεται ως επιπλέον παράμετρος.
  - Για τις συμβάσεις των προσδιορισμών εκτύπωσης δείτε (οπωσδήποτε) το εγχειρίδιο της γλώσσας!

Η printf()

- Αν οι εκφράσεις αποτίμησης είναι λιγότερες από τους προσδιορισμούς εκτύπωσης, ο μεταγλωττιστής δεν εμφανίζει λάθος!

- Αν οι προσδιορισμοί εκτύπωσης δεν είναι συμβατοί με τους αντίστοιχους τύπους των εκφράσεων αποτίμησης, ο μεταγλωττιστής δίνει μόνο προειδοποίηση!

Σε απλά ελληνικά…


```
#include <stdio.h>

int main(int argc, char* argv[]) {
  int a=1,b=2;

  printf("a is %d, b is %d and a+b is %d\n”,a, b, a+b);
  return(0);
}
```

```
>./myprog<enter>
a is 1, b is 2 and a+b is 3
>
```

Προσδιοριστές τύπου printf

- %c : char
- %d : int
- %x : hex
- %f : float
- %lf : double
- %s : συμβολοσειρά
- %p : διεύθυνση
- Πηγή: http://en.cppreference.com/w/cpp/io/c/printf

Επιπλέον λειτουργίες printf

- Εκτύπωση αριθμού κινητής υποδιαστολής με 2 δεκαδικά ψηφία:

```
double x = 1.2345;
printf("%.2lf", x);
```

Εκτυπώνει:
1.23

- Εκτύπωση αριθμού κινητής υποδιαστολής με 2 δεκαδικά ψηφία και συνολικό πλάτος 6 θέσεις:

```
double x = 12.345;
printf("%6.2lf", x);
```

Εκτυπώνει:
12.34

1 κενό + 2 ακέραια ψηφία + τελεία + 2 δεκαδικά → 6 θέσεις

Επιπλέον λειτουργίες printf

- Εκτύπωση ακέραιου αριθμού με συνολικό πλάτος 4 θέσεις:

```
int x = 12;
printf("%4d", x);
```

Εκτυπώνει:
12

2 κενά + 2 ακέραια ψηφία $\rightarrow$ 4 θέσεις

- Εκτύπωση ακέραιου αριθμού με συνολικό πλάτος 4 θέσεις και "γέμισμα" με μηδενικά:

```
int x = 12;
printf("%04d", x);
```

Εκτυπώνει:
0012

H scanf

- Δέχεται μια παράμετρο σε μορφή συμβολοσειράς από εκτυπώσιμους χαρακτήρες ASCII, και έναν **απεριόριστο** αριθμό **διευθύνσεων** μεταβλητών.
  - Η πρώτη παράμετρος περιέχει τους προσδιορισμούς ανάγνωσης για τις τιμές κάθε μιας μεταβλητής η **διεύθυνση** της οποίας δίνεται ως παράμετρος.
    - Για τις συμβάσεις των προσδιορισμών εκτύπωσης δείτε (οπωσδήποτε) το εγχειρίδιο της γλώσσας!
  - Αν οι προσδιορισμοί ανάγνωσης δεν είναι συμβατοί με τους τύπους των αντίστοιχων εκφράσεων αποτίμησης που δίνονται ως παράμετροι, τότε ο μεταγλωττιστής δίνει προειδοποίηση.

Και για να συνεννοηθούμε…


```
#include <stdio.h>

int main(int argc, char* argv[]) {
  int a,b;

  printf("enter 2 int values: ");

  scanf("%d %d",&a,&b);

  printf("a is %d and b is %d\n",a,b);
  return(0);
}
```

προσοχή: πάντα να υπάρχει το **&** πριν το όνομα της μεταβλητής (επεξήγηση προσεχώς…)

```
>./myprog<enter>
enter 2 int values: 5 10 15<enter>
a is 5 and b is 10
>
```

το πρόγραμμα δεν διαβάζει τα «επιπλέον» δεδομένα αφού δεν υπάρχουν αντίστοιχες εντολές στον κώδικα

scanf format specifiers

- %c : char
- %d : int
- %x : hex
- %g : float
- %lf : double
- %s : συμβολοσειρά
- %p : διεύθυνση
- Πηγή: http://en.cppreference.com/w/cpp/io/c/scanf

scanf παραδείγματα

- Ανάγνωση χαρακτήρα:

```
char letter;
scanf(" %c", &letter);
```

βάζετε πάντα ένα κενό ανάμεσα στο " και στο %

- Ανάγνωση τιμών όταν η είσοδος έχει τη μορφή 10/2011:

```
int month, year;
scanf("%d/%d", &month, &year);
```

- Ανάγνωση τιμών όταν η είσοδος έχει τη μορφή PRICE: 3.75 euro

```
double price;
scanf("PRICE: %lf euro",  &price);
```

Ανακατεύθυνση Εισόδου/Εξόδου

- Οι πράξεις εισόδου / εξόδου διαβάζουν / γράφουν από την είσοδο / έξοδο του προγράμματος.
  - Συνήθως η είσοδος αντιστοιχεί στο πληκτρολόγιο (με εκτύπωση των χαρακτήρων που εισάγονται στην οθόνη) και η έξοδος στην οθόνη.
  - Τόσο η είσοδος όσο και η έξοδος ενός προγράμματος μπορούν να ανακατευθυνθούν
    - Π.χ. σε αρχεία έτσι ώστε οι πράξεις εισόδου να διαβάζουν δεδομένα από ένα αρχείο και οι πράξεις εξόδου να γράφουν δεδομένα σε ένα αρχείο.
    - Τα αρχεία που δίνονται για είσοδο πρέπει να είναι αρχεία κειμένου (ASCII) και τα αρχεία που δημιουργούνται είναι ASCII.

είσοδος


πρόγραμμα

πρόγραμμα

αρχείο

αρχείο

πρόγραμμα

αρχείο

πρόγραμμα

αρχείο

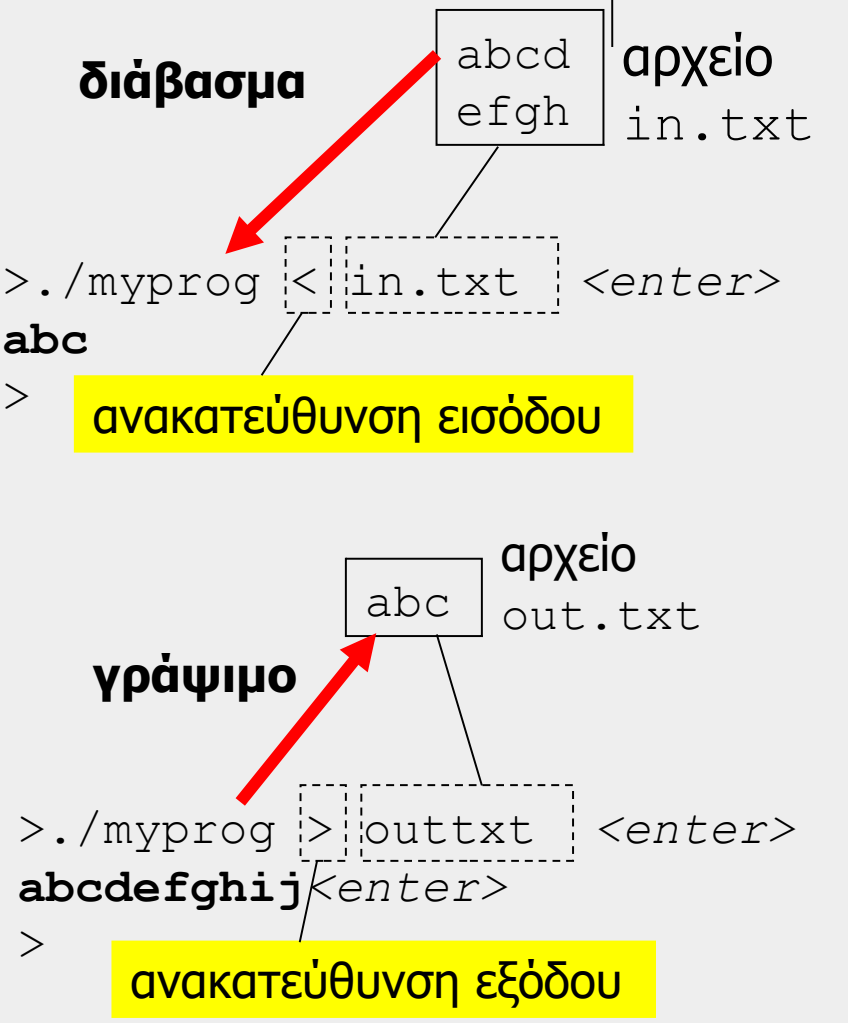
Παράδειγμα Ανακατεύθυνσης


```
#include <stdio.h>

int main(int argc, char*
argv[]) {
  char c1,c2,c3;

  c1=getchar();
  c2=getchar();
  c3=getchar();

  putchar(c1);
  putchar(c2);
  putchar(c3);
  return(0);
}
```



Πώς δουλεύει;

- Όταν το πρόγραμμα φτάσει σε μια εντολή εισόδου (π.χ. `getchar` ή `scanf`) τότε η εκτέλεση σταματά μέχρι να υπάρξουν δεδομένα έτοιμα προς ανάγνωση.
- Αν τα δεδομένα εισάγονται από το πληκτρολόγιο οι χαρακτήρες «στέλνονται» στο πρόγραμμα **αφού** πατηθεί το πλήκτρο `<enter>` (μέσω του οποίου δημιουργείται αυτόματα και ο χαρακτήρας `'\n'`).
- Αν τα δεδομένα εισόδου δίνονται μέσω ενός αρχείου (με ανακατεύθυνση), και το πρόγραμμα επιχειρήσει να διαβάσει χωρίς να υπάρχουν άλλα δεδομένα τότε επιστρέφεται η τιμή (σταθερά) EOF (end of file).
  - Το πρόγραμμα πρέπει να κάνει κατάλληλο έλεγχο.

Επανάληψη: Στοιχεία ενός προγράμματος

- Literal / κυριολεκτικό
  - Μια συγκεκριμένη τιμή που εμφανίζεται σε ένα πρόγραμμα (πχ. 3, “Hello”, 5.19)
- Variable / μεταβλητή
  - Μια ονομασμένη θέση στη μνήμη, όπου αποθηκεύεται μια τιμή. Η τιμή που αποθηκεύεται σε μια μεταβλητή μπορεί να αλλαχθεί κατά την εκτέλεση του προγράμματος.
- Constant / σταθερά
  - Μια ονομασμένη θέση στη μνήμη όπου αποθηκεύεται μια τιμή. Η τιμή που αποθηκεύεται σε μια σταθερά ΔΕ μπορεί να αλλαχθεί κατά την εκτέλεση του προγράμματος

Επανάληψη: Στοιχεία ενός προγράμματος

- Expression / έκφραση
  - Οτιδήποτε μπορεί να αποτιμηθεί
  - Μια έκφραση συνήθως αποτελείται από ένα συνδυασμό μεταβλητών, literals και τελεστών
    - Αργότερα θα δούμε κι άλλα είδη εκφράσεων.
- Statement / εντολή
  - Μια οδηγία, στο πρόγραμμα, η οποία μπορεί να εκτελεστεί.
  - Κάθε έκφραση γίνεται εντολή όταν βάλουμε στο τέλος της το ερωτηματικό.

Επανάληψη: Στοιχεία ενός προγράμματος

- Function / συνάρτηση
  - Ένα ονομασμένο γκρουπ (block) εντολών, που κάνουν μια καλά ορισμένη λειτουργία (πχ. εκτύπωση μηνύματος στην οθόνη)
  - Μια συνάρτηση μπορεί να δέχεται κάποια δεδομένα ως παραμέτρους και να επιστρέφει κάποια τιμή.
- Library / βιβλιοθήκη
  - Μια συλλογή συναρτήσεων που υλοποιούν παρεμφερείς λειτουργίες
    - Η stdio είναι μια συλλογή συναρτήσεων για είσοδο και έξοδο δεδομένων (πχ είσοδο από πληκτρολόγιο, έξοδο στην οθόνη)

Μια ιεραρχία των στοιχείων ενός προγράμματος