Θεματική ενότητα 2: Μεταβλητές – σταθερές – Ι/Ο κονσόλας

Μεταβλητές

- **Ίδια χρήση με εκείνη της άλγεβρας:** $3x + 5 = y$
  x και y είναι οι *μεταβλητές*

- **Αλλά γενικευμένη: η μεταβλητή είναι μία θέση μνήμης για ένα δεδομένο. Η τιμή της μπορεί να είναι άγνωστη έως ότου εκτελεσθεί το πρόγραμμα ('run-time').**

Δήλωση μεταβλητών

- **Με *πρόταση ορισμού* (τελειώνει πάντοτε με ;)**
  - **Η μορφή:** **data_type var, var, … ;**
  - **Παράδειγμα:** **int counter1, counter2;**

- **Πού; Οι μεταβλητές δηλώνονται στην αρχή μίας συνάρτησης, αμέσως μετά το εισαγωγικό άγκιστρο {**

- **Στην C τα ονόματα των μεταβλητών σχηματίζονται από:**
  - **τα γράμματα του αλφαβήτου**
  - **τα ψηφία 0 έως 9**
  - **το χαρακτήρα υπογράμμισης __ (underscore)**
- **Το όνομα πρέπει να ξεκινά με γράμμα ή τον χαρακτήρα υπογράμμισης (στη δεύτερη περίπτωση ο επόμενος χαρακτήρας πρέπει να είναι μικρό γράμμα).**
- **Το όνομα δεν πρέπει να είναι ίδιο με δεσμευμένη λέξη.**
- **Σημαντικοί είναι μόνο οι πρώτοι 31 χαρακτήρες του ονόματος. Οι υπόλοιποι δε λαμβάνονται υπόψη.**

```
/**********************************************
                    Hydra.c
**********************************************/

#include < stdio.h >

void main() {
    int heads;
    int eyes;

    heads = 3;              /* ανάθεση τιμής */
    eyes = heads * 2;       /* υπολογισμός τιμής */

    printf("It has %d heads and %d eyes! \n", heads, eyes);
}
```

Έξοδος:

```
> It has 3 heads and 6 eyes!
```

```
/**********************************************
                    Hydra.c
**********************************************/

#include < stdio.h >

main() {
    int heads;
    int eyes;

    heads = 3;              /* ανάθεση τιμής */
    eyes = heads * 2;       /* υπολογισμός τιμής */

    printf("It has %d heads and %d eyes! \n", heads, eyes);
}
```

Δήλωσε (declare) κάθε μεταβλητή. Γράψε μία πρόταση για να τεθεί το όνομά της και το είδος της

Έξοδος:

> It has 3 heads and 6 eyes!

```
/**************************************************
                        Hydra.c
**************************************************/

#include < stdio.h >

main() {
    int heads;
    int eyes;

    heads = 3;          /* ανάθεση τιμής */
    eyes = heads * 2;   /* υπολογισμός τιμής */

    printf("It has %d heads and %d eyes! \n", heads, eyes);
}
```

Μετά τον ορισμό, θέσε τιμές στις μεταβλητές

Έξοδος:

> It has 3 heads and 6 eyes!

Ονόματα μεταβλητών

- Έγκυρα ονόματα μεταβλητών:

  totalArea *max_amount* *counter1*

  Counter1 *_temp_in_F*

- Μη έγκυρα ονόματα μεταβλητών:

  $product *total%* *3rd*

- Απαράδεκτα ονόματα μεταβλητών:

  l *x2* *maximum_number_of_students_in_my_class*

Τύποι μεταβλητών

Υπάρχουν 4 βασικοί τύποι μεταβλητών στη γλώσσα C:

Τύπος	Λέξη κλειδί στη C:
Integer	*int*
Floating point	*float*
Double	*double*
Character	*char*

Ο τύπος του χαρακτήρα (char)

- Παριστάνει απλούς χαρακτήρες του αλφάβητου της γλώσσας. Βρίσκεται ανάμεσα σε απλά εισαγωγικά (π.χ. *'C'*, *'2'*, *'*'*, *')*).
- **Δήλωση**: char choice;
- **Δήλωση με αρχική τιμή**: char choice='A';
- **Εκτύπωση**: με χρήση της συνάρτησης *printf()* και του προσδιοριστή (specifier) *%c*.

```
printf( "The character is %c\n", choice );
```

- Εάν αντί του *%c* χρησιμοποιηθεί ο *%d*, η *printf()* θα εμφανίσει τον ASCII κωδικό του χαρακτήρα. Η πρόταση

```
printf( "The ASCII Code of %c is %d\n", choice,choice);
```

θα τυπώσει

```
The ASCII Code of A is 65
```

Ο τύπος του χαρακτήρα

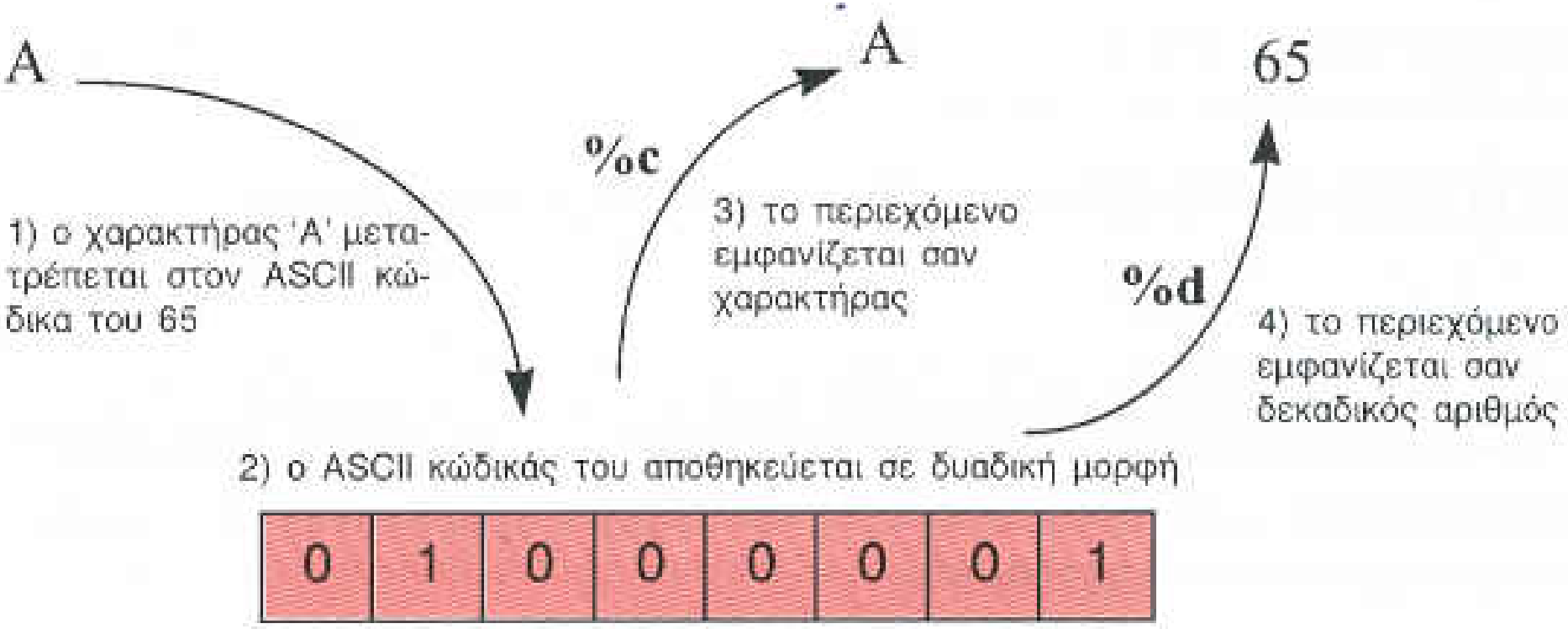
- **Εισαγωγή:** με χρήση της συνάρτησης *scanf()* και του προσδιοριστή (specifier) *%c*. Η πρόταση

```
scanf( "%c", &ch );
```

διαβάζει από την κύρια είσοδο (πληκτρολόγιο) ένα χαρακτήρα και τον αποδίδει στη μεταβλητή ch. Θα πρέπει να προσεχθεί η χρήση του & πριν από τη μεταβλητή. Ονομάζεται *τελεστής διεύθυνσης* και προηγείται πάντοτε των μεταβλητών στην εντολή *scanf()*.

- **Αποθήκευση και ανάκληση ASCII χαρακτήρα:** Ο μεταγλωττιστής απαιτεί 1 byte μνήμης για την αποθήκευση της τιμής μιας μεταβλητής χαρακτήρα. Η αποθήκευση και ανάκληση παρουσιάζεται στην επόμενη διαφάνεια:

Αποθήκευση και ανάκληση ASCII χαρακτήρα



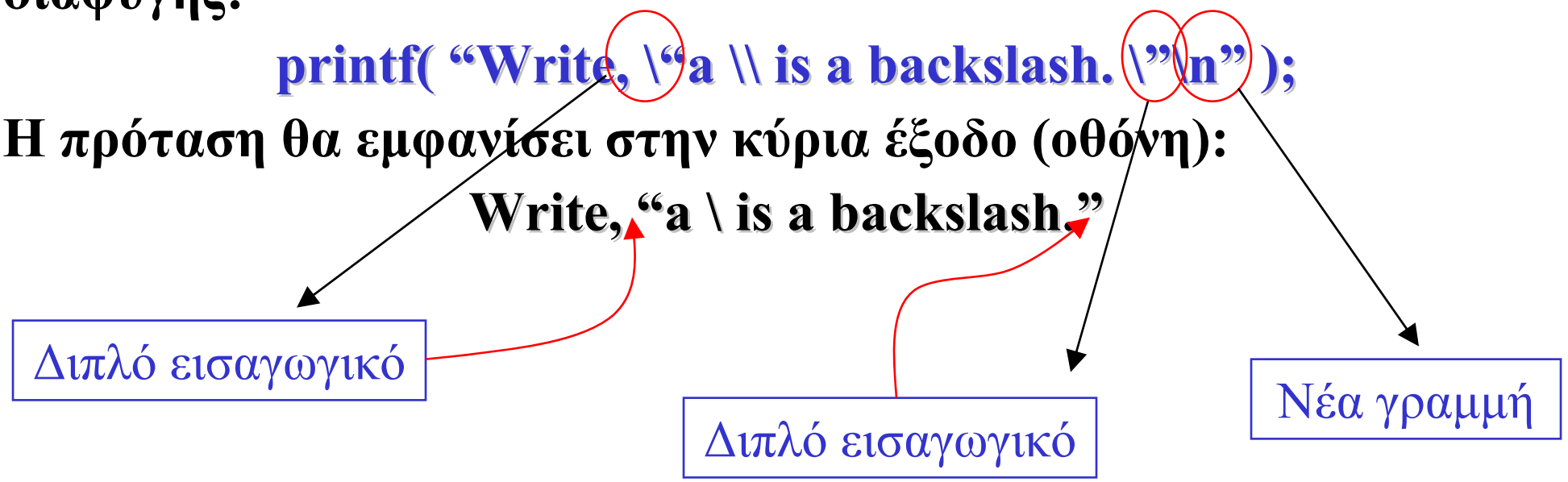
Μη εκτυπούμενοι χαρακτήρες

Οι <u>σταθερές τύπου χαρακτήρα</u> *'νέα γραμμή (new-line)'* και *'στηλοθέτης (tab)'* ανήκουν στην κατηγορία των μη εκτυπούμενων χαρακτήρων, τους οποίους η C αναπαριστά με τις *ακολουθίες διαφυγής (escape sequences)* '\n' '\t', αντίστοιχα. Η παρακάτω πρόταση δίνεται ως παράδειγμα χρήσης χαρακτήρων διαφυγής:

```
printf( "Write, \"a \\ is a backslash. \"\n" );
```

Η πρόταση θα εμφανίσει στην κύρια έξοδο (οθόνη):

```
Write, "a \ is a backslash."
```



Μη εκτυπούμενοι χαρακτήρες και αντίστοιχες ακολουθίες διαφυγής

Χαρακτήρας	ακολουθία διαφυγής	Χαρακτήρας	ακολουθία διαφυγής
συναγερμός (κουδούνι)	\a	πλάγια	\\
οπισθοχώρηση	\b	λατινικό ερωτηματικό	\?
αλλαγή σελίδας	\f	μονό εισαγωγικό	\’
νέα γραμμή	\n	διπλό εισαγωγικό	\”
επαναφορά κεφαλής	\r	οκταδικός αριθμός	\ooo
οριζόντιος στηλοθέτης	\t	16-δικος αριθμός	\xhhh
κατακόρυφος στηλοθέτης	\v		

Παράδειγμα: **Να καταστρωθεί πρόγραμμα που να επιτελεί τα παρακάτω:**

- *Ζήτησε από τον χρήστη ένα χαρακτήρα*
- *Πάρε από τον χρήστη τον χαρακτήρα*
- *Τύπωσε τον χαρακτήρα και τον ASCII κωδικό του*
- *Βρες τον επόμενο χαρακτήρα*
- *Τύπωσέ τον μαζί με τον κωδικό του*

```
/**********************************************************************
```

Το πρόγραμμα διαβάζει ένα χαρακτήρα και τυπώνει τον χαρακτήρα και τον επόμενό του καθώς και τους ASCII κωδικούς τους.

```
**********************************************************************/
#include <stdio.h>

main()
{
  char ch,next_ch;
  printf( "Write a character:\t" );
  scanf( "%c",&ch );
  printf( "The ASCII code of char %c is %d\n", ch, ch );
  next_ch=ch+1;      /* βρίσκει τον επόμενο χαρακτήρα */
  printf( "The ASCII code of char %c is %d\n", next_ch, next_ch );
 }
```

Για να τυπωθεί ο χαρακτήρας ένα στηλοθέτη δεξιότερα

Ο τύπος του ακεραίου

Δήλωση: δηλώνεται με τη λέξη κλειδί *int* και χρησιμοποιείται για να παραστήσει ακέραιους αριθμούς, αρνητικούς ή θετικούς. Η περιοχή τιμών εξαρτάται από την αρχιτεκτονική του μηχανήματος. Για έναν υπολογιστή με λέξη 16 bits η περιοχή τιμών του τύπου *int* είναι από -32767 έως +32767.

Εάν πριν από τη λέξη *int* τοποθετηθεί ο προσδιοριστής *long* τότε οι ακέραιοι *long int* εξασφαλίζουν αποθηκευτικό χώρο 32 bits. Αντίστοιχα, ο προσδιοριστής *unsigned* χρησιμοποιείται πριν από τη λέξη *int* για να χαρακτηρίσει τη μεταβλητή χωρίς πρόσημο, η οποία λαμβάνει τιμές από 0 έως 65535 για λέξη 16 bits.

Σε περιβάλλοντα 32 bits, όπως τα WinXP, ..., Win 7, οι ακέραιοι αποθηκεύουν τιμές στο διάστημα από -2.147.483.648 έως +2.147.483.648.

Ένας ακέραιος *short int* είναι τουλάχιστον 16 bits και ο *int* είναι τουλάχιστον τόσο μεγάλος όσο ο *short int*.

Εκτύπωση: με χρήση της συνάρτησης *printf* και των προσδιοριστών *%d, %o, %x* για την εμφάνιση σε δεκαδική, οκταδική και δεκαεξαδική μορφή, αντίστοιχα. Οι προσδιοριστές *l* (long), *h* (short), και *u* (unsigned) τοποθετούνται πριν από τους *d, o, x*.

```
printf( “dec=%d,  octal=%o,  hex=%x”, num,num,num );
```

Εισαγωγή: με χρήση της συνάρτησης *scanf* και του προσδιοριστή (specifier) *%d*. Η πρόταση

```
scanf( “%d”, &num );
```

διαβάζει από την κύρια είσοδο (πληκτρολόγιο) σε δεκαδική μορφή και αποδίδει την τιμή στην ακέραια μεταβλητή *num*.

Ακέραια σταθερά: Όταν στον πηγαίο κώδικα γράφουμε έναν αριθμό χωρίς δεκαδικό ή εκθετικό μέρος, ο compiler τον χειρίζεται ως ακέραια σταθερά. Η σταθερά 245 αποθηκεύεται ως *int*, ενώ η σταθερά 100.000 αποθηκεύεται ως *long int*. Εάν ορίσουμε τη σταθερά 8965 ως 8965L, ο compiler δεσμεύει χώρο για *long int*.

Παρατήρηση: Υπάρχει η δυνατότητα να καθορισθεί ο αριθμός των ψηφίων που θα εκτυπωθούν, τοποθετώντας τον επιθυμητό αριθμό ανάμεσα στο *%* και το *d*. Εάν ο αριθμός είναι μικρότερος από τον απαιτούμενο αριθμό ψηφίων του ακέραιου, η επιλογή δε θα ληφθεί υπόψη. Στην αντίθετη περίπτωση, στις πλεονάζουσες θέσεις θα τοποθετηθούν κενά.

Ο αριθμός που τοποθετείται στον προσδιοριστή *%d* ονομάζεται καθοριστικό ελάχιστου πλάτους πεδίου. Με αυτόν τον τρόπο, σε διαδοχικές *printf()* θα υπάρξει ευθυγράμμιση των αποτελεσμάτων κατά στήλες. Οι προτάσεις

```
printf( "dec=%1d,  octal=%4o,  hex=%4x", num,num,num );
printf( "dec=%4d,  octal=%4o,  hex=%4x", num,num,num );
```

θα τυπώσουν αντίστοιχα:

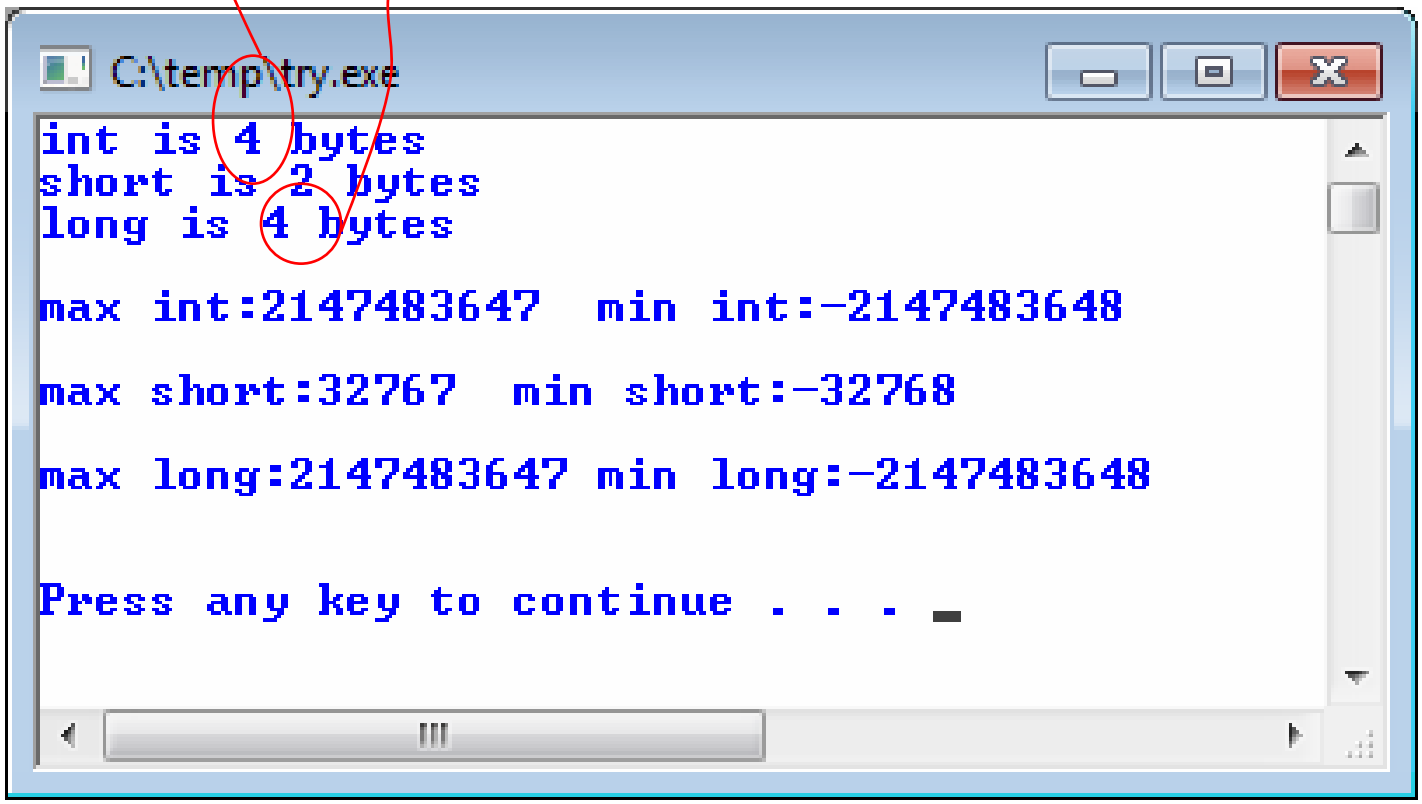
```
dec=46,  octal= 56,  hex= 2e
dec= 46,  octal= 56,  hex= 2e
```

```
/**********************************************************
Το πρόγραμμα εξετάζει το μήκος του τύπου ακεραίου.
**********************************************************/
#include <stdio.h>      // για την printf
#include <limits.h>     // climits.h αλλού

main() {
  int number_int=INT_MAX;  // Μέγιστος ακέραιος, οριζόμενος στο climits.h
  short int number_short=SHRT_MAX; // Μέγιστος short int
  long int number_long=LONG_MAX; // Μέγιστος long integer
// ο τελεστής sizeof δίνει το μέγεθος ενός τύπου δεδομένου ή μίας μεταβλητής
  printf( "int is %d bytes\n",sizeof(int) );
  printf( "short is %d bytes\n",sizeof(short) );
  printf( "long is %d bytes\n",sizeof(long) );
  printf( "\nmax int:%d  min int:%d\n",number_int,INT_MIN );
  printf( "\nmax short:%d  min short:%d\n",SHRT_MAX,SHRT_MIN );
  printf( "\nmax long:%d min long:%d\n",number_long,LONG_MIN );
}  // τέλος της main
```

Αποτελέσματα:

Ταυτίζονται τα bytes για int και long γιατί στην έκδοση 5 της C++ και με λειτουργικά WIN95 και μεταγενέστερα ο int καταλαμβάνει 4 bytes.



Τύποι πραγματικών αριθμών

Δήλωση: πραγματικοί είναι οι αριθμοί που διαθέτουν κλασματικό μέρος και εκφράζονται συνήθως στις ακόλουθες μορφές:

Fixed-point number	*Scientific notation*	*Exponential notation*
123.456	1.23456×10^{2}	1.23456e+02
0.00002	2.0×10^{-5}	2.0e-5
50000.0	5.0×10^{4}	5.0e+04

Η γλώσσα C διαθέτει δύο τύπους για αναπαράσταση πραγματικών αριθμών. Τον τύπο *float* για αριθμούς κινητής υποδιαστολής απλής ακρίβειας και τον τύπο *double* για αριθμούς κινητής υποδιαστολής διπλής ακρίβειας. Η πρόταση `float plank=6.63e-34;` δηλώνει τη μεταβλητή *plank* ως απλής ακρίβειας και της δίνει την τιμή 6.63e-34.

Η χρήση του προσδιοριστή *long* πριν από τον τύπο *double* χρησιμοποιείται για δήλωση μεταβλητής κινητής υποδιαστολής εκτεταμένης ακρίβειας, π.χ. `long double plank;`

Εκτύπωση: Με χρήση της *printf()* και των προσδιοριστών *%f* για εμφάνιση σε fixed point μορφή, *%e* για εμφάνιση σε εκθετική μορφή, και *%g* για να ανατεθεί στο σύστημα να επιλέξει μεταξύ των δύο προηγουμένων, με προτεραιότητα στη μορφή με το μικρότερο μέγεθος.

Αποθήκευση: Ως συνηθισμένα μεγέθη αναφέρονται για τους μεν *float* τα 32 bits, 8 για εκθέτη και πρόσημο και 24 για τη βάση, για τους δε *double* τα 64 bits, με τα επιπλέον 32 να χρησιμοποιούνται για αύξηση της ακρίβειας της βάσης (3 χρησιμοποιούνται στον εκθέτη).

Πραγματικές σταθερές: Πραγματικοί αριθμοί όπως οι

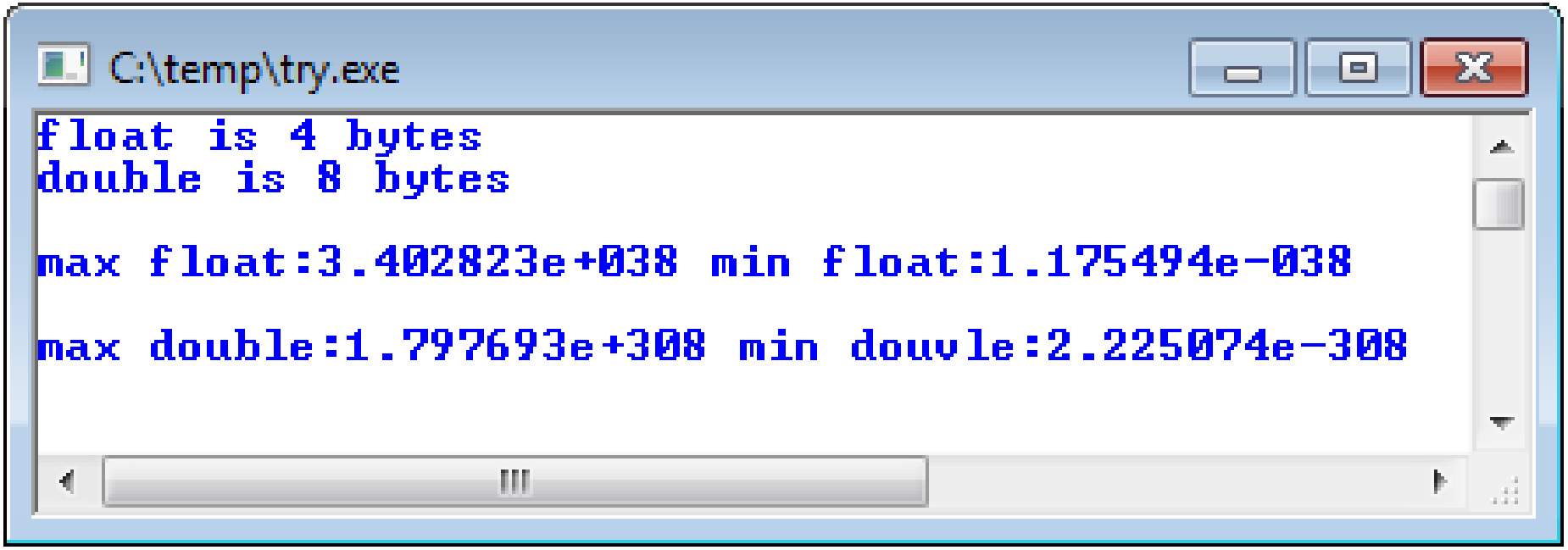
0.12 45.68 9e-5 24e09 0.0034e-08

όταν εμφανίζονται στον πηγαίο κώδικα αποτελούν τις πραγματικές σταθερές. Θεωρούνται από τον μεταγλωττιστή ως *double* και δεσμεύουν τον αντίστοιχο χώρο.

```
/**********************************************************************
        Το πρόγραμμα εξετάζει το μήκος του τύπου κινητής υποδιαστολής.
**********************************************************************/
#include <stdio.h>
#include <float.h>      // για το μέγιστο και τον ελάχιστο float, double
main()
{
  float num_float=FLT_MAX;    // Μέγιστος float
  double num_double=DBL_MAX; // Μέγιστος double

  printf("float is %d bytes\n",sizeof(float));
  printf("double is %d bytes\n",sizeof(double));
  printf("\nmax float:%e min float:%e\n",num_float,FLT_MIN);
  printf("\nmax double:%e,min double:%e\n" num_double,DBL_MIN);
}
```

Αποτελέσματα:



I/O κονσόλας

Η I/O κονσόλας αναφέρεται στις λειτουργίες που γίνονται στο πληκτρολόγιο και στην οθόνη του υπολογιστή. Εκτός από τις *printf()* και *scanf()* που χρησιμοποιήθηκαν προγουμένως (και αποτελούν τη φορμαρισμένη I/O κονσόλας γιατί μπορούν να διαβάζουν δεδομένα σε διάφορες φόρμες), υπάρχει μία σειρά απλούστερων συναρτήσεων που αναπτύσσεται ακολούθως.

Οι συναρτήσεις getche, getch

- **Η συνάρτηση *getche()* διαβάζει ένα χαρακτήρα από την κύρια είσοδο (πληκτρολόγιο). Αναμένει έως ότου πατηθεί ένα πλήκτρο και στη συνέχεια επιστρέφει την τιμή του, εμφανίζοντας στην οθόνη το πλήκτρο που πατήθηκε.**

- **Το πρωτότυπο της *getche()* είναι το ακόλουθο:**

```
int getche(void);
```

- **Η *getche()* επιστρέφει μεν έναν ακέραιο αλλά το byte χαμηλής τάξης περιέχει τον χαρακτήρα. Η χρήση ακεραίων γίνεται για λόγους συμβατότητας με τον αρχικό μεταγλωττιστή της UNIX C.**

Οι συναρτήσεις getche, getch (συνέχεια)

- **Στην Dev C++ το αρχείο κεφαλίδας της συνάρτησης *getche()* βρίσκεται στο *stdio.h*.**

- **Η συνάρτηση *getch()* αποτελεί παραλλαγή της *getche()* και βρίσκεται στο *stdio.h*. Λειτουργεί όπως ακριβώς η getche με τη διαφορά ότι η *getch()* δεν εμφανίζει τον πληκτρολογηθέντα χαρακτήρα στην οθόνη.**

- **Το πρωτότυπο της *getch()* είναι το ακόλουθο:**

```
int getch(void);
```

Η συνάρτηση getchar

- **Η συνάρτηση *getchar()* διαβάζει ένα χαρακτήρα από την κύρια είσοδο και τον επιστρέφει στο πρόγραμμα. Αποτελεί παραλλαγή της *getche()*. Είναι η αρχική συνάρτηση εισόδου χαρακτήρων που βασίζεται στο UNIX. Το πρόβλημα με τη συνάρτηση αυτή είναι ότι κρατά την είσοδο στην περιοχή προσωρινής αποθήκευσης μέχρι να δώσουμε επαναφορά κεφαλής. Έτσι, μετά την επιστροφή της *getchar()* περιμένουν ένας ή περισσότεροι χαρακτήρες στην ουρά εισόδου.**

- **Το πρωτότυπο της *getchar()* είναι το ακόλουθο:**

```
int getchar(void);
```

- **Το αρχείο κεφαλίδας της συνάρτησης *getchar()* βρίσκεται στο *stdio.h*.**

Η συνάρτηση putchar

- **Η συνάρτηση *putchar()* εμφανίζει στην οθόνη τον χαρακτήρα που έχει ως όρισμα (π.χ. *c*), στην τρέχουσα θέση του δρομέα.**
- **Το πρωτότυπο της *putchar()* είναι το ακόλουθο:**

```
int putchar(int c);
```

- **Η *putchar()* επιστρέφει μεν έναν ακέραιο αλλά το byte χαμηλής τάξης περιέχει τον χαρακτήρα. Η χρήση ακεραίων γίνεται για λόγους συμβατότητας με τον αρχικό μεταγλωττιστή της UNIX C.**
- **Το αρχείο κεφαλίδας της συνάρτησης *putchar()* βρίσκεται στο *stdio.h*.**

Η συνάρτηση kbhit

- **Η συνάρτηση *kbhit()* (keyboard hit) ελέγχει κατά πόσον ο χρήστης έχει πατήσει κάποιο πλήκτρο. Εφόσον έχει πατήσει κάποιο πλήκτρο η συνάρτηση επιστρέφει ως αληθής, σε αντίθετη περίπτωση επιστρέφει ως ψευδής. Η συνάρτηση *kbhit()* χρησιμοποιείται κυρίως για να διακόπτει ο χρήστης το πρόγραμμα κατά το δοκούν.**

- **Το πρωτότυπο της *kbhit()* είναι το ακόλουθο:**

```
int kbhit(void);
```

- **Στη Dev C++ το αρχείο κεφαλίδας της συνάρτησης *kbhit()* βρίσκεται στο *stdio.h*.**

Παράδειγμα: Το ακόλουθο πρόγραμμα παίρνει χαρακτήρες από το πληκτρολόγιο και μετατρέπει τα κεφαλαία σε μικρά και τούμπαλιν. Το πρόγραμμα σταματά μόλις πληκτρολογηθεί μία τελεία. Το αρχείο-κεφαλίδα ***ctype.h*** απαιτείται για τη συνάρτηση ***islower()***, που αληθεύει αν το όρισμά της είναι σε μικρά γράμματα, και τις συναρτήσεις ***toupper()***, ***tolower()***, που μετασχηματίζουν τα γράμματα.

(να μη γίνει ιδιαίτερη μνεία για την επαναληπτική πρόταση και την υπό συνθήκη διακλάδωση. Θα μελετηθούν εκτενώς αργότερα.)

```
#include <stdio.h>
#include <ctype.h>
main() {
  char ch;
  printf( "Start writing letters without ENTER\n\n");
  do {
      ch=getche();      printf( "  ->  " );
      if (islower(ch)) putchar(toupper(ch));
      else putchar(tolower(ch));     printf( "\n" );
  }  while (ch!='.');        // τέλος του βρόχου με το χαρακτήρα '.'
}
```

Αποτελέσματα: