Προγραμματισμός I (ECE115)

#4 κυριολεκτικά & μετατροπή τύπων

Χρήστος Αντωνόπουλος
cda@uth.gr

Κυριολεκτικά (literals)

- Συχνά θέλουμε να **αρχικοποιήσουμε** μεταβλητές του προγράμματος με μια **συγκεκριμένη** τιμή
  - υπάρχει επίσης η περίπτωση να επιθυμούμε να χρησιμοποιήσουμε μια ειδική συγκεκριμένη τιμή σε εκφράσεις αποτίμησης, π.χ. σταθερές όπως το π ή το e
- Μια έκφραση που ορίζει μια τιμή χωρίς αναφορά σε κάποια μεταβλητή ονομάζεται **κυριολεκτικό**
  - για κάθε τύπο δεδομένων ορίζονται διαφορετικές μορφές προσδιορισμού των αντίστοιχων κυριολεκτικών
- Η τιμή των κυριολεκτικών, καθώς και εκφράσεων που χρησιμοποιούν αποκλειστικά και μόνο κυριολεκτικά, είναι ήδη γνωστή πριν την εκτέλεση του κώδικα

Κυριολεκτικά `int`

- Έκφραση που αρχίζει με ένα δεκαδικό ψηφίο που δεν είναι `0`: ερμηνεύεται με το **δεκαδικό** σύστημα
- Αρχίζει με `0x`: ερμηνεύεται με το **δεκαεξαδικό** σύστημα
- Αρχίζει με `0`: ερμηνεύεται με το **οκταδικό** σύστημα
  - `97` ακέραια τιμή `97`
  - `0x61` ακέραια τιμή `97`
  - `0141` ακέραια τιμή `97`

- Με κατάληξη `l` ή `L` ερμηνεύονται ως `long int`
  - κατάληξη `u` ή `U`: ερμηνεύονται ως `unsigned int`
  - κατάληξη `ul` ή `UL`: ερμηνεύονται ως `unsigned long int`
  - διαφορετικά ερμηνεύονται ως `int`

Κυριολεκτικά `double`

- Η έκφραση δίνεται σε **δεκαδικό** σύστημα
- Προαιρετικά, σε «επιστημονική μορφή» με (πιθανά αρνητικό) εκθέτη μετά το `e` ή `E`
  - `3.14`
  - `-1.0`
  - `.1e2`
  - `1E-1`

- Εκφράσεις με κατάληξη `f` ή `F` ερμηνεύονται ως `float`
  - με κατάληξη `l` ή `L` ερμηνεύονται ως `long double`
  - διαφορετικά ως `double`

Κυριολεκτικά `char`

- Η έκφραση της μορφής '<c>' συμβολίζει τον αντίστοιχο εκτυπώσιμο χαρακτήρα ASCII <c>
- Η έκφραση '\<c>' συμβολίζει τον αντίστοιχο ειδικό μη εκτυπώσιμο χαρακτήρα ASCII (βλέπε manual)
- Η έκφραση '\x<d1d2>' συμβολίζει τον χαρακτήρα ASCII με τον αντίστοιχο **δεκαεξαδικό** κωδικό d1d2
  - τα d1 και d2 είναι αριθμητικά ψηφία
- Η έκφραση '\<d1d2d3>' συμβολίζει τον χαρακτήρα ASCII με τον αντίστοιχο **οκταδικό** κωδικό d1d2d3
  - τα d1, d2 και d3 είναι αριθμητικά ψηφία

Π.χ.:

- `'a','\x61','\141'` χαρακτήρας a
- `'\n','\x0A','\012'` newline / linefeed

Ο πίνακας κωδικοποίησης ASCII

δεύτερο δεκαεξαδικό ψηφίο

πρώτο δεκαεξαδικό ψηφίο	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	TAB	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL

- Οι αλφαβητικοί χαρακτήρες `'a','b','c',...` και `'A','B','C',...` έχουν **διαδοχικές** τιμές, αν ερμηνευτούν ως ακέραιοι
- Το ίδιο και οι χαρακτήρες `'0','1','2',...` των δεκαδικών ψηφίων

Αυτόματη μετατροπή/προαγωγή τύπων

- Γίνονται **αυτόματα** σε αποτιμήσεις και αναθέσεις
  `char` -> `int`, `short` -> `int`, `float` -> `double`
- **Προσοχή** στην μετατροπή `char` -> `int`
  - γίνεται **sign extension** αν το `char` είναι `signed`

- Αν ένα από τα ορίσματα μιας πράξης είναι `long double`, `double`, `float`, `long`, `unsigned`, `int` τότε και το άλλο όρισμα **«προάγεται»** αντίστοιχα

- Όταν μια «μεγάλη» τιμή ανατίθεται σε «μικρότερη» μεταβλητή τότε **χάνεται** μέρος των δεδομένων
  - μπορεί να χαθεί/διαστρεβλωθεί η πληροφορία

```
char c_s='\xff'; unsigned char c_u='\xff';
short i_s; unsigned short i_u;

i_s = i_u = c_s;  /* i_s,i_u γίνεται -1,65535(=2^16-1)
i_s = i_u = c_u;  /* i_s,i_u γίνεται 255,255 */
```

```
double d1=10.0,d2; int i1=3,i2=10;

d2 = d1 / i1;     /* d2 γίνεται 3.33… */
d2 = i2 / i1;     /* d2 είναι 3.0 */
d2 = i2 / d2;     /* d2 είναι 3.33… */
```

```
int i; char c;

i = 256;   /* i γίνεται 256 */
c = i;     /* c γίνεται 0 */
i = c;     /* i γίνεται 0 */
```

```
double d=3.3333;
int i;
char c;

i = 100*d;    /* c γίνεται 0x014D or 333 */
c = 100*d;    /* c γίνεται 0x4D or 'M' */
```

Αριθμητική με χαρακτήρες

- Η πιο χαρακτηριστική «εφαρμογή» της αυτόματης μετατροπής τύπων: **αριθμητική με χαρακτήρες**

- Μπορούμε να συνδυάσουμε ένα χαρακτήρα με ένα ακέραιο ή ένα χαρακτήρα με ένα χαρακτήρα

```
'a' + 1          /* 98,0x62,'b' */
'b' - 'a'          /* 1 */
'5' - '3' + '0'   /* 50,0x32,'2' */
```

- Μια συνάρτηση που δέχεται παράμετρο ένα ακέραιο, μπορεί να δεχτεί σαν παράμετρο έναν χαρακτήρα

```
putchar(97);      /* 'a' */
putchar('a'+2);     /* 'c' */
```

```c
/* ανάγνωση δύο χαρακτήρων, υπολογισμός της
   διαφοράς τους και εκτύπωση της ως ακέραιος */

#include <stdio.h>

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

  printf("enter two chars: ");
  scanf(" %c %c", &c1, &c2);

  diff = c1-c2;

  printf("%c=%d %c=%d diff=%d\n", c1, c1, c2, c2, diff);

  return(0);
}
```

Ρητή μετατροπή τύπων

- Ο προγραμματιστής μπορεί να **εκβιάσει** μια συγκεκριμένη μετατροπή τύπου με **type casting**

```
double d2; int i1=3,i2=10;

d2 = (double)i2 / i1;     /* d2 γίνεται 3.33… */
```

- Το type casting μπορεί να χρησιμοποιηθεί (και) για να μετατραπούν μπανάνες σε πορτοκάλια

- Χρειάζεται ιδιαίτερη **προσοχή!**
  - ο προγραμματιστής πρέπει να **ξέρει** τι κάνει
  - απαραίτητο σε κάποιες περιπτώσεις, βλέπε αργότερα …