

Διάλεξη 2 - Μνήμη και Μεταβλητές

Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών

Εισαγωγή στον Προγραμματισμό

Θανάσης Αυγερινός

Ανακοινώσεις / Διευκρινίσεις

- Το εργαστήριο ξεκινάει αυτήν την εβδομάδα!
- Σήμερα δεν θα κρατήσω ώρες γραφείου

Ερωτήσεις:

- Δεν λειτουργεί η εγκατάσταση WSL - τι κάνω;
- Μπορεί κάποιος να εντοπίσει που βρίσκεται η IP address μου;

Την Προηγούμενη Φορά

- Αρχιτεκτονική Υπολογιστικών Συστημάτων
- Linux και Γραμμή Εντολών
- Βασικές Εντολές
- Μεταγλωττιστές
- Hello World

Σήμερα

- Μνήμη στους υπολογιστές
- Δηλώσεις μεταβλητών
- Αναπαράσταση μεταβλητών στην μνήμη
- Τύπωμα μεταβλητών
- Συνέχεια του Hello World που αφήσαμε την άλλη φορά
- Συναρτήσεις και Ακέραιοι (Μαθηματικά vs C)
- Pair Programming



Bits & Bytes

Bit (binary digit): η μικρότερη μονάδα πληροφορίας σε υπολογιστή - 0 ή 1.

1

0

Byte: ομάδα από bit που αποθηκεύονται σε ένα κελί μνήμης

- Συνήθως κάθε byte είναι 8-bit (octet / οκτάδα)

0	0	1	0	1	0	1	0
---	---	---	---	---	---	---	---

Πόσα διαφορετικά 8-bit bytes μπορούμε να έχουμε;

Η Μνήμη Οργανώνεται σε Bytes

Το μέγεθος της μνήμης μετράται σε Bytes:

- 1 KB (KiloByte) = 1.000 Bytes
- 1 MB (MegaByte) = 1.000.000 Bytes
- 1 GB (GigaByte) = 1.000.000.000 Bytes

Μνήμη με
N Bytes

Byte 0

Byte 1

Byte 2

...

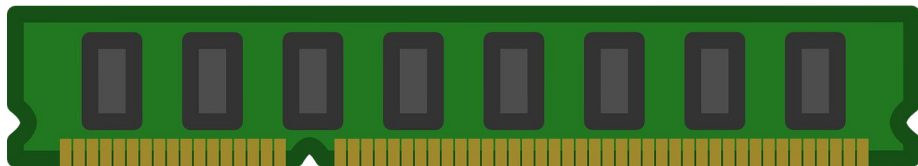
Byte N-1

Byte N

0	0	1	0	1	0	1	0
0	1	0	0	0	0	1	0
1	1	1	0	0	0	1	1

...

0	0	0	0	0	0	0	0
0	1	1	0	1	0	0	0



Διεύθυνση ενός Κελιού Μνήμης

Η θέση ενός κελιού στην μνήμη λέγεται

διεύθυνση (address).

πχ: στην διεύθυνση 2 υπάρχει το byte $11100011_{(2)}$

Μνήμη με
N Bytes

Byte 0

Byte 1

Byte 2

...

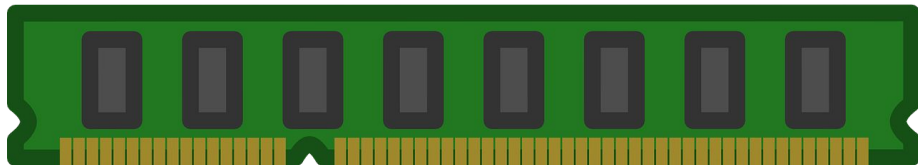
Byte N-1

Byte N

0	0	1	0	1	0	1	0
0	1	0	0	0	0	1	0
1	1	1	0	0	0	1	1

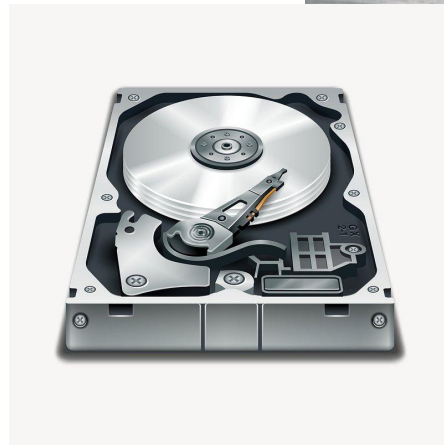
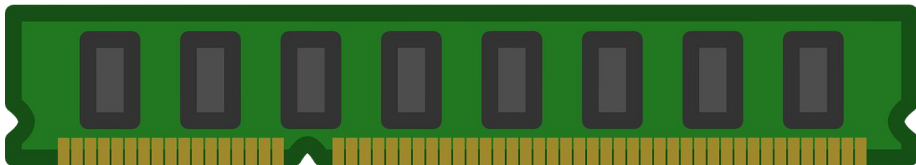
...

0	0	0	0	0	0	0	0
0	1	1	0	1	0	0	0



Κύρια και Δευτερεύουσα Μνήμη

- Η **δευτερεύουσα μνήμη** (**secondary memory**) αποθηκεύει δεδομένα μόνιμα, δηλαδή σε υλικό που τα διατηρεί ακόμα και χωρίς παροχή ρεύματος (σκληροί δίσκοι, flash drives, κτλ)
- Η **κύρια μνήμη** (**primary memory**) αποθηκεύει δεδομένα προσωρινά αλλά επιτρέπει στον επεξεργαστή την άμεση πρόσβασή και επεξεργασία τους



Γενικά όποτε αναφερόμαστε σε "μνήμη" εννοούμε την κύρια μνήμη εκτός αν διευκρινίσουμε περαιτέρω!

Δυαδικοί Αριθμοί

Ένας **δυαδικός αριθμός** εκφράζεται με μια ακολουθία από δύο σύμβολα: το 0 και το 1. Ισοδύναμα, λέμε ότι ο αριθμός εκφράζεται με βάση το 2 / εκφράζεται στο δυαδικό σύστημα / έχει δυαδική αναπαράσταση.

Για παράδειγμα, έστω ο δυαδικός αριθμός: $101010_{(2)}$

Στο δεκαδικό σύστημα ο ίδιος αριθμός γράφεται:

$$1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 42_{(10)}$$

Αντιστρέφοντας την διαδικασία μπορούμε να βρούμε την δυαδική αναπαράσταση οποιουδήποτε δεκαδικού αριθμού.

Δεκαεξαδικοί Αριθμοί και άλλες Βάσεις

Ένας δεκαεξαδικός αριθμός εκφράζεται με μια ακολουθία συνδυασμών από 16 σύμβολα:

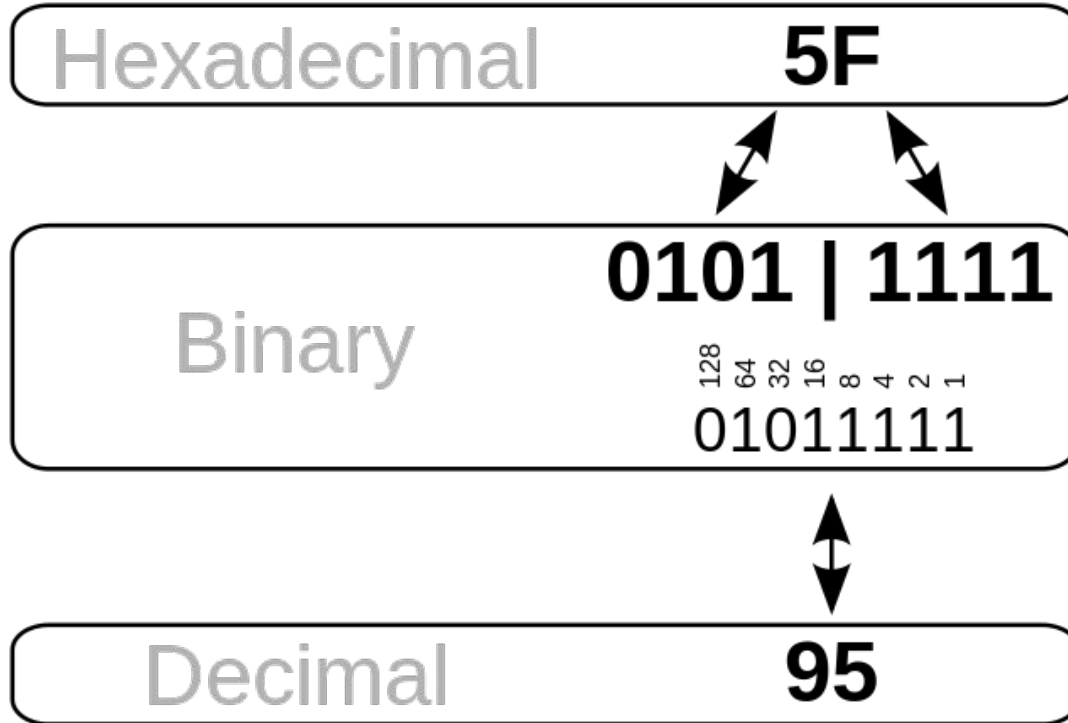
0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

Η μετατροπή ανάμεσα σε συστήματα γίνεται με τον ίδιο τρόπο:

$$2A_{(16)} = 2 \times 16^1 + A \times 16^0 = 42_{(10)}$$

Πόσα ψηφία χρειαζόμαστε στο δεκαεξαδικό σύστημα για να εκφράσουμε ένα byte; Είναι το ίδιο με το δεκαδικό;

Το ίδιο Byte - Διαφορετικά ψηφία σε κάθε βάση



Μεταβλητές και Χρήση Μνήμης

Δήλωση Μεταβλητής (Variable Declaration)

Μεταβλητή είναι ένα τμήμα της μνήμης με συγκεκριμένο **όνομα**.

Η μεταβλητή για να χρησιμοποιηθεί πρέπει να έχει **δηλωθεί** με κάποιον **τύπο**. Μορφή:

τύπος όνομα ;

Ο **τύπος (type)** της μεταβλητής λέει στον μεταγλωττιστή πόση μνήμη να δεσμεύσει για τα περιεχόμενα

Το **όνομα (name)** της μεταβλητής κάνει τον μεταγλωττιστή να διαλέξει την διεύθυνση της μνήμης που θα την αποθηκεύσει

Δήλωση Μεταβλητής (Variable Declaration)

Μεταβλητή είναι ένα τμήμα της μνήμης με συγκεκριμένο **όνομα**.

Η μεταβλητή για να χρησιμοποιηθεί πρέπει να έχει **δηλωθεί** με κάποιον **ΤΥΠΟ**.

int x;

Ο **τύπος (type)** της μεταβλητής λέει στον μεταγλωττιστή πόση μνήμη να δεσμεύσει για τα περιεχόμενα

Το **όνομα (name)** της μεταβλητής κάνει τον μεταγλωττιστή να διαλέξει την διεύθυνση της μνήμης που θα την αποθηκεύσει

π.χ. 4 bytes για την x ξεκινώντας από το 0

Byte 0

0	0	1	0	1	0	1	0
0	1	0	0	0	0	1	0
1	1	1	0	0	0	1	1
1	1	1	0	0	0	1	1

Byte 1

Byte 2

Byte 3

...

...

Byte N-1

Byte N

0	0	0	0	0	0	0	0
0	1	1	0	1	0	0	0

Δήλωση Μεταβλητής (Variable Declaration)

Μεταβλητή είναι ένα τμήμα της μνήμης με συγκεκριμένο **όνομα**.

Η μεταβλητή για να χρησιμοποιηθεί πρέπει να έχει **δηλωθεί** με κάποιον **ΤΥΠΟ**.

`char c;`

Ο **τύπος (type)** της μεταβλητής λέει στον μεταγλωττιστή πόση μνήμη να δεσμεύσει για τα περιεχόμενα

Το **όνομα (name)** της μεταβλητής κάνει τον μεταγλωττιστή να διαλέξει την διεύθυνση της μνήμης θα την αποθηκεύσει

π.χ. 1 byte για την c ξεκινώντας από το N-1

Byte 0	0	0	1	0	1	0	1	0
Byte 1	0	1	0	0	0	0	1	0
Byte 2	1	1	1	0	0	0	1	1
Byte 3	1	1	1	0	0	0	1	1
...	...							
Byte N-1	0	0	0	0	0	0	0	0
Byte N	0	1	1	0	1	0	0	0

Ερώτηση: ποια εντολή του Linux μας επιτρέπει να μάθουμε για
ένα πρόγραμμα στο σύστημά μας;

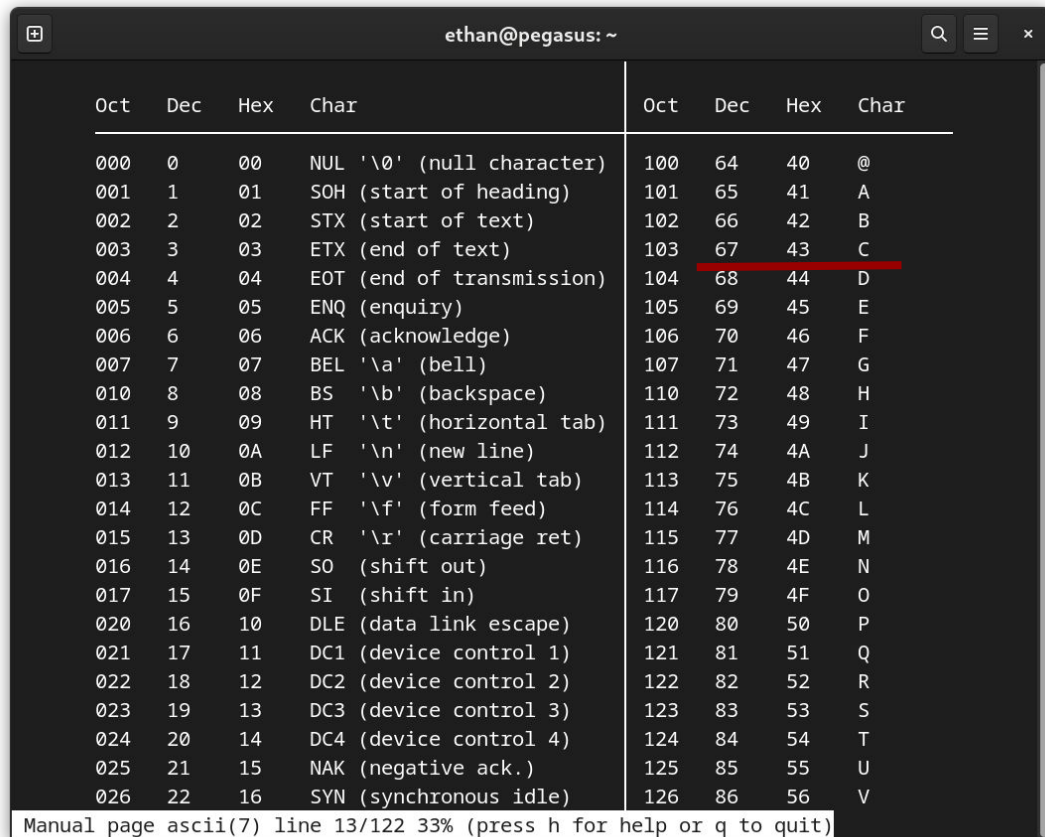
Ερμηνεία Δυαδικών Ψηφίων: Χαρακτήρες ASCII

Τα 8bits ενός char μπορεί να ερμηνευθούν ως χαρακτήρας κειμένου με την αντιστοίχιση του πίνακα [ASCII](#) (man ascii).

π.χ., ο αριθμός $67_{(10)} = 43_{(16)}$

αντιστοιχεί στο γράμμα 'C'

Πόσους διαφορετικούς χαρακτήρες έχουμε στο σύστημα ASCII; Γιατί;



Oct	Dec	Hex	Char	Oct	Dec	Hex	Char
000	0	00	NUL '\0' (null character)	100	64	40	@
001	1	01	SOH (start of heading)	101	65	41	A
002	2	02	STX (start of text)	102	66	42	B
003	3	03	ETX (end of text)	103	67	43	C
004	4	04	EOT (end of transmission)	104	68	44	D
005	5	05	ENQ (enquiry)	105	69	45	E
006	6	06	ACK (acknowledge)	106	70	46	F
007	7	07	BEL '\a' (bell)	107	71	47	G
010	8	08	BS '\b' (backspace)	110	72	48	H
011	9	09	HT '\t' (horizontal tab)	111	73	49	I
012	10	0A	LF '\n' (new line)	112	74	4A	J
013	11	0B	VT '\v' (vertical tab)	113	75	4B	K
014	12	0C	FF '\f' (form feed)	114	76	4C	L
015	13	0D	CR '\r' (carriage ret)	115	77	4D	M
016	14	0E	SO (shift out)	116	78	4E	N
017	15	0F	SI (shift in)	117	79	4F	O
020	16	10	DLE (data link escape)	120	80	50	P
021	17	11	DC1 (device control 1)	121	81	51	Q
022	18	12	DC2 (device control 2)	122	82	52	R
023	19	13	DC3 (device control 3)	123	83	53	S
024	20	14	DC4 (device control 4)	124	84	54	T
025	21	15	NAK (negative ack.)	125	85	55	U
026	22	16	SYN (synchronous idle)	126	86	56	V

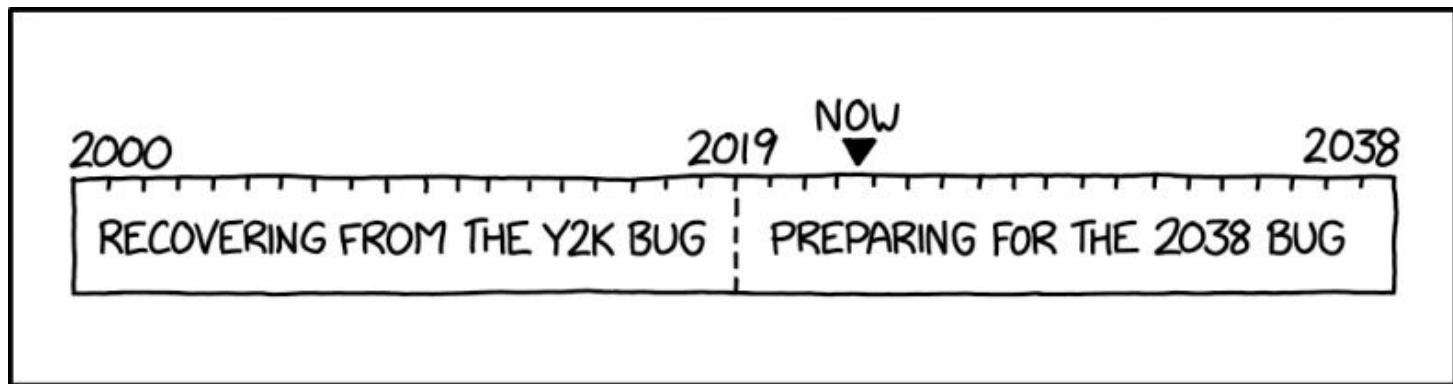
Manual page ascii(7) line 13/122 33% (press h for help or q to quit)

Προβλήματα Αναπαράστασης Ακεραίων

1. $2^{32} \sim 4$ δις ακέραιοι ίσως να μην αρκούν για το πρόγραμμά μας
 - Ευτυχώς η C προσφέρει τύπους με μεγαλύτερο μέγεθος (π.χ., 64bit)
2. Υπάρχουν θετικοί και αρνητικοί ακέραιοι. Πως αναπαριστούμε το γεγονός ότι ένας αριθμός είναι αρνητικός στην δυαδική αναπαράσταση; Τι κάνουμε με το πρόσημο;
 - Συμπλήρωμα του 2

Προβλήματα Αναπαράστασης Ακεραίων

1. $2^{32} \sim 4$ δις ακέραιοι ίσως να μην αρκούν για το πρόγραμμά μας
 - ο Ευτυχώς η C προσφέρει τύπους με μεγαλύτερο μέγεθος (π.χ., 64bit)



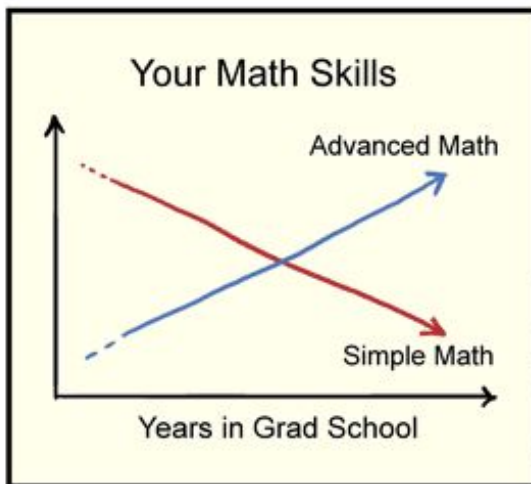
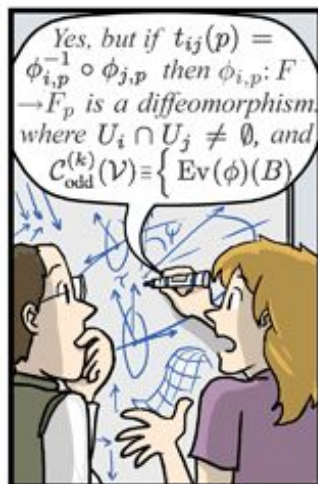
REMINDER: BY NOW YOU SHOULD HAVE FINISHED YOUR Y2K RECOVERY AND BE SEVERAL YEARS INTO 2038 PREPARATION.

Συμπλήρωμα του 2 (Two's complement)

Βασική ιδέα: να αναπαραστήσουμε το πρόσημο με το σημαντικότερο bit του αριθμού ($0 \rightarrow +$, $1 \rightarrow -$). Έστω ένας θετικός δυαδικός αριθμός N . Προκειμένου να πάρουμε την αναπαράσταση του $-N$, ακολουθούμε τα βήματα:

1. Αντίστρεψε (flip) όλα τα bit του N .
2. Πρόσθεσε 1.

Π.χ.: έστω $N = 11_{(10)} = 01011_{(2)}$, τότε το $-N$ είναι: $10100_{(2)} + 1_{(2)} = 10101_{(2)}$



WWW.PHDCOMICS.COM

Τύποι Μεταβλητών

Τύπος	Συνηθισμένο μέγεθος (bytes)	Εύρος τιμών (min-max)	Παράδειγμα Τιμής
char	1	-128 ... 127	'B', 0x42
short int	2	-32.768 ... 32.767	42
int	4	-2.147.483.648 ... 2.147.483.647	42
long int	4	-2.147.483.648 ... 2.147.483.647	42L
float	4	Μικρότερη θετική τιμή: $1.17 \cdot 10^{-38}$ Μεγαλύτερη θετική τιμή: $3.4 \cdot 10^{38}$	42.42F
double	8	Μικρότερη θετική τιμή: $2.2 \cdot 10^{-308}$ Μεγαλύτερη θετική τιμή: $1.8 \cdot 10^{308}$	42.42
long double	8, 10, 12, 16		42.42L
unsigned char	1	0 ... 255	'B', 0x42
unsigned short int	2	0 ... 65535	42
unsigned int	4	0 ... 4.294.967.295	42
unsigned long int	4	0 ... 4.294.967.295	42LU

Ανάθεση σε Μεταβλητή (Variable Assignment)

Ανάθεση σε μια μεταβλητή μπορεί να γίνει κατά τον ορισμό της:

```
int x = 42;
```

Ή μετά τον ορισμό της:

```
int x;
```

```
x = 42;
```

Ή με δεκαεξαδικό τρόπο:

```
int x = 0x2A;
```

Περιεχόμενο της x
πριν την ανάθεση

Byte 0

0	0	1	0	1	0	1	0
0	1	0	0	0	0	1	0
1	1	1	0	0	0	1	1
1	1	1	0	0	0	1	1

Byte 1

Byte 2

Byte 3

...

...

Byte N-1

Byte N

0	0	0	0	0	0	0	0
0	1	1	0	1	0	0	0

Ανάθεση σε Μεταβλητή (Variable Assignment)

Ανάθεση σε μια μεταβλητή μπορώ να γίνει κατά τον ορισμό της:

```
int x = 42;
```

Ή μετά τον ορισμό της:

```
int x;
```

```
x = 42;
```

Ή με δεκαεξαδικό τρόπο:

```
int x = 0x2A;
```

Γίνεται και σε οκταδικό: `x = 052;`

Περιεχόμενο της x
μετά την ανάθεση

Byte 0

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	1	0	1	0	1	0

Byte 1

Byte 2

Byte 3

...

...

Byte N-1

Byte N

0	0	0	0	0	0	0	0
0	1	1	0	1	0	0	0

Υπερχείλιση Ακεραίων (Integer Overflow)

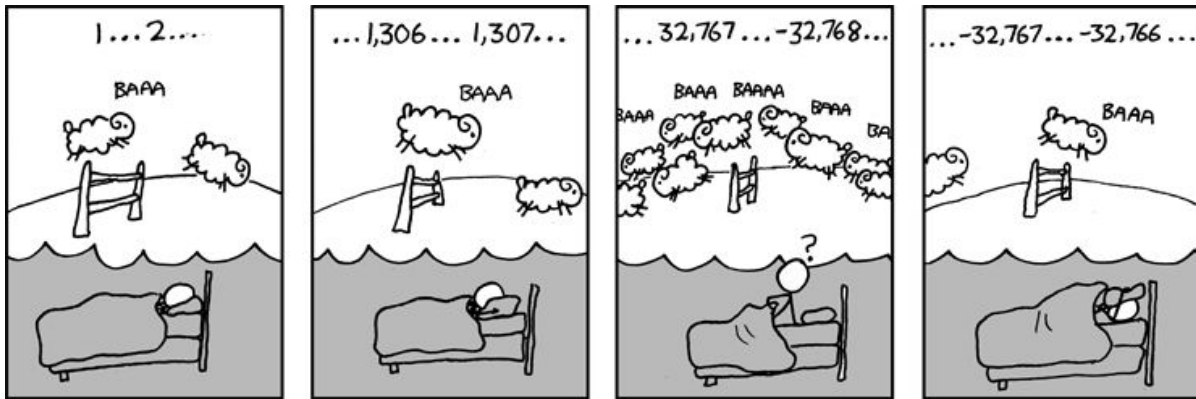
Τι θα τυπώσει το παρακάτω πρόγραμμα;

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

```
int main() {  
    printf("%d\n", 2000000000 + 2000000000);  
    return 0;  
}
```

```
$ ./overflow  
-294967296
```

Τι συνέβη; Πως το διορθώνουμε;



Ακέραιοι με συγκεκριμένη ακρίβεια

Είπαμε ότι το μέγεθος ενός `int` είναι συνήθως 4 bytes - υπάρχει τρόπος να είμαστε βέβαιοι;

- Με την χρήση της `sizeof` μπορούμε να βρούμε το μέγεθος ενός τύπου - περισσότερα σε μελλοντικές διαλέξεις
- Με την χρήση της βιβλιοθήκης `<stdint.h>` μπορούμε να δηλώσουμε ακεραίους με την επιθυμητή ακρίβεια:

`int8_t, uint8_t, int16_t, uint16_t, int32_t, uint32_t, int64_t, uint64_t`

Η Συνάρτηση printf

Η συνάρτηση printf() χρησιμοποιείται για το τύπωμα δεδομένων στο αρχείο εξόδου stdout (standard output)

- Έχει μία μεταβλητή λίστα παραμέτρων:
 - a. Η πρώτη παράμετρος είναι ένα αλφαριθμητικό μορφοποίησης (format string), δηλαδή μία ακολουθία χαρακτήρων μέσα σε διπλά εισαγωγικά (" ") η οποία καθορίζει τον τρόπο με τον οποίο θα τυπωθούν τα δεδομένα.
 - b. Οι επόμενες παράμετροι είναι προαιρετικές και, αν υπάρχουν, η printf() μπορεί να χρησιμοποιήσει τις τιμές τους
- Το αλφαριθμητικό μορφοποίησης (format string) μπορεί να περιέχει:
 - a. Απλούς χαρακτήρες (οι οποίοι εμφανίζονται όπως είναι στην οθόνη)
 - b. Ακολουθίες διαφυγής (Escape sequences)
 - c. Προσδιοριστικά μορφοποίησης (Format specifiers)

Ακολουθίες Διαφυγής (Escape Sequences)

Η ακολουθία διαφυγής αποτελείται από μία ανάστροφη κεκλιμένη (\) (backslash) και έναν χαρακτήρα

Escape Sequence	Σημασία
\n	Αλλαγή γραμμής, σαν το πλήκτρο Enter
\r	Επαναφορά του δρομέα (cursor) στην αρχή της τρέχουσας γραμμής
\t	Τύπωμα ενός κενού ίσο με το tab, σαν το πλήκτρο Tab
\\	Τύπωμα ανάστροφης κεκλιμένης (backslash)
\"	Τύπωμα διπλών εισαγωγικών (double quotes) (")
\xNN	Τύπωμα του χαρακτήρα που αντιστοιχεί σε NN σε δεκαεξαδικό
\a	Δημιουργία ηχητικού σήματος

Προσδιοριστικά Μορφοποίησης (Format Specifiers)

Τα προσδιοριστικά μορφοποίησης αποτελούνται από τον χαρακτήρα % ακολουθούμενο από έναν ή περισσότερους χαρακτήρες που προσδιορίζουν πως να γίνει η μορφοποίηση (πλήρης λίστα [εδώ](#))

Format Specifier	Σημασία
%c	Τύπων ASCII χαρακτήρα
%d ή %i	Τύπων ακεραίου
%u	Τύπων unsigned (μη-προσημασμένου) ακεραίου
%f	Τύπων float
%llu	Τύπων long long unsigned int
%%	Τύπων του χαρακτήρα %

Δηλώσεις Πολλών Μεταβλητών

Μεταβλητές του ίδιου τύπου μπορούν να δηλωθούν στην ίδια γραμμή, διαχωρισμένες με κόμμα (,):

```
int a;
```

```
int b;
```

```
int c;
```

Μπορεί να γραφτεί ισοδύναμα ως:

```
int a, b, c;
```

Δεσμευμένες Λέξεις (Reserved Keywords) στην C

Οι δεσμευμένες λέξεις έχουν προκαθορισμένο νόημα για τον μεταγλωττιστή και **δεν** επιτρέπεται να χρησιμοποιηθούν για τον ορισμό μεταβλητών ή συναρτήσεων.

<code>auto</code>	<code>do</code>	<code>goto</code>	<code>signed</code>	<code>unsigned</code>
<code>break</code>	<code>double</code>	<code>if</code>	<code>sizeof</code>	<code>void</code>
<code>case</code>	<code>else</code>	<code>int</code>	<code>static</code>	<code>volatile</code>
<code>char</code>	<code>enum</code>	<code>long</code>	<code>struct</code>	<code>while</code>
<code>const</code>	<code>extern</code>	<code>register</code>	<code>switch</code>	
<code>continue</code>	<code>for</code>	<code>return</code>	<code>typedef</code>	
<code>default</code>	<code>float</code>	<code>short</code>	<code>union</code>	

Kahoot Time!

Για την Επόμενη Φορά

- Από τις σημειώσεις του κ. Σταματόπουλου συνιστώ να έχετε καλύψει τα πάντα μέχρι την σελίδα 35 + σελίδες 58-71.
- [Bits and bytes](#)
- [Escape Sequences in C](#)
- Printf [tips](#) and [reference](#)
- Συμπλήρωμα ως προς 2 εξήγηση [γραπτή](#) και σε [βίντεο](#)

Ευχαριστώ και καλή εβδομάδα εύχομαι!
Keep (or Start!) coding ;)