



ΠΑΝΕΠΙΣΤΗΜΙΟ  
ΠΑΤΡΩΝ  
UNIVERSITY OF PATRAS

ΑΝΟΙΚΤΑ ακαδημαϊκά  
μαθήματα ΠΠ

# Εισαγωγή στους Υπολογιστές

Ενότητα 7: Ψηφιακή Αριθμητική

Βασίλης Παλιουράς

Πολυτεχνική Σχολή

Τμήμα Ηλεκτρολόγων Μηχανικών και Τεχνολογίας  
Υπολογιστών

# Σκοποί ενότητας

- Ψηφιακή Αριθμητική



# Περιεχόμενα ενότητας

- Τύποι δεδομένων στην ρυθση
- Αναπαράσταση ακεραίων και μετατροπές βάσεων
  - Γενικές περιπτώσεις
    - Εφαρμογή κανόνα αθροίσματος γινομένων ψηφίων με αντίστοιχα βάρη
    - Κανόνας πηλίκων
  - Ειδικές περιπτώσεις
    - Ομαδοποιήσεις ψηφίων και εύκολη απεικόνιση
      - Όταν η μια βάση είναι δύναμη της άλλης



# Ψηφιακή Αριθμητική

# Αναπαράσταση Δεδομένων

- Στον υπολογιστή ή σε οποιαδήποτε ψηφιακή συσκευή
- Πολλοί τρόποι αναπαράστασης γιατί
  - Υπάρχουν διαφορετικές ανάγκες
    - Λέξεις **‘Βιβλίο’**
    - Μετρήσεις φυσικών μεγεθών ή αριθμητικά δεδομένα σε τεχνικούς υπολογισμούς: **3.141516**
    - Μετρητές, απαρίθμηση πραγμάτων: **5** φορές
  - Οι πράξεις *καθορίζονται* από την αναπαράσταση
    - Ιδιότητες και υλοποίηση



# Τύποι και τελεστές

```
Python Shell
File Edit Shell Debug Options Windows Help
>>>
>>> a = 'Hi '
>>> b = 'there!'
>>> print a + b
Hi there!
>>> a = 3
>>> b = 4
>>> print a + b
7
>>> a = 3.2
>>> b = 4
>>> print a + b
7.2
>>>
>>>
```



# Πόσο κάνει $0.1 * 0.1$ ;

```
Python Shell
File Edit Shell Debug Options Windows Help
>>>
>>> 0.1 * 0.1 == 0.01
False
>>>
>>>
>>>
>>>
```



# Πόσο κάνει $0.1 * 0.1$ ;

```
Python Shell
File Edit Shell Debug Options Windows Help
>>>
>>>
>>>
>>>
>>>
>>>
>>> 0.1 * 0.1
0.010000000000000002
>>> 0.25*0.25
0.0625
>>> 0.0625 * 0.25
0.015625
>>> |
```

```
Python Shell
File Edit Shell Debug Options Windows Help
>>>
>>> a = 0.1 * 0.1
>>> print a
0.01
>>> a
0.010000000000000002
>>>
>>>
>>>
```





# Θεμελιώδεις τύποι δεδομένων

- Άμεση υποστήριξη από το hardware
  - Ακέραιοι (integers, **int**),
  - Κινητής υποδιαστολής (floating point, **float**),
  - Χαρακτήρες, αλφαριθμητικά (**str**)
- Από τους θεμελιώδεις, το προγραμματιστικό περιβάλλον «χτίζει» νέους τύπους δεδομένων

Τύποι στο software

Τύποι στο hardware



# Διαφορετικές γλώσσες - διαφορετική συμπεριφορά C *versus* Python

```
#include <stdio.h>
int main(int argc, char *argv[]) {
    int i = 1<<30;
    printf("%d\n", i);
    system("pause");
}
```

1073741824

```
>>> a = 1 <<30
```

```
>>> a
```

1073741824

Σημαίνει  $2^{30}$

```
#include <stdio.h>
int main(int argc, char *argv[]) {
    int i = 1<<31;
    printf("%d\n", i);
    system("pause");
}
```

-2147483648

```
>>> a = 1 <<31
```

```
>>> a
```

2147483648L



# Ένα πείραμα

```
global a
a = 1<<30
def test():
    for i in range(0, 50):
        b = a + 1
if __name__ == '__main__':
    from timeit import Timer
    t = Timer('test()', 'from __main__ import test')
    print t.timeit(1000000)
```

**Χρόνος: 10.7764198383 sec**

```
global a
a = 1<<31
def test():
    for i in range(0, 50):
        b = a + 1
if __name__ == '__main__':
    from timeit import Timer
    t = Timer('test()', 'from __main__ import test')
    print t.timeit(1000000)
```

**Χρόνος: 21.0824803152 sec**

**Διπλάσιος χρόνος!**



# Ένας μηχανισμός - *preview*

1

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

Για αριθμούς με πρόσημο σε αναπαράσταση συμπληρώματος του δύο, Αν όχι πρόσημο, τότε  $+2^7$

βάρη

$-2^7$	$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$
--------	-------	-------	-------	-------	-------	-------	-------

$64=2^6$

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

$1 \ll 6$

65

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

-128

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

$1 \ll 7$



Αν χρειαζόμαστε το +128 ;;;

-128

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

-2 <sup>15</sup>	2 <sup>14</sup>	2 <sup>13</sup>	2 <sup>12</sup>	2 <sup>11</sup>	2 <sup>10</sup>	2 <sup>9</sup>	2 <sup>8</sup>	2 <sup>7</sup>	2 <sup>6</sup>	2 <sup>5</sup>	2 <sup>4</sup>	2 <sup>3</sup>	2 <sup>2</sup>	2 <sup>1</sup>	2 <sup>0</sup>
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Χρειαζονται δύο λέξεις ⇒ περισσότερη δουλειά!



# Αν χρειαζόμαστε το +128 ;;;

-128

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

-2 <sup>15</sup>	2 <sup>14</sup>	2 <sup>13</sup>	2 <sup>12</sup>	2 <sup>11</sup>	2 <sup>10</sup>	2 <sup>9</sup>	2 <sup>8</sup>	2 <sup>7</sup>	2 <sup>6</sup>	2 <sup>5</sup>	2 <sup>4</sup>	2 <sup>3</sup>	2 <sup>2</sup>	2 <sup>1</sup>	2 <sup>0</sup>
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Χρειαζονται δύο λέξεις  $\Rightarrow$  περισσότερη δουλειά!



# Μετατροπές βάσης σε python

- `int ( x, [base])`
- `hex( )`
- `oct( )`
- `bin( )`



# Μετατροπές μεταξύ βάσεων

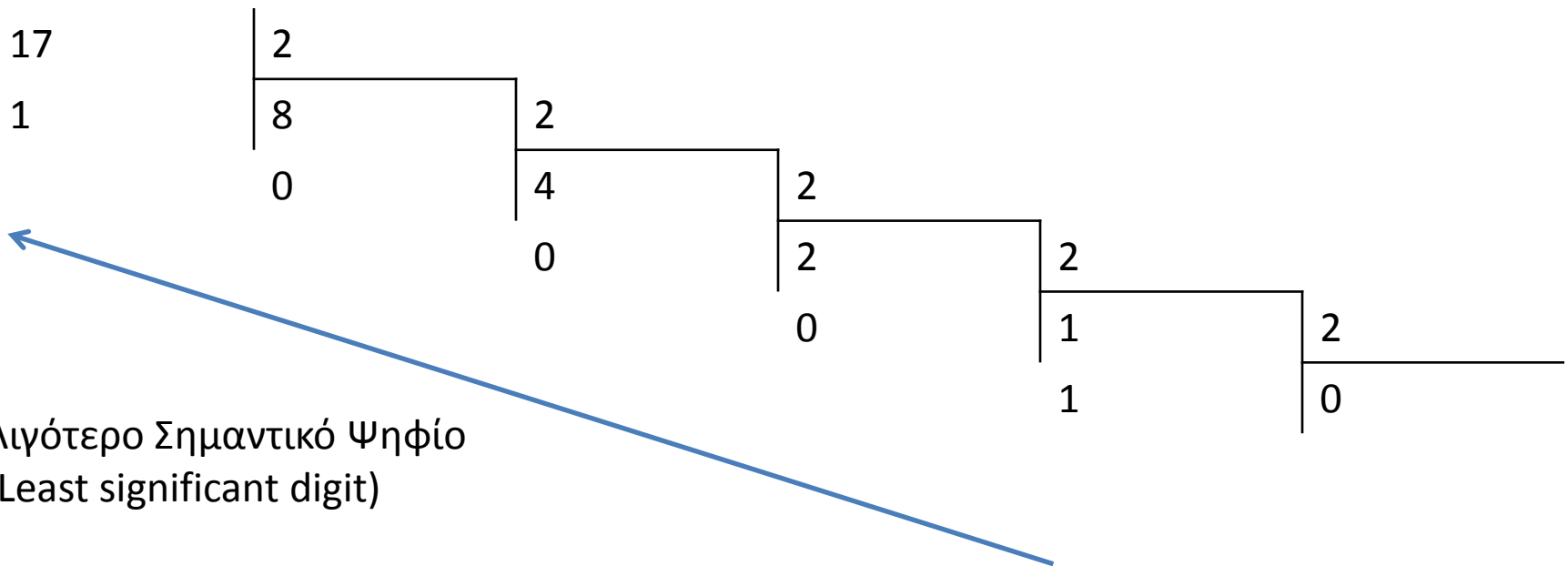
Μέθοδος διαδοχικών διαιρέσεων:

- **Νέο πηλίκο:** ακέραια διαίρεση του παλιού πηλίκου με **τη βάση**, μέχρις ότου να μηδενιστεί το πηλίκο.
- Η ζητούμενη αναπαράσταση σχηματίζεται από τα υπόλοιπα των διαιρέσεων.
  - Τα περισσότερο σημαντικά ψηφία υπολογίζονται τελευταία.





# Μετατροπή από βάση 10 σε βάση 2



$$\begin{aligned} 17 &= 8 \times 2 + 1 \\ &= (4 \times 2 + 0) \times 2 + 1 \\ &= ((2 \times 2 + 0) \times 2 + 0) \times 2 + 1 \\ &= (((1 \times 2 + 0) \times 2 + 0) \times 2 + 0) \times 2 + 1 \\ &= (((((0 \times 2 + 1) \times 2 + 0) \times 2 + 0) \times 2 + 0) \times 2 + 0) \times 2 + 1 \\ &= 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 \end{aligned}$$

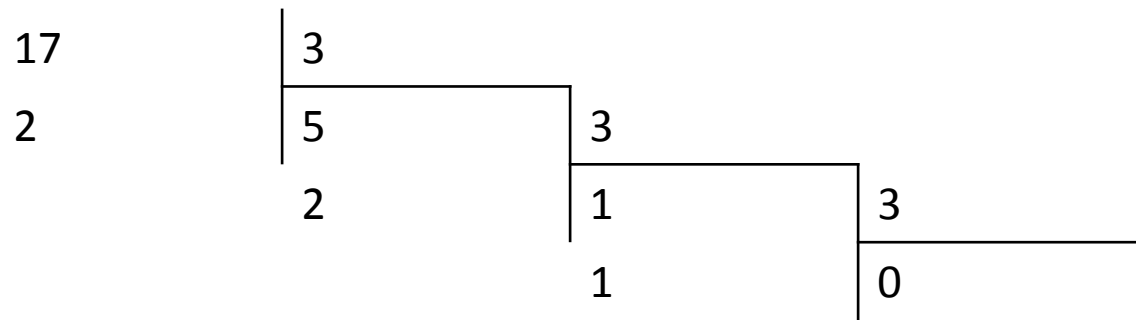


# Γιατί ισχύει η μετατροπή;

- $17 = 8 \times 2 + 1$   
 $= (4 \times 2 + 0) \times 2 + 1$   
 $= ((2 \times 2 + 0) \times 2 + 0) \times 2 + 1$   
 $= (((1 \times 2 + 0) \times 2 + 0) \times 2 + 0) \times 2 + 1$   
 $= (((((0 \times 2 + 1) \times 2 + 0) \times 2 + 0) \times 2 + 0) \times 2 + 0) \times 2 + 1$   
 $= 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$



# Ισχύει για μετατροπή σε κάθε βάση $r$



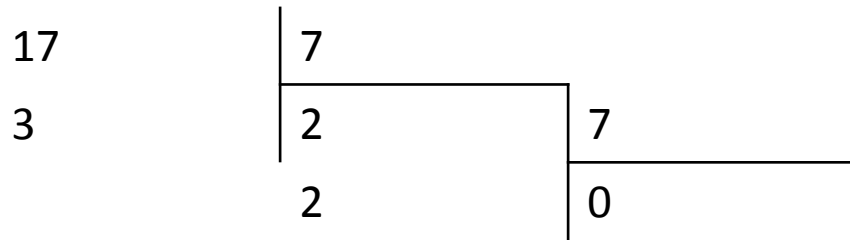
$$r = 3$$

$$17_{10} = 122_3$$

$$122_3 = 1 \times 3^2 + 2 \times 3^1 + 2 \times 3^0 = 9 + 6 + 2 = 17$$



# Βάση $r = 7$



$$r = 7$$

$$17_{10} = 23_7$$

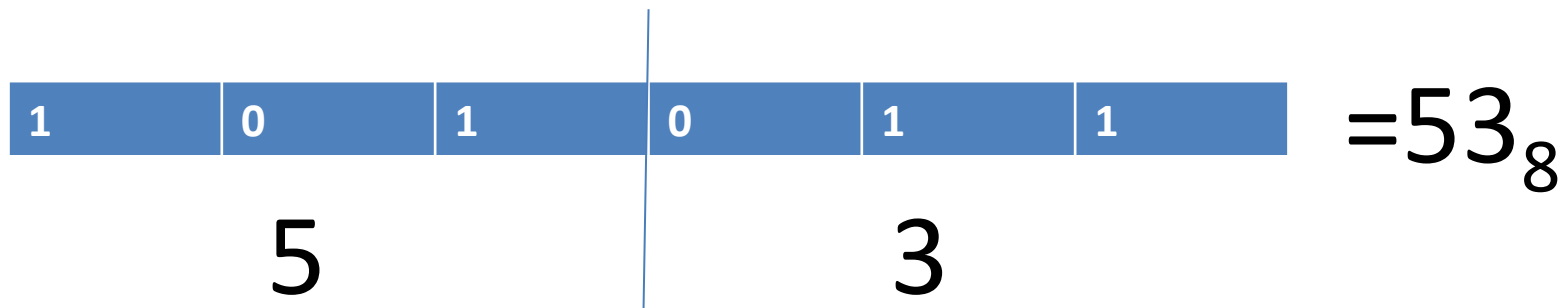
$$23_7 = 2 \times 7^1 + 3 \times 7^0$$

Πόσα ψηφία χρειάζομαι για να γράψω τον αριθμό  $X$  (θετικός ακέραιος) σε βάση  $r$ ;



# Χρήσιμες -ειδικές περιπτώσεις μετατροπής βάσεων

- Ιδιότητες απλουστεύουν τους υπολογισμούς
- Μετατροπές μεταξύ συγκεκριμένων βάσεων
  - Βάση 8  $\leftrightarrow$  Βάση 2  $\leftrightarrow$  Βάση 16
  - Ιδιότητα  $r_1, r_2$ , οι βάσεις (radix)



# Χρήσιμες -ειδικές περιπτώσεις μετατροπής βάσεων

- Παραδείγματα μετατροπών ακεραίων μεταξύ βάσεων
- Γενικές περιπτώσεις
  - Από βάση  $r$  σε βάση 10
  - Από βάση 10 σε βάση  $r$
- Ειδικές περιπτώσεις εύκολων μετατροπών



Τέλος Ενότητας

# Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στο πλαίσιο του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Αθηνών**» έχει χρηματοδοτήσει μόνο την αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.





Σημειώματα

# Σημείωμα Ιστορικού Εκδόσεων Έργου

Το παρόν έργο αποτελεί την έκδοση **1.0**.

Έχουν προηγηθεί οι κάτωθι εκδόσεις:

- Έκδοση **1.0** διαθέσιμη [εδώ](#).



# Σημείωμα Αναφοράς

Copyright Πανεπιστήμιο Πατρών, Αβούρης Νικόλαος, Παλιουράς Βασίλειος, Κουκιάς Μιχαήλ, Σγάρμπας Κυριάκος. «Εισαγωγή στους Υπολογιστές Ι, Ψηφιακή Αριθμητική». Έκδοση: 1.0. Πάτρα 2014. Διαθέσιμο από τη δικτυακή διεύθυνση:

[https://eclass.upatras.gr/modules/course\\_metadata/opencourses.php?fc=15](https://eclass.upatras.gr/modules/course_metadata/opencourses.php?fc=15)



# Σημείωμα Αδειοδότησης

Το παρόν υλικό διατίθεται με τους όρους της άδειας χρήσης Creative Commons Αναφορά, Μη Εμπορική Χρήση Παρόμοια Διανομή 4.0 [1] ή μεταγενέστερη, Διεθνής Έκδοση. Εξαιρούνται τα αυτοτελή έργα τρίτων π.χ. φωτογραφίες, διαγράμματα κ.λ.π., τα οποία εμπεριέχονται σε αυτό και τα οποία αναφέρονται μαζί με τους όρους χρήσης τους στο «Σημείωμα Χρήσης Έργων Τρίτων».



[1] <http://creativecommons.org/licenses/by-nc-sa/4.0/>

Ως **Μη Εμπορική** ορίζεται η χρήση:

- που δεν περιλαμβάνει άμεσο ή έμμεσο οικονομικό όφελος από την χρήση του έργου, για το διανομέα του έργου και αδειοδόχο
- που δεν περιλαμβάνει οικονομική συναλλαγή ως προϋπόθεση για τη χρήση ή πρόσβαση στο έργο
- που δεν προσπορίζει στο διανομέα του έργου και αδειοδόχο έμμεσο οικονομικό όφελος (π.χ. διαφημίσεις) από την προβολή του έργου σε διαδικτυακό τόπο

Ο δικαιούχος μπορεί να παρέχει στον αδειοδόχο ξεχωριστή άδεια να χρησιμοποιεί το έργο για εμπορική χρήση, εφόσον αυτό του ζητηθεί.

# Διατήρηση Σημειωμάτων

Οποιαδήποτε αναπαραγωγή ή διασκευή του υλικού θα πρέπει να συμπεριλαμβάνει:

- το Σημείωμα Αναφοράς
- το Σημείωμα Αδειοδότησης
- τη δήλωση Διατήρησης Σημειωμάτων
- το Σημείωμα Χρήσης Έργων Τρίτων (εφόσον υπάρχει)

μαζί με τους συνοδευόμενους υπερσυνδέσμους.



# Σημείωμα Χρήσης Έργων Τρίτων

Το Έργο αυτό κάνει χρήση των ακόλουθων έργων:

**Εικόνες/Σχήματα/Διαγράμματα/Φωτογραφίες**

**Διαφάνειες 6, 7, 8: χρήση του Python Editor IDLE**

