Πρόγραμμα Μεταπτυχιακών Σπουδών
Πληροφορική και Υπολογιστική Βιοϊατρική

Θέματα Προγραμματισμού Η/Υ

Ενότητα 5:

Θεματική Ενότητα: Μεταβλητές και Μαθηματικοί και λογικοί τελεστές

ΘΕΜΑΤΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ Η/Υ

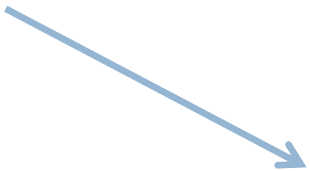
Θεματική Ενότητα 5
Μεταβλητές και Μαθηματικοί και λογικοί τελεστές

Πληροφορική και Υπολογιστική Βιοϊατρική
Α. Κακαρούντας, Γ. Σπαθούλας, Π. Κοντού

Παράδειγμα I/O (variable type)

```
# Είσοδος του ονόματος, της ηλικίας και του βάρους του χρήστη.
name = input('Πώς σε λένε; ')
age = input('Πόσο χρονών είσαι; ')
weight = input('Ποιο είναι το βάρος σου; ')

# Εμφάνιση των δεδομένων.
print('Τα δεδομένα που έδωσες:')
print('Όνομα:', name)
print('Ηλικία:', age)
print('Βάρος:', weight)
```



```
Πώς σε λένε; Bill [ENTER]
Πόσο χρονών είσαι; 25 [ENTER]
Ποιο είναι το βάρος σου; 82.3 [ENTER]
Τα δεδομένα που έδωσες:
Όνομα: Bill
Ηλικία: 25
Βάρος: 82.3
>>> type(age)    # ή type(weight)
<class 'str'>
>>>
```

Είσοδος Αριθμών με τη Συνάρτηση input (1)

- Η input εκχωρεί **μόνο** συμβολοσειρές σε μεταβλητές.
- Με χρήση άλλων συναρτήσεων οι συμβολοσειρές *μετατρέπονται* σε αριθμητικούς τύπους.
  - **int(item):** Η συνάρτηση int() δέχεται ένα όρισμα και επιστρέφει την τιμή του αφού τη μετατρέψει σε int.

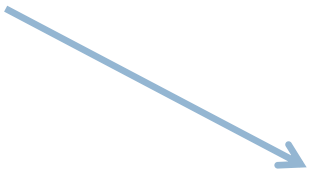
    π.χ.
    ```
    varos = input('Δώστε το βάρος σας ')
    varos = int(varos)
    ```
  - **float(item):** Η συνάρτηση float() δέχεται ένα όρισμα και επιστρέφει την τιμή του αφού τη μετατρέψει σε float.

    π.χ.
    ```
    mikos = float(input('Ποιο το μήκος του τοίχου? '))
    ```

Παράδειγμα I/O (με αριθμητικές τιμές)

```
# Είσοδος του ονόματος, της ηλικίας και του βάρους του χρήστη.
name = input('Πώς σε λένε; ')
age = int(input('Πόσο χρονών είσαι; '))
weight = float(input('Ποιο είναι το βάρος σου; '))

# Εμφάνιση των δεδομένων.
print('Τα δεδομένα που έδωσες:')
print('Όνομα:', name)
print('Ηλικία:', age)
print('Βάρος:', weight)
```



```
Πώς σε λένε; Bill [ENTER]
Πόσο χρονών είσαι; 25 [ENTER]
Ποιο είναι το βάρος σου; 82.3 [ENTER]
Τα δεδομένα που έδωσες:
Όνομα: Bill
Ηλικία: 25
Βάρος: 82.3
<class 'int'>     # print(type(age))
<class 'float'>   # print(type(weight))
>>>
```

Είσοδος Αριθμών με τη Συνάρτηση input (2)

- Οι συναρτήσεις int() και float() λειτουργούν μόνο αν τα ορίσματά τους είναι αποδεκτές αριθμητικές τιμές, δηλαδή είναι συμβολοσειρές αριθμητικών τιμών του τύπου στον οποίο ζητάμε να μετατραπούν.

```
>>> age = int(input('Πόσο χρονών είσαι; '))
Πόσο χρονών είσαι; 23.3
Traceback (most recent call last):
  File "<pyshell#2>", line 1, in <module>
    age = int(input('Πόσο χρονών είσαι; '))
ValueError: invalid literal for int() with base 10: '23.3'

>>> number = float(input('Δώσε έναν πραγματικό αριθμό: '))
Δώσε έναν πραγματικό αριθμό: test
Traceback (most recent call last):
  File "<pyshell#6>", line 1, in <module>
    number = float(input('Δώσε έναν πραγματικό αριθμό: '))
ValueError: could not convert string to float: 'test'
```

Μετατροπές Τύπων Δεδομένων

- Μετατροπή float σε int

```
float_value = 5.7
int_value = int(float_value)          # int_value = 5
                                      # Αποκοπή δεκαδικού μέρους

float_value = -3.8
int_value = int(float_value)          # int_value = -3
```

- Μετατροπή int σε float

```
int_value = 4
float_value = float(int_value)        # float_value = 4.0
```

Περισσότερα για την print (\, end)

- Για σύνταξη της print (και κάθε εντολής) σε περισσότερες από μια γραμμές, χωρίς όμως να εισάγει νέα γραμμή στην έξοδο, χρησιμοποιείται ο τελεστής " \ ":
  - print('Αγόρασε', num_of_oranges, 'πορτοκάλια', \
    'και πλήρωσε', cost)
- Κάθε print τυπώνει μία ή περισσότερες γραμμές αλλά εισάγει και μια νέα γραμμή στο τέλος της εξόδου, εκεί όπου τυπώνεται η επόμενη:
  - print('Ένα')
    print('Δύο')
    print('Τρία') # 3 γραμμές εξόδου
- Για να μην δημιουργεί η print νέα γραμμή μετά την εμφάνιση της εξόδου της, χρησιμοποιείται το ειδικό όρισμα end=' ' στη συνάρτηση:
  - print('Ένα', end=' ')
    print('Δύο', end=' ')
    print('Τρία') # τυπώνει: Ένα Δύο Τρία

    Κενός χαρακτήρας. Για εκτύπωση χωρίς κενό: end=

καθώς οι 2 πρώτες print θα εμφανίσουν ένα κενό αντί για το χαρακτήρα νέας γραμμής στο τέλος της εξόδου τους.

Περισσότερα για την print (sep)

- Όταν μεταβιβάζονται πολλαπλά ορίσματα στη συνάρτηση print, στην οθόνη διαχωρίζονται αυτόματα με ένα κενό:
  - >>> print('1', '2', '3')
    1 2 3
    >>>
- Για να μην εμφανιστεί κενό ανάμεσα σε δύο στοιχεία, χρησιμοποιείται το όρισμα sep='':
  - >>> print('1', '2', '3', sep='')
    123
    >>>
- Το όρισμα sep χρησιμοποιείται για να καθοριστεί κάποιος συγκεκριμένος χαρακτήρας, διαφορετικός του κενού, για το διαχωρισμό πολλαπλών στοιχείων:
  - >>> print('1', '2', '3', sep='*')
    1*2*3
    >>>

Περισσότερα για την print (1)
(Ακολουθίες Διαφυγής - Escape Characters)

- *Ακολουθία διαφυγής: ο συνδυασμός κάποιου ειδικού χαρακτήρα με το χαρακτήρα* διαφυγής (\), όταν εμφανίζονται μέσα σε συμβολοσειρά.
- Οι ακολουθίες αυτές αντιμετωπίζονται ως ειδικές εντολές που εμπεριέχονται στη συμβολοσειρά.

Ακολουθία Διαφυγής	Δράση
\n	Κάνει την έξοδο να προχωρήσει στην επόμενη γραμμή.
\t	Κάνει την έξοδο να μετατοπιστεί στον επόμενο οριζόντιο στηλοθέτη (tab).
\'	Προκαλεί την εμφάνιση ενός μονού εισαγωγικού.
\"	Προκαλεί την εμφάνιση ενός διπλού εισαγωγικού.
****	Προκαλεί την εμφάνιση μιας ανάστροφης καθέτου (backslash).

Περισσότερα για την print (2)
(Ακολουθίες Διαφυγής - Escape Characters)

- Παραδείγματα:
  - print('1\n2\n3')

```
1
2
3
```

  - print('Δευτ\tΤρ\tΤετ')
    print('Πεμ\tΠαρ\tΣαβ')

```
Δευτ    Τρ      Τετ
Πεμ     Παρ     Σαβ
```

  - print('Μ\'αυτόν τον τρόπο)
    print('μπορούμε να τυπώσουμε \" και \'.')

```
Μ'αυτόν τον τρόπο
μπορούμε να τυπώσουμε " και '
```

  - print('C:\\user\\data')

```
C:\user\data
```

Περισσότερα για την print (Τελεστής +)

- Εμφάνιση Πολλαπλών Στοιχείων
- Όταν ο τελεστής + χρησιμοποιείται με δύο συμβολοσειρές πραγματοποιεί *συνένωσή τους (string concatenation)*.
  - print('Αυτό είναι ' + 'μία συμβολοσειρά.')

Αυτό είναι μία συμβολοσειρά.

  - my_text = 'Αυτό είναι ' + 'μία συμβολοσειρά.'
    print(my_text)

Περισσότερα για την print (Μορφοποίηση αριθμών - Στρογγυλοποίηση)

- Όταν ένας αριθμός κινητής υποδιαστολής εμφανίζεται από τη συνάρτηση print, μπορεί να εμφανιστεί με έως και 12 σημαντικά ψηφία
  - numerator = 164.0
    fraction= numerator / 12.0
    print('Η διαίρεση δίνει:', fraction)

Ο αριθμητικός τελεστής της διαίρεσης. Παρουσιάζεται παρακάτω.

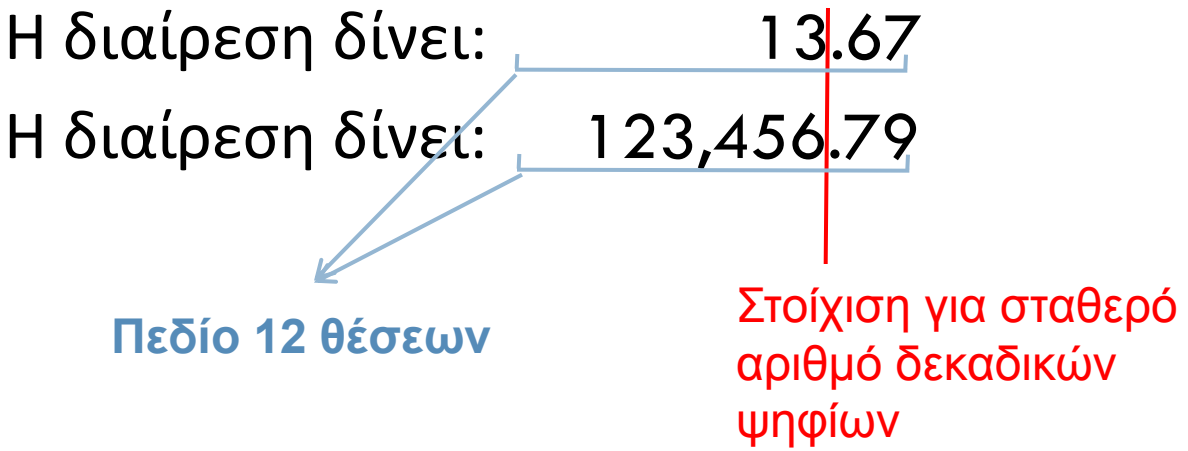
→ Η διαίρεση δίνει: 13.6666666667

- Η συνάρτηση format δίνει τη δυνατότητα για μορφοποίηση των αριθμητικών τιμών.
  - Στρογγυλοποίηση:
    print('Η διαίρεση δίνει:', format(fraction, '.2f')) → Η διαίρεση δίνει: 13.67
    print('Η διαίρεση δίνει:', format(fraction, '.1f')) → Η διαίρεση δίνει: 13.7
  - Εισαγωγή Διαχωριστικών Κομμάτων
    print(format(123456.789, ',.2f')) → 123,456.79

Περισσότερα για την print (Πλάτος πεδίου – στοίχιση αριθμών)

- Η format μπορεί επίσης να καθορίσει το ελάχιστο πλάτος πεδίου – τον ελάχιστο αριθμό θέσεων που χρησιμοποιούνται για την εμφάνιση του αριθμού
  - print('Η διαίρεση δίνει:', format(fraction, '12.2f'))

    print('Η διαίρεση δίνει:' ,format(123456.789, '12,.2f'))



Περισσότερα για την print (Μορφοποίηση ποσοστού - ακεραίων)

- Αντί για τη χρήση του f ως προσδιοριστή τύπου ενός αριθμού κινητής υποδιαστολής, μπορεί να χρησιμοποιηθεί το σύμβολο % για τη μορφοποίησή του ως ποσοστό.
  - Το σύμβολο % έχει ως αποτέλεσμα ο αριθμός να πολλαπλασιάζεται με το 100 και να εμφανίζεται ακολουθούμενος από το σύμβολο %
  - print(format(0.5, '%')) → 50.000000%
  - print(format(0.5, '.0%')) → 50%
- Ομοίως η format χρησιμοποιείται για μορφοποίηση ακεραίων (εκτός από την ακρίβεια), με προσδιοριστή τύπου d
  - print(format(123456, ',d'))
  - print(format(123456, '10d'))
  - print(format(123456, '10,d'))

→

```
123,456
123456
123,456
```

Εκτέλεση υπολογισμών

- Η ουσιαστικότερη λειτουργία ενός υπολογιστή και το βασικότερο τμήμα ενός αλγορίθμου.
- Πραγματοποιούνται μέσω των *μαθηματικών τελεστών* της γλώσσας προγραμματισμού:

Σύμβολο	Λειτουργία	Περιγραφή
+	Πρόσθεση	Προσθέτει δύο αριθμούς
-	Αφαίρεση	Αφαιρεί έναν αριθμό από έναν άλλο
*	Πολλαπλασιασμός	Πολλαπλασιάζει έναν αριθμό με έναν άλλο
/	Διαίρεση	Διαιρεί έναν αριθμό με έναν άλλο και δίνει το αποτέλεσμα ως έναν αριθμό κινητής υποδιαστολής
//	Ακέραια διαίρεση	Διαιρεί έναν αριθμό με έναν άλλο και δίνει το αποτέλεσμα ως έναν ακέραιο
%	Υπόλοιπο διαίρεσης	Διαιρεί έναν αριθμό με έναν άλλο και δίνει το υπόλοιπο
**	Ύψωση σε δύναμη	Υψώνει έναν αριθμό σε μία δύναμη

- Οι τελεστές χρησιμοποιούνται στην δημιουργία *μαθηματικών εκφράσεων* (μία *μαθηματική έκφραση πραγματοποιεί έναν υπολογισμό και έχει ως αποτέλεσμα* κάποια τιμή)

Εκτέλεση υπολογισμών (παραδείγματα)

```
>>> 12 + 2
14
>>> birth_year = 1984
>>> current_year = 2014
>>> age = current_year – birth_year
>>> print(age)
30
>>> year-birth
30
>>> name = 'kostas'
>>> name – age
TypeError: unsupported operand type(s) for -: 'str' and 'int'
>>> result = var1 * 2 + var2 * 3 + \
    var3 * 4 + var4 * 5
```

Παράδειγμα: Υπολογισμός έκπτωσης

```
# Το πρόγραμμα δέχεται σαν είσοδο την αρχική τιμή ενός προϊόντος
# και το ποσοστό της έκπτωσης και υπολογίζει την τελική τιμή

# Είσοδος της αρχικής τιμής του προϊόντος
initial_price = float(input('Δώσε την αρχική τιμή του προϊόντος: '))
# Είσοδος του ποσοστού της έκπτωσης
discount_rate = float(input('Δώσε το ποσοστό της έκπτωσης σαν δεκαδικό αριθμό: '))

# Υπολογισμός του ποσού της έκπτωσης.
discount = initial_price * discount_rate

# Υπολογισμός της τελικής (μειωμένης) τιμής.
sale_price = initial_price - discount

# Εμφάνιση της τελικής τιμής πώλησης.
print('Η τελική τιμή πώλησης είναι', sale_price)
```

Εκφράσεις Μεικτών Τύπων

- Όταν πραγματοποιείται κάποια πράξη μεταξύ δύο τιμών τύπου int, το αποτέλεσμα είναι τύπου int.
- Όταν πραγματοποιείται κάποια πράξη μεταξύ δύο τιμών τύπου float, το αποτέλεσμα είναι τύπου float.
- Όταν πραγματοποιείται κάποια πράξη μεταξύ ενός int και ενός float, η τιμή τύπου int μετατρέπεται προσωρινά σε float και το αποτέλεσμα της πράξης είναι τύπου float. (Μια έκφραση που χρησιμοποιεί τελεστέους διαφορετικού τύπου ονομάζεται *έκφραση μεικτών τύπων).*
  - my_number = 5 * 2.0

Όταν εκτελείται η εντολή αυτή, η τιμή 5 μετατρέπεται σε float (5.0) και στη συνέχεια πολλαπλασιάζεται με το 2.0. Το αποτέλεσμα, 10.0, εκχωρείται στη μεταβλητή my_number.

Η μετατροπή int σε float του προηγούμενου παραδείγματος, γίνεται έμμεσα, αφανώς (χωρίς χρήση της συνάρτησης float()).

Διαίρεση Κινητής Υποδιαστολής και Ακέραια Διαίρεση

- Ο τελεστής **/** δίνει το αποτέλεσμα ως έναν πραγματικό αριθμό (float), ενώ ο τελεστής **//** δίνει το αποτέλεσμα ως έναν ακέραιο αριθμό (int).

```
>>> 5 / 2
2.5
>>> 5 // 2
2
>>> -5 // 2
-3
>>>
```

- Όταν το αποτέλεσμα είναι θετικός αριθμός, το δεκαδικό του μέρος αποκόπτεται.
- Όταν το αποτέλεσμα είναι αρνητικός αριθμός, στρογγυλοποιείται στον πλησιέστερο προς το μείον άπειρο ακέραιο

Προτεραιότητα Τελεστών

- Ποίο το αποτέλεσμα της έκφρασης;
  - result = 5 + 12 / 3 * 2
- Η Python ακολουθεί την ίδια προτεραιότητα με τα μαθηματικά:
  - Αρχικά, εκτελούνται οι πράξεις που περικλείονται σε παρενθέσεις. Στη συνέχεια, όταν δύο τελεστές έχουν έναν κοινό τελεστέο, εφαρμόζεται πρώτα ο τελεστής με την υψηλότερη *προτεραιότητα.*
  - *Η προτεραιότητα των μαθηματικών τελεστών, από υψηλότερη προς χαμηλότερη,* είναι:
    - Ύψωση σε δύναμη: **
    - Πολλαπλασιασμός, διαίρεση και υπόλοιπο διαίρεσης: *, /, //, %
    - Πρόσθεση και αφαίρεση: +, -
  - Όταν δύο τελεστές με την ίδια προτεραιότητα μοιράζονται έναν τελεστέο, η πράξη πραγματοποιείται *από αριστερά προς τα δεξιά([1])*.
- Άρα ο υπολογισμός της παραπάνω έκφρασης γίνεται με την εξής σειρά:
  - 12 / 3 = 4, 4 * 2 = 8, 5 + 8 = 13

([1]) Εξαίρεση στον κανόνα: η ύψωση σε δύναμ
Π.χ. 2**3**4 υπολογίζεται ως 2**(3**4)

Ομαδοποίηση με Παρενθέσεις

- Για αλλαγή της προκαθορισμένης προτεραιότητας, χρησιμοποιούνται παρενθέσεις (όπως στα μαθηματικά)
  - 10 / (5 – 3)
  - (6 – 3) * (2 + 7) / 3
- Π.χ. Υπολογισμός μέσου όρου
  - average = (a + b + c) / 3

Ύψωση σε Δύναμη και Υπόλοιπο Διαίρεσης

- Ύψωση σε δύναμη: Τελεστής **
- x ** y = x^y
  - >>> 4 ** 2
    16
    >>> 5 ** 3
    125
- Ο τελεστής του υπολοίπου της διαίρεσης (%) ονομάζεται και *τελεστής modulo*
  - >>> 11 % 3
    2
    >>>
  - 11 / 3 = 3 με υπόλοιπο 2

Επίλυση προβλήματος

- Έκφραση προβλήματος με Μαθηματικές Εξισώσεις
- Μετατροπή Μαθηματικών Εξισώσεων σε Προγραμματιστικές Εντολές
- Σύνταξη και έλεγχος κώδικα

ΘΕΜΑΤΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ Η/Υ

Θεματική Ενότητα 5
Μεταβλητές και Μαθηματικοί και λογικοί τελεστές

Πληροφορική και Υπολογιστική Βιοϊατρική
Α. Κακαρούντας, Γ. Σπαθούλας, Π. Κοντού