



ΠΕΡΙΕΧΟΜΕΝΑ:

1. Εγκατάσταση - Εκτυπώσεις - Σχόλια
2. Μεταβλητές - Τύποι Δεδομένων
3. Δομή Ελέγχου
4. Δομές Επανάληψης
5. Λίστες - Πλειάδες
6. Σύνολα - Λεξικά
7. Συναρτήσεις 1
8. Συναρτήσεις 2
9. Κλάσεις 1
10. Κλάσεις 2
11. Κληρονομικότητα

psounis@gmail.com

1] ΕΓΚΑΤΑΣΤΑΣΗ - ΕΚΤΥΠΩΣΕΙΣ - ΣΧΟΛΙΑ

Link: www.repl.it

Εκτύπωση συμβολοσειράς:

```
print("Hello World!")
```

- Το "Hello World!" είναι μία συμβολοσειρά (σύμβολα μέσα σε μονά ή διπλά εισαγωγικά) ex01
- Η print είναι μία συνάρτηση (ενσωματωμένη, τυπώνει τη συμβολοσειρά - όρισμα στην κονσόλα)
- \n: Χαρακτήρας αλλαγής γραμμής
- #: Σχόλιο μία γραμμής
- """" Σχόλια πολλών γραμμών """" ex02

2] ΜΕΤΑΒΛΗΤΕΣ - ΤΥΠΟΙ ΔΕΔΟΜΕΝΩΝ

Τύποι Δεδομένων:

- **Συμβολοσειρά (string)**, παραδειγμα τιμής: "hello"
- **Ακέραιος (integer)**, παραδειγμα τιμής: 2
- **Πραγματικός (float)**, παραδειγμα τιμής: 2.1
- **Λογική μεταβλητή (boolean)** τιμές: True, False
- Δεν ορίζουμε ρητά τον τύπο στις μεταβλητές
- Εξάγεται αυτόματα
- Τελεστής Καταχώρησης: **Μεταβλητή = Τιμή** ex01

Πράξεις:

- Συμβολοσειρές: + (συνένωση συμβολοσειρών)
- Ακέραιοι, πραγματικοί: +, -, *, /, //, % και +=, -=, *=, /=, //=

Μετατροπές Τύπων:

- **int()**: Μετατρέπει το όρισμα σε ακέραιο αριθμό, π.χ.:

```
int(3.84)
```

αποκόπτει το δεκαδικό μέρος

- **float()**: Μετατρέπει το όρισμα σε πραγματικό αριθμό
 - Γενικά θα ξέρουμε ότι πράξεις μεταξύ ακεραίων και πραγματικών επιστρέφουν πραγματικό.
- **str()**: Μετατρέπει το όρισμα σε συμβολοσειρά
 - π.χ.

```
str(5.19)
```

- **bool()**: Μετατρέπει το όρισμα σε λογική μεταβλητή

Είσοδος από το πληκτρολόγιο:

- **input()**: Επιστρέφει συμβολοσειρά

```
name = input("Type your name: ")
```

Τελεστής καταχώρησης (ξανά):

```
ΜΕΤΑΒΛΗΤΗ = ΠΑΡΑΣΤΑΣΗ
```

Παράσταση = Τιμή, υπολογιζόμενη παράσταση ή συνάρτηση ex03

3] ΔΟΜΗ ΕΛΕΓΧΟΥ

• Σχεσιακοί Τελεστές

- == (ίσα)
- != (όχι ίσα)
- >, <, >=, <= (μεγαλύτερο από κ.ο.κ.)

• Λογικοί Τελεστές

- and (λογικό και)
- or (λογικό ή)
- not (λογικό όχι)

Δομή Ελέγχου: (Μόνο if-else, δεν υπάρχει switch):

```
if συνθήκη-if:
    εντολές-if
[elif συνθήκη-elif:
    εντολές-elif ]
[else:
    εντολές-else]
```

- Το σώμα με τις εντολές ένα tab δεξιά (αυστηρά!)
- Προσοχή στην άνω κάτω τελεία

ex01

ex02

συντμήσεις:

```
if συνθήκη-if: εντολή-if
εντολή-if if συνθήκη-if else: εντολή-else
```

4] ΔΟΜΕΣ ΕΠΑΝΑΛΗΨΗΣ

WHILE και FOR: (δεν υπάρχει do..while)

```
while συνθήκη:
    εντολή-1
    ...
    εντολή-N
```

```
for element in sequence:
    εντολή-1
    ...
    εντολή-N
```

ex01

όπου sequence μπορεί να είναι:

- Η συνάρτηση range: range(start, finish, step)
 - Επιστρέφει μία ακολουθία ακεραίων με αρχή το **start**, τέλος το **finish-1** και βήμα **step**.
 - Το step αν παραλειφθεί, θεωρείται ίσο με 1.
 - Το start αν παραλειφθεί, θεωρείται ίσο με 0.
- Δομές Δεδομένων (όπως η λίστα, η πλειάδα κ.α.) υποστηρίζονται οι **break** και **continue** και εκτελείται μπλοκ **else** αν δεν διακοπεί η εκτέλεση με break

ex02

ex03

ex04

5] ΛΙΣΤΕΣ - ΠΛΕΙΑΔΕΣ

Λίστα (ορισμός σε αγκύλες)

```
list_int = [1, 3, 4]
list_float = [3.12, 5.11, 1.1]
list_collection = [1, "big", [1, 2]]
```

ex01

ex02

ex03

- Indexing: πχ list_name[0], list_name[-1]
- Μέρος Λίστας: list_name[S:F] (από S έως F-1)
- Μέθοδοι:
 - list_name.append(element): στο τέλος
 - list_name.insert(index, element): στη θέση index
 - list_name.pop(index): Διαγράφει το στοιχείο στη θέση index
 - list_name.clear(): Διαγράφει όλα τα στοιχεία
 - list_name.sort(): Ταξινομεί τα στοιχεία
 - list_name.reverse(): Αντιστροφή των στοιχείων
- element in list: Χρησιμοποιείται σε if και for
- list * n = παραθέτει n φορές τη λίστα
- Built-in συναρτήσεις:
 - len(list_name): μήκος λίστας
 - min(list_name) και max(list_name)

ex04

ex05

Πλειάδα (tuple) (ορισμός σε παρενθέσεις)

```
int_tuple = (1, 2, 3)
```

- Ίδια συμπεριφορά, αλλά είναι immutable (δεν μπορεί να τροποποιηθεί)

6] ΛΕΞΙΚΑ - ΣΥΝΟΛΑ

Σύνολα: (ορισμός σε άγκιστρα)

```
int_set = {1, 2, 3}
empty_set = set()
```

ex01

- element in set: Χρησιμοποιείται σε if και for
- Μέθοδοι:
 - set_name.add(element)
 - set_name.remove(element)
 - set_name.clear()
 - set_name.union(set) # και τομή, διαφορά κ.λπ.

Λεξικά: (ζεύγη κλειδιού - τιμής)

```
empty = {}
person = {
    "grade": 13,
    (1,2): 29,
    "name": "Jim",
}
```

ex02

ex03

- dict_name[key]:** πρόσβαση αλλά και προσθήκη κλειδιού
- διαφορετικές μέθοδοι για διαπέραση (ex02) και
 - dict_name.pop(key)

Μετατροπές δομών:

- Μπορούμε να μετατρέπουμε από μία δομή στην άλλη (εφόσον είναι εφικτό, με τις: **list()**, **tuple()**, **set()** και **dict()**)

7] ΠΕΡΙΓΡΑΦΙΚΟΙ ΟΡΙΣΜΟΙ (COMPREHENSIONS) κ F-strings

Λίστα:

```
my_list = [number for number in range(3)]
```

```
my_list = [number for number in range(10) if number%2 == 0]
```

ex01

Σύνολο:

```
my_set = {number for number in range(3)}
```

```
my_set = {number for number in range(10) if number%2 == 0}
```

ex02

Λεξικό:

```
dict1 = {v:v**2 for v in range(10)}
```

```
dict2 = {v:v**2 for v in range(10) if v%2==0}
```

ex03

F-strings:

- `string = f"result: {x}"`
- Για πραγματικούς: `{f}` ή `{f:a}` ή `{f:.b}` ή `{f:a.b}`
- Για ακέραιους: `{i}` ή `{i:x}` ή `{i:o}` ή `{i:e}`

ex04

8] ΣΥΝΑΡΤΗΣΕΙΣ

Ορισμός:

```
def func_name(param1, param2,...):
```

ex01

```
...
```

```
return value(s)
```

ex02

Call by Object Reference

- Τα immutables (strings, tuples, int, float, boolean) δεν αλλάζουν
- Τα mutables (lists, sets, dict's) αλλάζουν σε όλες τις όλες τις τροποποιήσεις εκτός από καταχωρήσεις.

ex03

ex04

ex05

Προκαθορισμένες τιμές και keyword arguments:

```
def func(par1, par2 =0, par3 = 0, par4 = 0):  
    print(f"par1={par1} par2={par2} par3={par3} par4={par4}")  
  
func(1, par3="5")
```

ex06

Μεταβλητός Αριθμός Ορισμάτων

```
def my_sum(*numbers):  
    print(numbers) # a list  
    ...  
  
print(f"sum={my_sum(1,2,3,4,5)}")
```

ex07

9] ΚΛΑΣΕΙΣ

Ορισμός Κλάσης:

```
class Cow:
    def __init__(self, weight, hunger):
        self.weight = weight
        self.hunger = hunger

    def express(self):
        if self.hunger > 5:
            print("Moooooooooooooooooooo")
        else:
            print("Mowwww")

molly = Cow(500, 10)
molly.express()
```

- `init`: κατασκευαστής (δεν υπάρχει υπερφόρτωση) ex01
- Μέλη: Ορίζονται στον κατασκευαστή
- `self`: Αναφορά στο ίδιο το αντικείμενο
- 1^ο όρισμα υποχρεωτικά το `self` στις μεθόδους της κλάσης
- Όλα εξ'ορισμού `public`:
 - Δεν συνηθίζονται τα ιδιωτικά μέλη, αλλά υπάρχει μία σύμβαση: το όνομα του μέλους να ξεκινά με διπλό underscore. ex02
- Μπορούμε να ορίσουμε στατικά μέλη (χαρ/κά κλάσης) ex03

10] MAGIC or DUNDER methods

Οι μαγικές μέθοδοι προσφέρουν έξτρα λειτουργικότητα στην κλάση. Είναι πάρα πολλές, βλέπουμε μερικές:

- Υπερφόρτωση τελεστών, π.χ.

Τελεστής	Μέθοδος
<code>==</code>	<code>__eq__(self, other)</code> ex01
<code>+</code>	<code>__add__(self, other)</code>
<code>+=</code>	<code>__iadd__(self, other)</code> ex02

- να επιτρέψουμε επανάληψη επί του αντικειμένου:

Τελεστής	Μέθοδος
<code>len()</code>	<code>__len__(self)</code>
<code>[pos]</code>	<code>__getitem__(self, pos)</code> ex03

- Το αντικείμενο να μπορεί να λειτουργήσει ως συνάρτηση(!)

Τελεστής	Μέθοδος
<code>(params)</code>	<code>__call__(self, parameters)</code> ex04

11] ΚΛΗΡΟΝΟΜΙΚΟΤΗΤΑ

Ορισμός:

```
class Base:
    def __init__(self, b_attr):
        self.b_attr = b_attr
class Derived(Base):
    def __init__(self, b_attr, d_attr):
        super().__init__(b_attr)
        self.d_attr = d_attr
```

ex01

Υποστηρίζονται:

- Επαναορισμός (override) μεθόδων
- Πολλαπλή κληρονομικότητα
- Αφηρημένες κλάσεις

ex02

ex03

ex04



python (packages / frameworks)

python (professional)



3rd party tools:



python (advanced)

PIP / PyPi
RegEx
lambda
Decorators
Iterators
Namespaces
Attributes
meta-classes
...

python (modules)

math
random
json
os
sys
...

python 3



(games)



(databases)



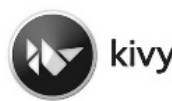
DBMS



ORACLE

(GUI)

(NUI)



(web development)



BACK-END



(AI / ML)

(computer vision)



(visualization)



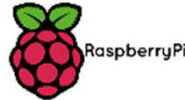
(ML Algorithms)



(math)



(robotics)



(web scraping)



(automation - scripting)



(IOT)



(Other)

(Data Science)
(Big Data)
(blockchain)
(cryptocurrencies)
...

1] ΕΓΚΑΤΑΣΤΑΣΗ - ΕΚΤΥΠΩΣΕΙΣ - ΣΧΟΛΙΑ

Άσκηση 1: Συντακτικά Λάθη

- Αν πληκτρολογήσουμε κάτι λάθος, τότε ο interpreter δεν είναι σε θέση να καταλάβει τι εννοούμε.
- Πειραματιστείτε κάνοντας τα εξής συντακτικά λάθη:
 - Αντί για print, γράψτε π.χ. pirnt
 - Ξεχάστε να κλείσετε την παρένθεση.
 - Ξεχάστε να κλείσετε τα εισαγωγικά της συμβολοσειράς

2] ΕΓΚΑΤΑΣΤΑΣΗ.

Άσκηση 2: Περίμετρος και Εμβαδόν κύκλου

Κατασκευάστε ένα πρόγραμμα το οποίο θα δέχεται από την είσοδο την ακτίνα R ενός κύκλου και θα υπολογίζει και θα τυπώνει:

- Την περίμετρό του (από τον τύπο $2\pi R$)
- Το εμβαδόν του (από τον τύπο πR^2)
- Για τον υπολογισμό, θεωρήστε ότι $\pi=3.14$

3] ΔΟΜΗ ΕΛΕΓΧΟΥ

Άσκηση 3: Μορφοποίηση ώρας

Κατασκευάστε ένα πρόγραμμα το οποίο θα δέχεται από την είσοδο πλήθος ωρών, πλήθος λεπτών και πλήθος δευτερολέπτων

- και τυπώνει στην οθόνη την ώρα στη μορφή: ΩΩ:ΛΛ:ΔΔ

4] ΔΟΜΕΣ ΕΠΑΝΑΛΗΨΗΣ

Άσκηση 4: Πυθαγόρειες τριάδες

Κατασκευάστε πρόγραμμα το οποίο:

- Θα τυπώνει όλες τις τριάδες (α, β, γ) με την ιδιότητα ότι:
$$\alpha^2 + \beta^2 = \gamma^2$$
- (για ακέραιους: $0 \leq \alpha, \beta, \gamma \leq 20$)

5] ΛΙΣΤΕΣ - ΠΛΕΙΑΔΕΣ

Κατασκευάστε ένα πρόγραμμα το οποίο:

- Θα αρχικοποιεί μία λίστα με 4 αγαπημένες σας ταινίες

Έπειτα:

- Θα ζητάει από το χρήστη να εισάγει μία νέα αγαπημένη του ταινία.
- Αν η ταινία υπάρχει ήδη στη λίστα, θα ενημερώνει το χρήστη ότι δεν έγινε η αποθήκευση
- Αν η ταινία δεν υπάρχει στη λίστα, θα την προσθέτει, θα ταξινομεί τη λίστα και έπειτα θα τυπώνει τη λιστά, καθώς και το πλήθος των αγαπημένων ταινιών του χρήστη.

6] ΣΥΝΟΛΑ - ΛΕΞΙΚΑ

Κάντε copy paste ένα κείμενο στα αγγλικά από κάποια σελίδα και αποθηκεύστε το σε μία συμβολοσειρά (ας είναι μια παράγραφος 5-6 γραμμών). Έπειτα:

- Μετατρέψτε το string σε λίστα
- Τυπώστε τη λίστα και παρατηρήστε τα περιεχόμενα της.
- Διατρέξτε τη λίστα ώστε να κατασκευάστε ένα λεξικό πλήθους εμφανίσεων κάθε χαρακτήρα (για κάθε χαρακτήρα, να αποθηκεύεται πόσες φορές εμφανίζεται ο χαρακτήρας στο κείμενο)
- Τυπώστε τον χαρακτήρα (ή τους χαρακτήρες) με το μέγιστο πλήθος εμφανίσεων.

7] ΠΕΡΙΓΡΑΦΙΚΟΙ ΟΡΙΣΜΟΙ

Κατασκευάστε μία λίστα που περιέχει τους άρτιους αριθμούς που είναι και πολλαπλάσια του 3, χρησιμοποιώντας για την κατασκευή, περιγραφική λίστα.

8] ΣΥΝΑΡΤΗΣΕΙΣ

Κατασκευάστε μία συνάρτηση με όνομα `float_average` η οποία δέχεται αυθαίρετο πλήθος πραγματικών αριθμών και υπολογίζει και επιστρέφει το μέσο όρο τους.

9] ΚΛΑΣΕΙΣ

Επεκτείνετε την κλάση σκύλος του w09ex01in.py με τα εξής:

- Το μέλος mood (ακέραιος μεταξύ 5 και 10). Να αρχικοποιείται σε 5.
- Τη μέθοδο eat (αυξάνει την διάθεση κατά 1)
- Τη μέθοδο bark:
 - Αν η διάθεση είναι πάνω από 5, να τυπώνεται "Woof Woof Woof"
 - Αλλιώς να τυπώνεται μόνο "Woof"
- Τη μέθοδο walk:
 - Αυξάνει τη διάθεση κατά 1

Έπειτα υλοποιήστε το σενάριο:

- Ορίστε έναν σκύλο
- Ο σκύλος γαβγίζει
- Πάει βόλτα
- Γαβγίζει
- Πάει βόλτα
- Γαβγίζει
- Τρώει
- Γαβγίζει

10(-) - 11] ΚΛΗΡΟΝΟΜΙΚΟΤΗΤΑ

Ο Βασιλιάς:

- έχει ένα βασίλειο (συμβολοσειρά)
- διοικεί (rule) τυπώνοντας "Now, I rule"

Ο Φιλόσοφος:

- ανήκει σε μία φιλοσοφική σχολή (συμβολοσειρά)
- σκέφτεται (think) τυπώνοντας "Now, I think"

Ο Μάρκος Αυρήλιος (121-180 μ.Χ.) ήταν αυτοκράτορας της Ρωμαϊκής Αυτοκρατορίας, που με το έργο του "Στοχασμοί" ανέδειξε χαρακτηριστικά της Στωϊκής φιλοσοφίας.

Η καθημερινή του ρουτίνα ήταν: σκεφτόταν, διοικούσε και μετά σκεφτόταν.

Προσομοιώστε μία μέρα της ζωής του Μάρκου Αυρήλιου με κατάλληλο πρόγραμμα Python.