

Εισαγωγή στην Python

Σημειώσεις μαθήματος

Ενότητα 4 – Συναρτήσεις

Δρ.Ν.Μανδέλλος, Δρ.Φ.Δογάνης

Εισαγωγή στην Python

Στόχοι ενότητας:

Δημιουργία και χρήση συναρτήσεων

Συναρτήσεις στην Python

Σύνταξη:

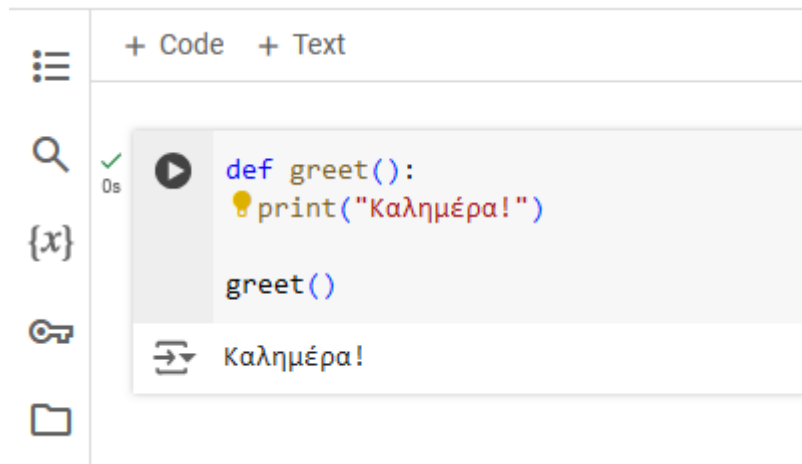
def όνομα_συνάρτησης([παράμετροι]):

Εντολές που εκτελούνται όταν καλείται η συνάρτηση

return [αποτέλεσμα] *# Προαιρετικό: μια συνάρτηση μπορεί και να μην επιστρέφει κάποια τιμή, δηλαδή να είναι void*

Συναρτήσεις στην Python

Παράδειγμα: Δημιουργία συναρτήσεως χωρίς παραμέτρους (arguments)



The screenshot shows a Python IDE interface. On the left is a sidebar with icons for a menu, search, variables, keybindings, and a file explorer. The main area has a tab labeled '+ Code + Text'. Below the tab, there is a code editor with the following Python code:

```
def greet():  
    print("Καλημέρα!")  
  
greet()
```

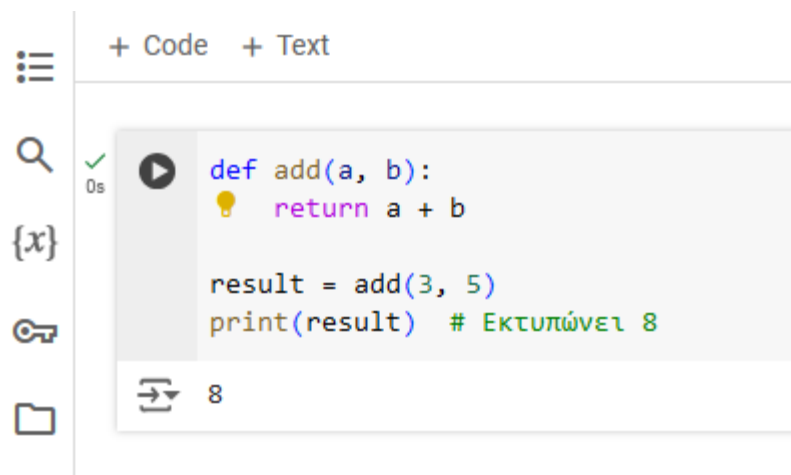
To the left of the code editor, there is a vertical toolbar with a play button icon. To the right of the play button, there is a green checkmark and the text '0s'. Below the code editor, there is a console area showing the output of the code: 'Καλημέρα!'.

Συναρτήσεις στην Python

Παράδειγμα: Δημιουργία συναρτήσεως με παραμέτρους (arguments) που επιστρέφει τιμές.

```
def add(a, b):  
    return a + b
```

```
result = add(3, 5)  
print(result)
```



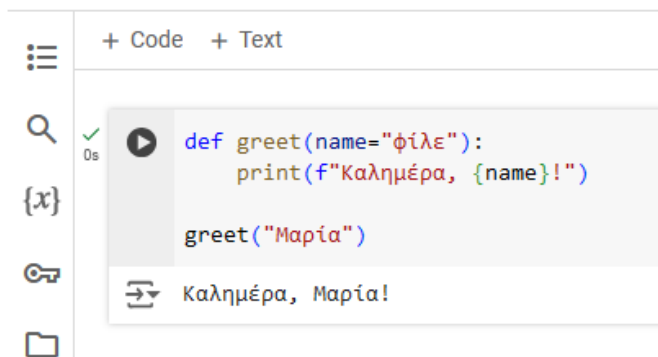
The screenshot shows a code editor interface with a sidebar on the left containing icons for a menu, search, variables, keybindings, and a file explorer. The main editor area has a tab labeled '+ Code + Text'. The code in the editor is:

```
def add(a, b):  
    return a + b  
  
result = add(3, 5)  
print(result) # Εκτυπώνει 8
```

Below the code, there is a console output area showing the result of the execution: 8. The output is preceded by a play button icon and a green checkmark, indicating successful execution.

Συναρτήσεις στην Python

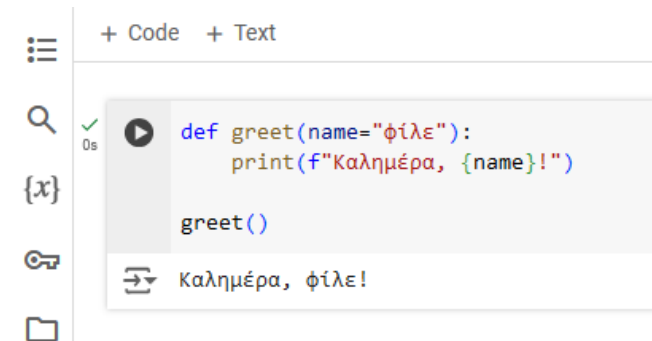
Παράδειγμα: Δημιουργία συναρτήσεως με παραμέτρους (arguments) χωρίς να επιστρέφει τιμές. Επίσης έχουμε και default values.



The screenshot shows a code editor with a sidebar on the left containing icons for a menu, search, variables, key, and file. The editor has tabs for '+ Code' and '+ Text'. The code is as follows:

```
def greet(name="φίλε"):  
    print(f"Καλημέρα, {name}!")  
  
greet("Μαρία")
```

Below the code, the output is displayed: Καλημέρα, Μαρία!



The screenshot shows a code editor with a sidebar on the left containing icons for a menu, search, variables, key, and file. The editor has tabs for '+ Code' and '+ Text'. The code is as follows:

```
def greet(name="φίλε"):  
    print(f"Καλημέρα, {name}!")  
  
greet()
```

Below the code, the output is displayed: Καλημέρα, φίλε!

Άσκηση 1

Να γράψετε πρόγραμμα σε Python το οποίο να ορίζει μια συνάρτηση με όνομα `celsius_to_kelvin`.

Η συνάρτηση θα δέχεται ως είσοδο τη θερμοκρασία σε βαθμούς Κελσίου και θα επιστρέφει τη θερμοκρασία σε Kelvin.

Στη συνέχεια:

- καλέστε τη συνάρτηση για θερμοκρασία 25 °C,
- και εμφανίστε το αποτέλεσμα στην οθόνη.

Άσκηση 1

```
# Reusable temperature converter  
def celsius_to_kelvin(celsius):  
    kelvin= celsius + 273.15  
    return kelvin  
  
# use the function  
result= celsius_to_kelvin(25.0)  
print(result)  # 298.15
```

Άσκηση 2

Να γράψετε πρόγραμμα σε Python το οποίο να υπολογίζει τον όγκο του αρχικού (stock) διαλύματος που απαιτείται για την παρασκευή ενός αραιωμένου διαλύματος.

Ορίστε μια συνάρτηση με όνομα ***dilution_volume*** η οποία θα δέχεται:

- **M1**: τη συγκέντρωση του αρχικού (stock) διαλύματος,
- **M2**: τη συγκέντρωση του τελικού διαλύματος,
- **V2**: τον συνολικό όγκο του τελικού διαλύματος.

Η συνάρτηση θα υπολογίζει και θα επιστρέφει τον απαιτούμενο όγκο V_1 του αρχικού διαλύματος, σύμφωνα με τη σχέση: $M_1 \cdot V_1 = M_2 \cdot V_2$

Στη συνέχεια:

- χρησιμοποιήστε τη συνάρτηση για να υπολογίσετε πόσα λίτρα διαλύματος 50% απαιτούνται ώστε να παρασκευαστεί 1 L διαλύματος συγκέντρωσης 10%,
- εμφανίστε το αποτέλεσμα στην οθόνη με κατάλληλο μήνυμα.

Άσκηση 2

```
# calculate volume of stock solution needed
def dilution_volume(M1, M2, V2):
    V1 = (M2 * V2) / M1
    return V1

# Example: 50% stock to 10% final, 1 L total
stock_vol = dilution_volume(0.50, 0.10, 1.0)
print(f"Need {stock_vol} L of stock solution")
```

Άσκηση 3

Να γράψετε πρόγραμμα σε Python το οποίο να υπολογίζει **τόσο τον όγκο του αρχικού (stock) διαλύματος όσο και τον όγκο του διαλύτη** που απαιτούνται για την παρασκευή ενός αραιωμένου διαλύματος.

Ορίστε μια συνάρτηση με όνομα ***dilution_calculator*** η οποία θα δέχεται:

M_1 : τη συγκέντρωση του αρχικού (stock) διαλύματος,

M_2 : τη συγκέντρωση του τελικού διαλύματος,

V_2 : τον συνολικό όγκο του τελικού διαλύματος.

Η συνάρτηση θα:

- Υπολογίζει τον απαιτούμενο όγκο V_1 του αρχικού διαλύματος, σύμφωνα με τη σχέση: $M_1V_1 = M_2V_2$
- Υπολογίζει τον όγκο του διαλύτη ως: $V_{\text{διαλύτη}} = V_2 - V_1$
- Επιστρέφει και τις δύο τιμές ως πλειάδα.

Ζητούμενο: καλέστε τη συνάρτηση για τη μετατροπή διαλύματος 50% σε διάλυμα 10% με τελικό όγκο 1 L,

- εμφανίστε στην οθόνη τον όγκο του αρχικού διαλύματος και τον όγκο του νερού (διαλύτη).

Άσκηση 3

Συνάρτηση που υπολογίζει τους όγκους

```
def dilution_calculator(M1, M2, V2):
```

```
    V1 = (M2 * V2) / M1
```

```
    diluent = V2 - V1
```

```
    return (V1, diluent)
```

Υπολογισμοί

```
result = dilution_calculator(0.5, 0.1, 1.0)
```

```
stock = result[0]
```

```
water = result[1]
```

Output

```
print(f"Stock:{stock} L, water:{water} L")
```

#Η πλειάδα μπορεί να οριστεί δυναμικά και ως

```
(stock, water) = dilution_calculator(0.5, 0.1, 1.0)
```

```
print(f"Stock:{stock} L, water:{water} L")
```