Συναρτήσεις

Εντολή `def`

Εισαγωγή. Τα περισσότερα προγράμματα τα οποία φτιάξαμε μέχρι τώρα θα μπορούσαμε να τα δούμε και ως ολοκληρωμένες μεθόδους οι οποίες αποτελούνται από τα εξής μέρη: (1) εισάγονται δεδομένα, (2) γίνονται υπολογισμοί και (3) εξάγεται και τυπώνεται ένα αποτέλεσμα. Για παράδειγμα, μπορούμε να βρούμε τον μεγαλύτερο μεταξύ δύο αριθμών ως εξής:

```
x = 5.0
y = 6.0

if x >= y:
   maxx = x
else
   maxx = y

print(maxx)
```

Το τμήμα του παραπάνω προγράμματος το οποίο κάνει τον υπολογισμό του μεγίστου μπορούμε να το διαχωρίσουμε ως εξής:

```
def maximum(x,y):
   if x >= y:
      maxx = x
   else
      maxx = y
   return maxx
```

Η λέξη `def` είναι δεσμευμένη, δηλαδή, είναι μία εντολή της python και ορίζει μία συνάρτηση.

- Η πρώτη γραμμή περιέχει το όνομα της συνάρτησης (η παραπάνω λέγεται `maximum`), επίσης τα ορίσματα σε παρένθεση και η γραμμή τελειώνει με την άνω-κάτω τελεία. Το όνομα μιας συνάρτησης ακολουθεί τους συνηθισμένους κανόνες της Python για τα ονόματα μεταβλητών.
- Όλες τις εντολές που περιέχονται στη συνάρτηση βρίσκονται σε εσοχή (tab).
- Η συνάρτηση τελειώνει με την εντολή `return` η οποία καθορίζει ποιά είναι η τιμή της συνάρτησης.

Το πρόγραμμά μας τώρα θα αποτελείται από τις γραμμές που ορίζουν την συνάρτηση και επίσης από τις παρακάτω εντολές:

```
result = maximum(5.0,6.0)
print(result)
```

Οι δύο παραπάνω εντολές αποτελούν το κύριο πρόγραμμα. Δείτε ότι το πρόγραμμά μας αποτελείται τώρα από δύο τμήματα: τη συνάρτηση και το κύριο πρόγραμμα (το οποίο ακολουθεί τη συνάρτηση και περιέχει μία γραμμή για τη χρήση της συνάρτησης.)

Ας δούμε τι ακριβώς συμβαίνει όταν στο πρόγραμμά μας περιέχεται μία συνάρτηση (π.χ. `maximum`) και γράφουμε `maximum(a,b)`.

- Λέμε ότι η συνάρτηση `maximum` *καλείται* με τις **πραγματικές παραμέτρους** (ή **ορίσματα**) `a,b`. Αυτές μπορεί να είναι αριθμοί (π.χ., `maxx=maximum(5.0,6.0)`) αλλά μπορεί να είναι και μεταβλητές οι οποίες έχουν τιμές (π.χ., `a=3.0; b=4.0; maxx=maximum(a,b)`).
- Με την κλήση της συνάρτησης συμβαίνουν δύο πράγματα: Οι **τυπικές παράμετροι** της συνάρτησης παίρνουν τιμές (`x=5.0; y=6.0` είτε `x=a; y=b`) και το πρόγραμμα συνεχίζει με τις εντολές εντός της συνάρτησης.
- Όταν βρεθεί εντολή `return` ο έλεγχος επιστρέφει στο κύριο πρόγραμμα.
- Η τιμή της μεταβλητής που βρίσκεται δίπλα στην `return` (δηλαδή, `maxx`) είναι η *τιμή της συνάρτησης*. Αυτή δίδεται στο κύριο πρόγραμμα (στη μεταβλητή `result`) αριστερά της ισότητας στην κλήση της συνάρτησης.
- Ο έλεγχος έχει τώρα επιστρέψει στο κύριο πρόγραμμα και εκτελούνται οι εντολές μετά της κλήση της συνάρτησης.
- Η τελευταία εντολή του κυρίου προγράμματος τυπώνει το αποτέλεσμα που πήραμε από την εκτέλεση της συνάρτησης.

Παράδειγμα. Γράψτε μία συνάρτηση η οποία να υπολογίζει την τιμή του πολυωνύμου $f(x) = x^2 - 2x + 1$. Καλέστε την από το κύριο πρόγραμμα για μία τιμή του x και τυπώστε τα $x, f(x)$.

```
def f(x):
    fvalue = x**2 - 2*x + 1
    return fvalue

x = 3.0
result = f(x)
print(x,result)
```

Παράδειγμα. Γράψτε μία συνάρτηση η οποία να ελέγχει αν το όνομά μας τελειώνει σε "ακης" (akis). Σε αυτή την περίπτωση η τιμή της θα είναι `True`, αλλιώς η τιμή της συνάρτησης θα είναι `False`.

```
def name_end(s):
    if s[-4:] == 'akis':
        return True
    else:
        return False

name = input("Give a name: ")
print(name_end(name))
```

Εντολή `return`. Σημειώστε ότι η εμφάνιση της εντολής `return` σε οποιοδήποτε σημείο στο σώμα μιας συνάρτησης τερματίζει την εκτέλεση των εντολών της συνάρτησης και επιστρέφει τη ροή του προγράμματος στο σημείο αμέσως μετά την κλήση της. Η χρήση της εντολής `return` είναι προαιρετική. Αν αυτή δεν εμφανίζεται ή η εμφάνισή της δεν ακολουθείται από κάποια έκφραση, τότε η συνάρτηση επιστρέφει την τιμή *None*.

Παράδειγμα. [Πηγή: Σημειώσεις Μ. Πλεξουσάκη.] Ας ορίσουμε μία συνάρτηση η οποία τυπώνει το μήνυμα `Hello!` όποτε κληθεί:

```
def sayHello():
    print('Hello!')
```

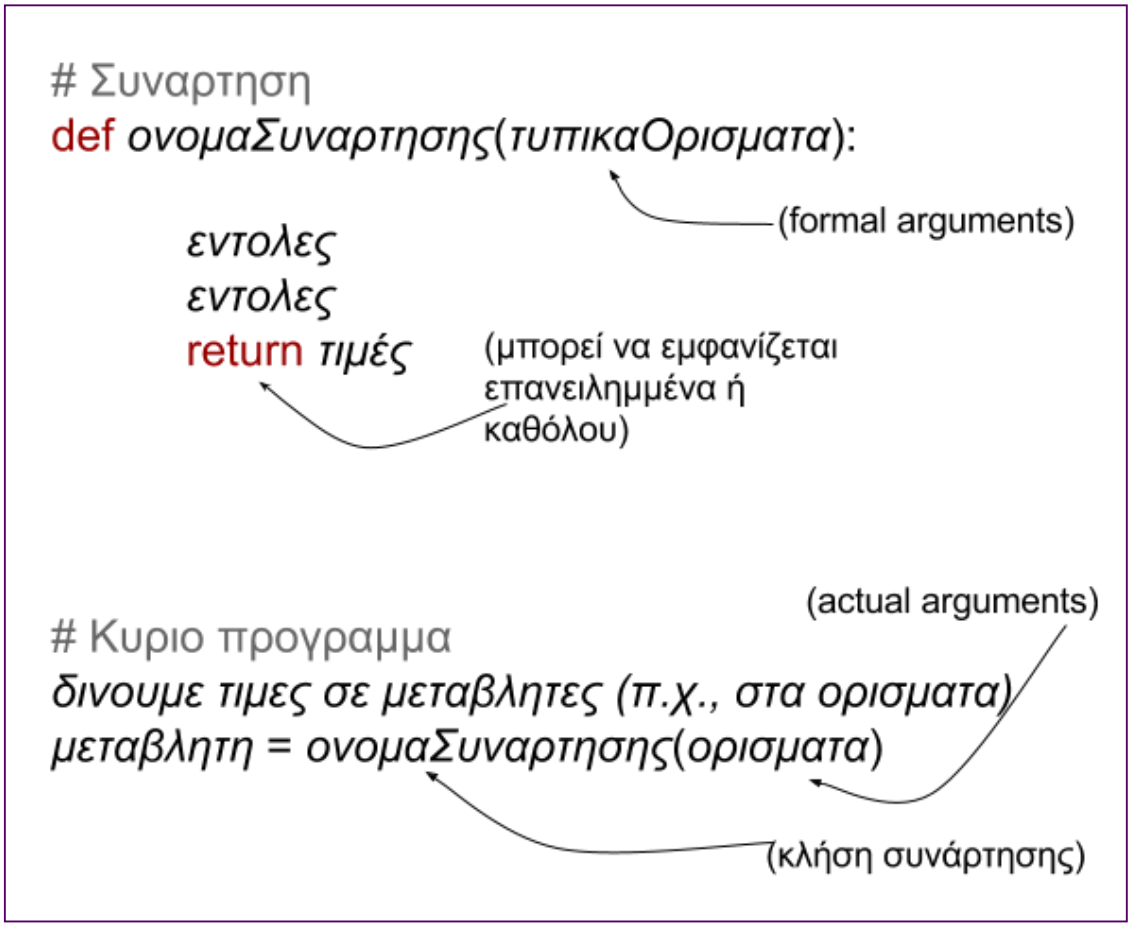
Ας την καλέσουμε:

```
>>> sayHello()
Hello!
```

Αν όμως είχαμε γράψει `print(sayHello(), 'Maria')` θα βλέπαμε το μήνυμα `None Maria` γιατί `None` είναι η τιμή η οποία επιστρέφει η συνάρτηση `sayHello`. Δείτε ακόμα ότι η συνάρτηση αυτή δεν έχει τυπικά ορίσματα και γι' αυτό η λίστα των τυπικών ορισμάτων της είναι κενή, αλλά η κλήση της συνάρτησης διατηρεί την κενή λίστα των ορισμάτων.

Παρατήρηση. Το κεντρικό πλεονέκτημα των συναρτήσεων είναι ότι μπορούμε να τις καλούμε επανειλημμένα από το κύριο πρόγραμμα. Ο κώδικας που περιέχεται στη συνάρτηση εκτελείται σε κάθε κλήση της από το κύριο πρόγραμμα.

Γραφική επανάληψη



Σχήμα. Η γενική μορφή μιας συνάρτησης και η κλήση της από το κύριο πρόγραμμα.

Ενσωματωμένες συναρτήσεις (built-in functions)

Πρέπει να παρατηρήσουμε ότι έχουμε ήδη δει συναρτήσεις σε προηγούμενα μαθήματα, όπως αυτές τις οποίες παρέχει το πακέτο `math`.

- `math.sin(x)`, `math.cos(x)`, `math.abs(x)`, etc.

Παρατηρήστε ότι όλες οι παραπάνω παίρνουν ένα όρισμα και επιστρέφουν μία τιμή, όπως ακριβώς οι συναρτήσεις που ορίζονται με την εντολή `def`.

Επίσης, έχουμε δει ενσωματωμένες συναρτήσεις της python.

- `type(a)`, `print()`, `len(s)` etc.

Παρατήρηση. Η χρήση συναρτήσεων είναι ιδαίτερα διαδεδομένη σε μία γλώσσα προγραμματισμού (όπως η python). Αυτό ισχύει και για τις ενσωματωμένες συναρτήσεις και για αυτές τις οποίες γράφει ο προγραμματιστής. Και τα δύο είδη συναρτήσεων χρησιμοποιούνται με τον ίδιο τρόπο, όπως είδαμε.

Μελέτη

Βιβλιογραφία

1. Δημήτριος Καρολίδης, *Μαθαίνετε εύκολα python* (Εκδόσεις Καρολίδη, 2016).
2. Κ. Μαγκούτης, Χ. Νικολάου, *Εισαγωγή στον αντικειμενοστραφή προγραμματισμό με Python*, (Αποθετήριο "Κάλλιπος", 2016) - Κεφάλαιο 4. Συναρτήσεις και εκτέλεση υπό συνθήκη.
3. J.V. Guttag, *Υπολογισμοί και προγραμματισμός με την python*, Κεφάλαιο 4.

Γράψτε τι άλλο θέλετε από αυτές τις σημειώσεις

Προηγούμενο Επόμενο