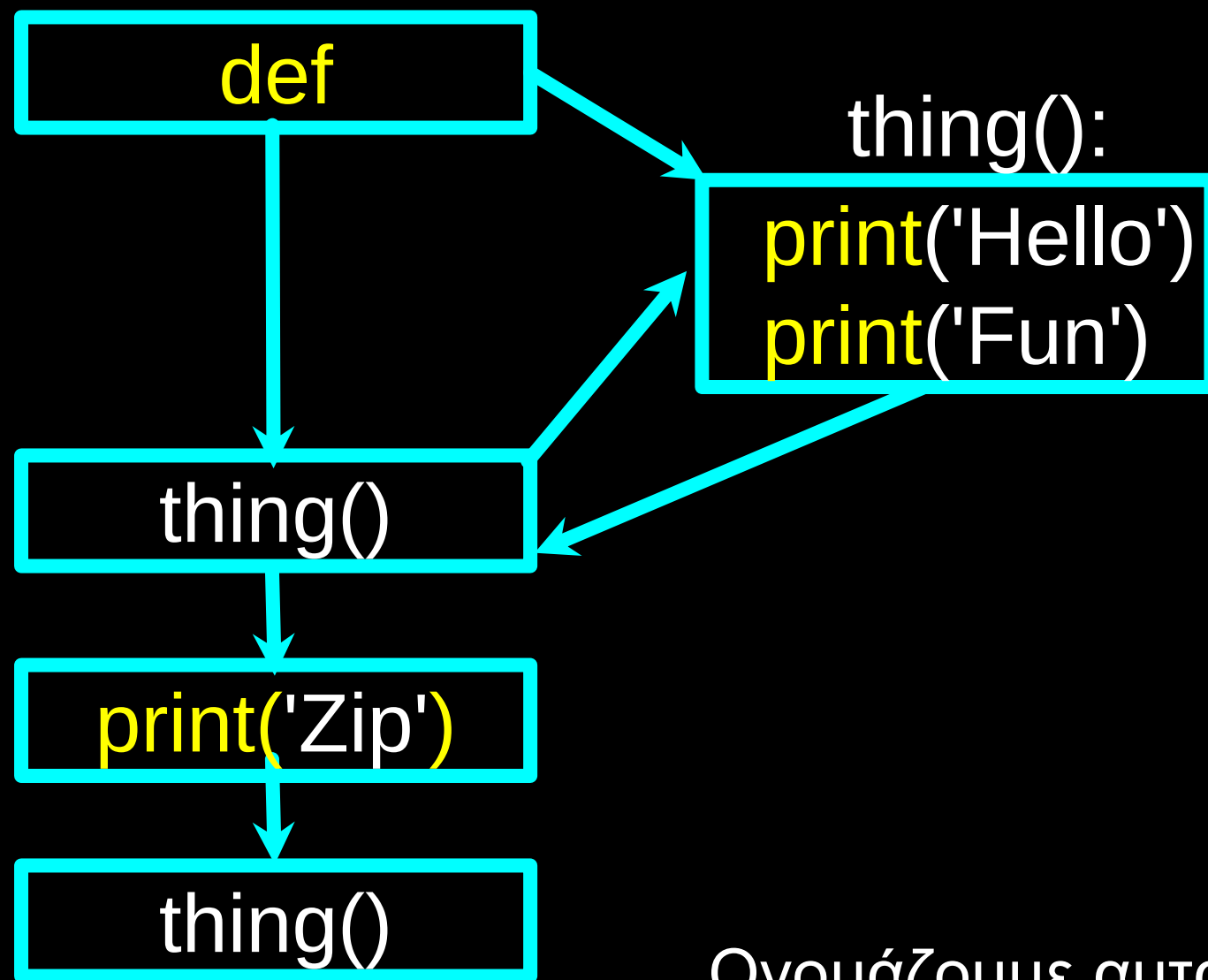


Συναρτήσεις

4

Προγράμματα

Αποθηκευμένα (και επαναχρησιμοποίηση) Βήματα



Πρόγραμμα:

```
def thing():  
    print('Hello')  
    print('Fun')
```

```
thing()  
print('Zip')  
thing()
```

Έξοδος:

Hello
Fun
Zip
Hello
Fun

Ονομάζουμε αυτά τα επαναχρησιμοποιήσιμα κομμάτια κώδικα «συναρτήσεις»

Συναρτήσεις στην Python

- Υπάρχουν δύο είδη συναρτήσεων στην Python.
 - **Ενσωματωμένες συναρτήσεις - Built-in functions** που παρέχονται ως μέρος της Python - `print()`, `input()`, `type()`, `float()`, `int()` ...
 - **Λειτουργίες που ορίζουμε μόνοι μας** και στη συνέχεια χρησιμοποιούμε
- Αντιμετωπίζουμε τα ονόματα των συναρτήσεων ως «νέες» **δεσμευμένες λέξεις** (δηλαδή, τις αποφεύγουμε ως ονόματα μεταβλητών)

Ορισμός Συνάρτησης

- Στην Python μια **συνάρτηση** είναι κάποιος επαναχρησιμοποιήσιμος κώδικας που λαμβάνει **όρισμα**(τα) ως είσοδο, κάνει κάποιους υπολογισμούς και μετά επιστρέφει ένα αποτέλεσμα ή αποτελέσματα
- Ορίζουμε μια **συνάρτηση** χρησιμοποιώντας τη δεσμευμένη λέξη **def**
- Καλούμε/εκτελούμε τη συνάρτηση χρησιμοποιώντας το όνομα της **συνάρτησης**, παρενθέσεις και **ορίσματα** μέσα σε μια παράσταση

big = **max**('Hello world')

Εκχώρηση

'w'

Αποτέλεσμα

Όρισμα

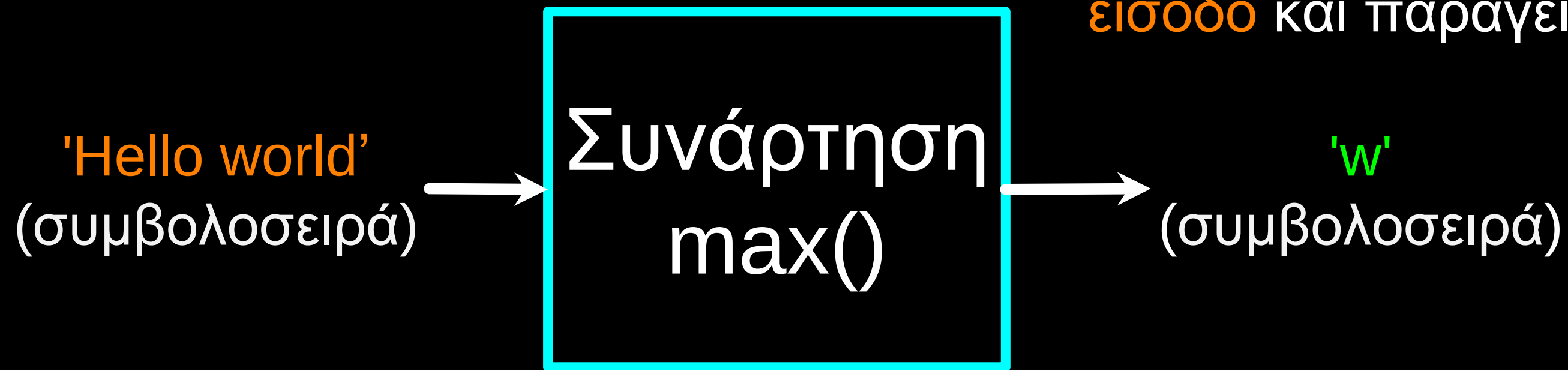
```
>>> big = max('Hello world')
>>> print(big)
w
>>> tiny = min('Hello world')
>>> print(tiny)

>>>
```

Η Συνάρτηση max

```
>>> big = max('Hello world')
>>> print(big)
w
```

Μια **συνάρτηση** είναι αποθηκευμένος κώδικας που χρησιμοποιούμε. Μια συνάρτηση λαμβάνει κάποια **είσοδο** και παράγει μια **έξοδο**.



Ο Guido έγραψε αυτόν τον κωδικό

Η Συνάρτηση max

```
>>> big = max('Hello world')
>>> print(big)
w
```

Μια **συνάρτηση** είναι αποθηκευμένος κώδικας που χρησιμοποιούμε. Μια συνάρτηση λαμβάνει κάποια **είσοδο** και παράγει μια **έξοδο**.

'Hello world'
(συμβολοσειρά)



```
def max(inp):
    blah
    blah
    for x in inp:
        blah
        blah
```



'w'
(συμβολοσειρά)

Ο Guido έγραψε αυτόν τον κωδικό

Μετατροπές Τύπου

- Όταν συνδυάζετε έναν ακέραιο και έναν κινητής υποδιαστολής σε μια έκφραση, ο ακέραιος αριθμός μετατρέπεται **σιωπηρά** σε κινητής υποδιαστολής
- Ελεγχόμενα η μετατροπή γίνεται με τις ενσωματωμένες συναρτήσεις `int()` και `float()`

```
>>> print(float(99) / 100)
0.99
>>> i = 42
>>> type(i)
<class 'int'>
>>> f = float(i)
>>> print(f)
42.0
>>> type(f)
<class 'float'>
>>> print(1 + 2 * float(3) / 4 - 5)
-2.5
>>>
```

Μετατροπές Συμβολοσειρών

- Μπορείτε επίσης να χρησιμοποιήσετε τις `int()` και `float()` σε μετατροπές μεταξύ συμβολοσειρών και ακεραίων
- Θα προκύψει **error / λάθος** αν η συμβολοσειρά δεν περιέχει αριθμητικούς χαρακτήρες

```
>>> sval = '123'
>>> type(sval)
<class 'str'>
>>> print(sval + 1)
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
TypeError: Can't convert 'int' object
to str implicitly
>>> ival = int(sval)
>>> type(ival)
<class 'int'>
>>> print(ival + 1)
124
>>> nsv = 'hello bob'
>>> niv = int(nsv)
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
ValueError: invalid literal for int()
with base 10: 'x'
```

Δικές μας Συναρτήσεις...

Κατασκευάζοντας τις Δικές μας Συναρτήσεις

- Δημιουργούμε μια νέα συνάρτηση χρησιμοποιώντας τη λέξη - κλειδί **def** ακολουθούμενη προαιρετικά από παραμέτρους σε παρένθεση
- Εισάγουμε εσοχή στο σώμα της συνάρτησης
- Αυτό **ορίζει** τη συνάρτηση αλλά **δεν** εκτελεί το σώμα της συνάρτησης

```
def print_lyrics():  
    print("Είμαι ξυλοκόπος και είμαι εντάξει.")  
    print('Κοιμάμαι όλη τη νύχτα και δουλεύω όλη μέρα.')
```

`print_lyrics()`:

```
print("Είμαι ξυλοκόπος και είμαι εντάξει.")  
print('Κοιμάμαι όλη τη νύχτα και δουλεύω όλη μέρα.')
```

```
x = 5  
print('Γειά')
```

```
def print_lyrics():  
    print("Είμαι ξυλοκόπος και είμαι εντάξει.")  
    print('Κοιμάμαι όλη τη νύχτα και δουλεύω όλη μέρα.')
```

```
print('Yo')  
x = x + 2  
print(x)
```

Γειά
Yo
7

Ορισμός και Χρήση

- Αφού **ορίσουμε** μια συνάρτηση, μπορούμε να την **καλέσουμε** (ή να την **εκτελέσουμε**) όσες φορές θέλουμε
- Αποτελεί πλέον ένα **αποθηκευμένο** και **επαναχρησιμοποιήσιμο** μοτίβο

```
x = 5  
print('Γειά')
```

```
def print_lyrics():  
    print("Είμαι ξυλοκόπος και είμαι εντάξει.")  
    print('Κοιμάμαι όλη τη νύχτα και δουλεύω όλη μέρα.')
```

```
print('Yo')  
print_lyrics()  
x = x + 2  
print(x)
```

Γειά

Yo

Είμαι ξυλοκόπος και είμαι εντάξει.

Κοιμάμαι όλη τη νύχτα και δουλεύω όλη μέρα.

7

Ορίσματα

- Ένα **όρισμα** είναι μια τιμή που περνάμε στη **συνάρτηση** ως **είσοδο** της όταν καλούμε τη συνάρτηση
- Χρησιμοποιούμε **ορίσματα** για να μπορέσουμε να οδηγήσουμε τη **συνάρτηση** να κάνει διαφορετικά είδη εργασίας σε **διαφορετικές** κλήσεις
- Τα **ορίσματα** τοποθετούνται σε παρένθεση μετά το **όνομα** της συνάρτησης

big = **max**('Hello world')



Όρισμα

Παράμετροι

Μια **παράμετρος** είναι μια μεταβλητή που χρησιμοποιούμε **μέσα** στον **ορισμό** της συνάρτησης. Είναι ένα «ψευδώνυμο» που επιτρέπει στον κώδικα της συνάρτησης να έχει πρόσβαση στα **ορίσματα** κατά την κλήση της συνάρτησης.

```
>>> def χαιρετισμός(γλώσσα) :  
...     if γλώσσα == 'es':  
...         print('Hola')  
...     elif γλώσσα == 'fr':  
...         print('Bonjour')  
...     else:  
...         print('Hello')  
...  
>>> χαιρετισμός('en')  
Hello  
>>> χαιρετισμός('es')  
Hola  
>>> χαιρετισμός('fr')  
Bonjour  
>>>
```

Επιστροφή Τιμών

Συχνά μια συνάρτηση δέχεται τα ορίσματά της, κάνει κάποιους υπολογισμούς και **επιστρέφει** μια τιμή που θα χρησιμοποιηθεί ως τιμή της καλούμενης συνάρτησης στην **έκφραση κλήσης**. Για αυτό χρησιμοποιείται η λέξη - κλειδί **return**.

```
def χαιρετισμός():  
    return "Γειά σου"
```

```
print(χαιρετισμός(), "Κώστα")  
print(χαιρετισμός(), "Έλη")
```

Γειά σου Κώστα
Γειά σου Έλη

Επιστροφή Τιμών

- Μια «παραγωγική» **συνάρτηση** είναι αυτή που παράγει ένα **αποτέλεσμα** (ή **τιμή επιστροφής**)
- Η δήλωση **return** τερματίζει την εκτέλεση της **συνάρτησης** και «επιστέφει» το **αποτέλεσμα** της **συνάρτησης**

```
>>> def χαιρετισμός(γλώσσα):  
...     if γλώσσα == 'es':  
...         return 'Hola'  
...     elif γλώσσα == 'fr':  
...         return 'Bonjour'  
...     else:  
...         return 'Hello'  
...  
>>> print(χαιρετισμός('en'), 'Glenn')  
Hello Glenn  
>>> print(χαιρετισμός('es'), 'Sally')  
Hola Sally  
>>>  
print(χαιρετισμός('fr'), 'Michael')  
Bonjour Michael  
>>>
```

Ορίσματα, Παράμετροι και Αποτελέσματα

```
>>> big = max('Hello world')  
>>> print(big)  
w
```

Ορίσμα
→ 'Hello world'

```
def max(inp):  
    blah  
    blah  
    for x in inp:  
        blah  
        blah  
    return 'w'
```

Παράμετρος
→

→ 'w'
↑
Αποτέλεσμα

Πολλαπλές Παράμετροι / Ορίσματα

- Μπορούμε να δηλώσουμε περισσότερες από μία παραμέτρους κατά τον ορισμό της συνάρτησης
- Απλά προσθέτουμε περισσότερα ορίσματα όταν καλούμε τη συνάρτηση
- Ο αριθμός και η σειρά των ορισμάτων και των παραμέτρων συμβαδίζουν

```
def άθροισμαδύο(a, b):  
    άθροισμαδύο = a + b  
    return άθροισμαδύο
```

```
x = άθροισμαδύο(3, 5)  
print(x)
```

8

Κενές (μη-παραγωγικές) Συναρτήσεις

- Όταν μια συνάρτηση δεν επιστρέφει τιμή, την ονομάζουμε «**κενή**» συνάρτηση
- Οι συναρτήσεις που επιστρέφουν τιμές είναι «παραγωγικές» συναρτήσεις
- Οι **κενές** συναρτήσεις «δεν είναι παραγωγικές»

Το function or not to function...

- Οργανώστε τον κώδικά σας σε «παραγράφους» - καταγράψτε μια ολοκληρωμένη σκέψη και «ονομάστε τον»
- Μην επαναλαμβάνετε - κάντε το να λειτουργήσει μία φορά και μετά ξαναχρησιμοποιήστε το
- Εάν κάτι γίνει μακροσκελές ή πολύπλοκο, χωρίστε το σε λογικά κομμάτια και βάλτε τα κομμάτια σε διαφορετικές συναρτήσεις
- Φτιάξτε μια βιβλιοθήκη από πράγματα που κάνετε ξανά και ξανά - ίσως μοιραστείτε τη με τους φίλους σας...

Σύνοψη

- Συναρτήσεις
- Ενσωματωμένες Συναρτήσεις
- Μετατροπές Τύπου (int, float)
- Μετατροπές Συμβολοσειρών
- Παράμετροι
- Ορίσματα
- Αποτελέσματα (καρποφόρες συναρτήσεις)
- Κενές (μη-καρποφόρες) συναρτήσεις
- Γιατί να χρησιμοποιήσεις συναρτήσεις?

Άσκηση

Δημιουργήστε μια συνάρτηση που ονομάζεται `computepay`, η οποία απαιτεί δύο παραμέτρους (ώρες και ρυθμός) και ξαναγράψτε σε αυτή τον υπολογισμό της αμοιβής σας με μιάμιση φορά το ωρομίσθιο για υπερωρίες.

Enter Hours: 45

Enter Rate: 10

Pay: 475.0

$$475 = 40 * 10 + 5 * 15$$