ΜΕΜ104 Γλώσσα Προγραμματισμού Ι

Μιχάλης Πλεξουσάκης, Γιάννης Λιλής

Εβδομάδα 21-25 Σεπτεμβρίου 2020

Μαθηματικά και Εφαρμοσμένα Μαθηματικά

Περιεχόμενα

Γιατί προγραμματισμός με τη γλώσσα Python;

Γιατί προγραμματισμός με τη γλώσσα Python

- Ο προγραμματισμός είναι ένα ευέλικτο εργαλείο για τη λύση προβλημάτων στις επιστήμες και την τεχνολογία
- Αν χρησιμοποιείτε ένα κινητό τηλέφωνο, ή ένα μηχάνημα ανάληψης χρημάτων τότε, αλληλεπιδράτε με ένα υπολογιστικό σύστημα μέσω ενός προγράμματος
- Facebook, tweeter, Instagram, dropbox, ...
- Ηλεκτρονικά παιχνίδια, αυτοματοποίηση εργασιών ή καθηκόντων

Γιατί προγραμματισμός με τη γλώσσα Python

- Η Python είναι μια γλώσσα προγραμματισμού με απλό συντακτικό, εξαιρετική αναγνωσιμότητα, φορητότητα (portability) και μοντέρνα χαρακτηριστικά που την κάνουν κατάλληλη ως πρώτη γλώσσα προγραμματισμού.
- Η Python είναι μία γλώσσα *υψηλού επιπέδου* (όπως η C/C++, Java, Fortran, κλπ). Ο κώδικας μία τέτοιας γλώσσας πρέπει να μετατραπεί στη λεγόμενη *γλώσσα μηχανής* ώστε να εκτελεστεί από τον Η/Υ. Η επεξεργασία αυτή γίνεται από διερμηνευτές (interpreters) και μεταγλωττιστές (compilers). Στην περίπτωση της Python η επεξεργασία γίνεται από διερνηνευτή.

Γιατί προγραμματισμός με τη γλώσσα Python

- Η Python γράφτηκε από τον Guido van Rossum στα τέλη της δεκαετίας 1980-90. Η έκδοση 2.0 δημοσιεύτηκε στις 16 Οκτωβρίου 2000 και η έκδοση 3.0, η οποία δεν είναι, εν γένει, σύμφωνη (compatible) με τις προηγούμενες εκδόσεις, στις 3 Δεκεμβρίου 2008. Στο μάθημα θα χρησιμοποιήσουμε την έκδοση 3.5 ή νεώτερη.
- Αν ο προσωπικός σας υπολογιστής χρησιμοποιεί το λειτουργικό σύστημα Linux ή το MacOS τότε η Python είναι ήδη εγκατεστημένη (βεβαιωθείτε όμως ότι έχετε τον διερμηνέα για την Python 3 γράφοντας python3 στη γραμμή εντολών).

Γιατί προγραμματισμός με τη γλώσσα Python

- Αν έχετε το λειτουργικό σύστημα Windows μπορείτε να εγκαταστήσετε την Python από τη σελίδα
  `http://python.org`
- Ανεξάρτητα από το λειτουργικό σύστημα, μπορείτε να εγκαταστήσετε το περιβάλλον εργασίας Canopy από τη σελίδα
  `https://assets.enthought.com/downloads/`
- Αν δεν θέλετε να εγκαταστήσετε τη γλώσσα Python στον υπολογιστή σας μπορείτε να τη χρησιμοποιήσετε διαδικτυακά, π.χ. στη διεύθυνση
  `https://repl.it/languages/python3`

Γιατί προγραμματισμός με τη γλώσσα Python

- Οι ελεύθερα προσβάσιμοι υπολογιστές του Τμήματος χρησιμοποιούν το λειτουργικό σύστημα Linux και έχουν, φυσικά, εγκατεστημένο τον διερμηνέα για την Python 3. Περιέχουν επίσης, το κέλυφος IDLE, έναν διαδραστικό διερμηνέα της Python με ενσωματωμένο κειμενογράφο.
- Τα εργαστήρια του μαθήματος θα γίνονται σε αυτά τα υπολογιστικά συστήματα, αλλά μπορείτε να χρησιμοποιείτε τον δικό σας φορητό υπολογιστή.

Γιατί προγραμματισμός με τη γλώσσα Python

Συνοψίζοντας, επιλέγουμε τη γλώσσα Python γιατί είναι

- Γλώσσα ανοικτού κώδικα
- Γενικής χρήσης, υψηλού επιπέδου γλώσσα προγραμματισμού
- Απλό συντακτικό, εξαιρετική αναγνωσιμότητα
- Κατάλληλη για αρχάριους και για έμπειρους προγραμματιστές
- Υποστηρίζεται από εκατοντάδες βιβλιοθήκες, για κάθε είδους ανάγκη

Γιατί προγραμματισμός με τη γλώσσα Python;

Python was named the fastest growing programming language of 2017 by Stack Overflow. This was attributed to its ties to machine learning and data science. But we're not all trying to build HAL 9000. Python's newest fans also include business majors, analysts, and content marketers. Why?

- Python has become a go-to language for analyzing data
- As a first programming language, it's simple and straightforward
- While easy to learn, Python has the power to grow with you
- It's an in-demand skill for anyone who wants to be data-driven

Πως μοιάζει η γλώσσα Python;

Πώς μοιάζει η γλώσσα Python;

- Η Python αναπτύχθηκε αρχικά ως γλώσσα διδασκαλίας
- Το απλό συντακτικό και η ευκολίας εκμάθησης την έκαναν δημοφιλή τόσο σε αρχάριους όσο και σε έμπειρους προγραμματιστές
- Είναι πολλές φορές ευκολότερο να διαβάζει και να καταλαβαίνει κάποιος κώδικα Python παρά ψευδοκώδικα

Πώς μοιάζει η γλώσσα Python;

```
# set the midpoint
midpoint = 5

# make two empty lists
lower = []; upper = []

# split the numbers into lower and upper
for i in range(10):
    if i < midpoint:
        lower.append(i)
    else:
        upper.append(i)

print('lower:', lower)
print('upper:', upper)
```

Πώς μοιάζει η γλώσσα Python;

Αν εκτελέσουμε τον παραπάνω κώδικα θα δούμε

```
lower: [0, 1, 2, 3, 4]
upper: [5, 6, 7, 8, 9]
```

Ας εξηγήσουμε μερικά στοιχεία από αυτό το πρόγραμμα:

- Οι γραμμές στις οποίες ο πρώτος μη κενός χαρακτήρας είναι η δίεση # αποτελούν σχόλια και αγνοούνται από τον διερμηνέα της Python
- Οι κενοί χαρακτήρες (μπορεί να) είναι σημαντικοί. Η Python χρησιμοποιεί τη λεγόμενη *εσοχή* (indentation) για να απομονώσει συγκεκριμένα κομμάτια κώδικα
- Το τέλος μιας εντολής σημειώνεται είτε με αλλαγή γραμμής είτε με τον χαρακτήρα `;`

Πώς μοιάζει η γλώσσα Python;

- Μπορούμε να χρησιμοποιήσουμε συμβολικά ονόματα, όπως `midpoint`, `lower` και `upper` για να αναφερθούμε σε σταθερές (π.χ. το 5) είτε σε πιο πολύπλοκες δομές (οι λίστες `lower` και `upper`)
- Αποτελέσματα της εκτέλεσης του προγράμματος εμφανίζονται με τις εντολές `print`
- Δεν είναι δύσκολο να καταλάβει κάποιος ότι το παραπάνω πρόγραμμα χώρισε τους ακέραιους στο διάστημα $[0, 10)$ σε αυτούς που είναι μικρότεροι από το 5 και αυτούς που είναι μεγαλύτεροι (από το 5).
- Μπορείτε να προβλέψετε ποιά αλλαγή θα χρειαζόταν το πρόγραμμα αν θέλαμε να κάνουμε το ίδιο για τους ακέραιους στο διάστημα $[0, 20)$;

Αντικείμενα, παραστάσεις και αριθμητικοί τύποι

Αντικείμενα, παραστάσεις και αριθμητικοί τύποι

- Ένα πρόγραμμα Python είναι μια ακολουθία από ορισμούς και εντολές
- Οι ορισμοί *αποτιμούνται* και οι εντολές *εκτελούνται* σε ένα περιβάλλον που ονομάζεται *κέλυφος* (shell).
- Τα αντικείμενα (objects) είναι τα βασικά στοιχεία που χρησιμοποιεί η Python. Κάθε αντικείμενο έχει έναν *τύπο* (type) που ορίζει τι μπορεί να κάνουν τα προγράμματα με αντικείμενα αυτού του τύπου.
- Τα αντικείμενα μπορεί να είναι *βαθμωτά* (scalar) ή μή.

Αντικείμενα, παραστάσεις και αριθμητικοί τύποι

Η Python έχει τέσσερις τύπους βαθμωτών αντικειμένων:

- Ο τύπος `int` χρησιμοποιείται για την αναπαράσταση ακεραίων. Οι τιμές τύπου `int` γράφονται με τον συνήθη τρόπο, π.χ. 9, -4 ή 12345.
- Ο τύπος `float` χρησιμοποιείται για την αναπαράσταση πραγματικών αριθμών. Οι τιμές τύπου `float` περιλαμβάνουν πάντοτε μια υποδιαστολή, όπως οι 2.0, -3.4, ή 123.78. Είναι επίσης δυνατόν να γράφονται με τον λεγόμενο *επιστημονικό συμβολισμό* (scientific notation), όπως π.χ. 2.34E3 ή 1.88e-2, δηλαδή τιμές ίσες με 2.34×10^3 και 1.88×10^{-2}, αντίστοιχα.

Αντικείμενα, παραστάσεις και αριθμητικοί τύποι

- Ο τύπος `bool` χρησιμοποιείται για την αναπαράσταση λογικών τιμών `True` (αληθής) και `False` (ψευδής).
- Ο τύπος `None` με μία μοναδική τιμή, για τον οποίο θα μιλήσουμε αργότερα.

Αντικείμενα μπορούν να συνδιάζονται με *τελεστές* (operators) για να σχηματίσουν *παραστάσεις* (expressions), κάθε μια από τις οποίες αποτιμάται σε ένα αντικείμενο συγκεκριμένου τύπου.

Για παράδειγμα, η παράσταση `3+2` δηλώνει το αντικείμενο `5`, τύπου `int`, ενώ η παράσταση `3.0+2.0` δηλώνει το αντικείμενο `5.0` τύπου `float`.

Ο τελεστής `==` ελέγχει αν δυο παραστάσεις έχουν την ίδια τιμή και ο τελεστής `!=` ελέγχει αν έχουν διαφορετικές τιμές.

Αντικείμενα, παραστάσεις και αριθμητικοί τύποι

```
>>> 3 + 2
5
>>> 3.0 + 2.0
5.0
>>> 3 != 2
True
>>> 2 != 2.0
False
>>> 3 + 2 == 3.0 + 2
True
```

Το σύμβολο `>>>` είναι η λεγόμενη *προτροπή* του κελύφους (shell prompt) που υποδεικνύει ότι ο διερμηνευτής της Python περιμένει από τον χρήση να πληκτρολογήσει εντολές.

Αντικείμενα, παραστάσεις και αριθμητικοί τύποι

Για να βρούμε τον τύπο ενός αντικειμένου χρησιμοποιούμε τη συνάρτηση `type`:

```
>>> type(3)
<class 'int'>
>>> type(2.0)
<class 'float'>
```

Οι αριθμητικοί τελεστές της Python που χρησιμοποιούνται με τους τύπους `int` και `float` είναι οι ακόλουθοι:

- `i + j`. Αν τα `i` και `j` έχουν τύπο `int`, τότε το αποτέλεσμα είναι μια τιμή `int`. Αν κάποιο από αυτά έχει τύπο `float`, τότε το αποτέλεσμα είναι μια τιμή `float`. Όμοια για την αφαίρεση.

Αντικείμενα, παραστάσεις και αριθμητικοί τύποι

- `i * j`. Αν τα `i` και `j` έχουν τύπο `int`, τότε το αποτέλεσμα είναι μια τιμή `int`. Αν κάποιο από αυτά έχει τύπο `float`, τότε το αποτέλεσμα είναι μια τιμή `float`.
- `i / j`. Η διαίρεση του `i` με το `j`. Το αποτέλεσμα είναι *πάντοτε* μια τιμή `float`.
- Η παράσταση `i // j` είναι η διαίρεση ακεραίων. Η διαίρεση ακεραίων επιστρέφει το πηλίκο και αγνοεί το υπόλοιπο. Για παράδειγμα, η τιμή της παράσταση `6 // 4` είναι `1`, ενώ η τιμή της παράστασης `-5 // 2` είναι `-3`.
- `i % j` είναι το υπόλοιπο της διαίρεσης του ακεραίου `i` με τον ακέραιο `j`.

Έχουμε, `x % y` $=$ `x` $-$ (`x // y`) * `y`, όπου `x // y` $= \lfloor$ `x`$/$`y` $\rfloor$ και $\lfloor \cdot \rfloor$ είναι η συνάρτηση πάτωμα, $\lfloor x \rfloor = \max\{n \in \mathbb{Z} : n \leq x\}$.

Αντικείμενα, παραστάσεις και αριθμητικοί τύποι

- `i**j` είναι το `i` υψωμένο στη δύναμη `j`. Αν τα `i` και `j` έχουν τύπο `int`, τότε το αποτέλεσμα είναι μια τιμή `int`. Διαφορετικά, το αποτέλεσμα είναι μια τιμή τύπου `float`.

Οι τελεστές σύγκρισης είναι οι `==` (ίσον), `!=` (διάφορο του), `>`, `>=`, `<`, `<=` με τη προφανή σημασία τους. Οι τελεστές που εφαρμόζονται στον τύπο `bool` είναι οι `and`, `or` και `not`:

- Η παράσταση `not a` δίνει τιμή `True` αν το `a` είναι `False` και τιμή `False` αν το `a` είναι `True`.
- Η παράσταση `a or b` είναι `True` αν τουλάχιστον ένα από τα `a` ή `b` είναι `True`, διαφορετικά είναι `False`.
- Η παράσταση `a and b` είναι `True` αν και το `a` και `b` είναι `True`, διαφορετικά είναι `False`.

Αντικείμενα, παραστάσεις και αριθμητικοί τύποι

Η *προτεραιότητα* των τελεστών είναι (υψηλή προς χαμηλή)

Τελεστής	Προτεραιότητα
$(\cdot)$	Παρενθέσεις
`**`	Ύψωση σε δύναμη
`+, -`	Τελεστές προσήμου
`*, /, //, %`	Πολ/μος, διαίρεση, ακέραια διαίρεση, υπόλοιπο
`+, -`	Πρόσθεση, αφαίρεση
`==, !=, >, >=, <, <=`	Τελεστές σύγκρισης
`not`	Άρνηση
`and`	Λογικό ‘και’
`or`	Λογικό ‘ή’

Αντικείμενα, παραστάσεις και αριθμητικοί τύποι

Όταν δύο τελεστές έχουν την ίδια προτεραιότητα, τότε η *προσεταιριστικότητά* τους ορίζει τη σειρά των πράξεων. ¨Ολοι οι τελεστές του προηγούμενου πίνακα είναι αριστερά προσεταιριστικοί, δηλαδή ομαδοποιούν από αριστερά προς τα δεξια, με εξαίρεση τον τελεστή `**` ο οποίος είναι δεξιά προσεταιριστικός.

```
>>> 12/3*5
20.0
>>> 2**-3
0.125
>>> 2**3**2
512
>>> (2**3)**2
64
```