1. Με ποια εντολή μπορούμε να ελέγξουμε τον τύπο δεδομένων. Να γράψετε τη σύνταξη και τη λειτουργία της. (και σελ 30)

type()

Η συνάρτηση type χρησιμοποιείται για να εμφανίσουμε στην οθόνη τον τύπο μιας μεταβλητής (ενός δεδομένου). Χρησιμοποιείται γράφοντας την εντολή type και δίπλα το δεδομένο μέσα σε παρενθέσεις, και όταν εκτελείται η εντολή, επιστρέφεται ο τύπος του.

παραδείγματα

```
>>>text = "Hello World."
```
#Η μεταβλητή text θα πάρει ως τιμή το κείμενο: Hello World

```
>>>type(text)
```
#Η python αναγνωρίζει ότι η μεταβλητή text είναι τύπου str και θα μας το εμφανίσει στην οθόνη.

```
>>>x = 35.2
```
#Η μεταβλητή x θα πάρει ως τιμή το 35,2

```
>>>type(x)
```
#Η python αναγνωρίζει ότι η μεταβλητή x είναι τύπου float και θα μας το εμφανίσει στην οθόνη.

2. Πως γίνεται η εφαρμογή των σχεσιακών τελεστών σε συμβολοσειρές. (και σελ 128)

Για τη σύγκριση μεταξύ συμβολοσειρών η λειτουργία των τελεστών στηρίζεται στη λεξικογραφική διάταξη των χαρακτήρων, με την εξής σειρά:

1. Αριθμοί (ως συμβολοσειρά)
2. Κεφαλαίοι λατινικοί χαρακτήρες
3. Μικροί λατινικοί χαρακτήρες
4. Κεφαλαίοι ελληνικοί χαρακτήρες
5. Μικροί ελληνικοί χαρακτήρες

Ισχύει δηλαδή:

'0' < '1' <…< '9' < 'A' < 'B' < 'C <…< 'Z' < 'a' < 'b' <…< 'z' < 'Α' <' Β' < 'Γ' <…< 'Ω' < 'α' < 'β' < …<'ω'

Η σύγκριση γίνεται συγκρίνοντας το πρώτο γράμμα μιας λέξης με το πρώτο γράμμα της άλλης. Αν είναι διαφορετικά, τότε το αποτέλεσμα της σύγκρισης θα είναι True ή False σύμφωνα με την παραπάνω διάταξη χαρακτήρων. Αν τα δυο γράμματα είναι ίσα τότε συγκρίνονται τα δεύτερα, τα τρίτα κοκ μέχρι να βρεθούν διαφορετικά γράμματα.

3. Να περιγράψετε τη λειτουργία του λογικού τελεστή and.

Ο λογικός τελεστής AND εκτελεί την πράξη της σύζευξης 2 πράξεων. Δέχεται 2 λογικές εκφράσεις και παράγει αποτέλεσμα ΠΑΝΤΑ FALSE, όταν τουλάχιστον μια από τις 2 εκφράσεις είναι FALSE.

ή

Ο λογικός τελεστής AND εκτελεί την πράξη της σύζευξης 2 πράξεων. Δέχεται 2 λογικές εκφράσεις και παράγει αποτέλεσμα TRUE, ΜΟΝΟ όταν και οι 2 εκφράσεις είναι TRUE.

4. Να περιγράψετε τη λειτουργία του λογικού τελεστή or.

Ο λογικός τελεστής OR εκτελεί την πράξη της διάζευξης 2 πράξεων. Δέχεται 2 λογικές εκφράσεις και παράγει αποτέλεσμα ΠΑΝΤΑ TRUE, όταν τουλάχιστον μια από τις 2 εκφράσεις είναι TRUE.

ή

Ο λογικός τελεστής OR εκτελεί την πράξη της διάζευξης 2 πράξεων. Δέχεται 2 λογικές εκφράσεις και παράγει αποτέλεσμα FALSE, ΜΟΝΟ όταν και οι 2 εκφράσεις είναι FALSE.

5. Να περιγράψετε τη λειτουργία του λογικού τελεστή not.

Ο λογικός τελεστής NOT εκτελεί την πράξη της άρνησης 2 πράξεων. Δέχεται μια λογική έκφραση και παράγει αποτέλεσμα ΠΑΝΤΑ την αντίθετη τιμή της έκφρασης.

6. Ποια είναι η προτεραιότητα μεταξύ όλων των πράξεων/τελεστών στην Python.

ΠΡΟΤΕΡΑΙΟΤΗΤΑ ΤΕΛΕΣΤΩΝ

Αριθμητικοί > Σχεσιακοί > Λογικοί

ΠΡΟΤΕΡΑΙΟΤΗΤΑ ΑΡΙΘΜΗΤΙΚΩΝ ΠΡΑΞΕΩΝ

1. Ύψωση σε δύναμη
2. Πολλαπλασιασμός, διαίρεση, MOD (%), DIV (//)
3. Πρόσθεση, αφαίρεση

ΠΡΟΤΕΡΑΙΟΤΗΤΑ ΛΟΓΙΚΩΝ ΠΡΑΞΕΩΝ

1. NOT
2. AND
3. OR

- Όταν σε μια έκφραση υπάρχουν τελεστές ίδιας προτεραιότητας, τότε οι πράξεις εκτελούνται από αριστερά προς τα δεξιά.
- Για να αλλάξουμε τη σειρά της προτεραιότητας χρησιμοποιούμε παρενθέσεις οι οποίες εκτελούνται ΠΑΝΤΑ ΠΡΩΤΕΣ.

7. Τι είναι εκφράσεις.

Είναι «προτάσεις» που σχηματίζονται από σταθερές, μεταβλητές, συναρτήσεις, τελεστές και παρενθέσεις. Κάθε έκφραση έχει ένα αποτέλεσμα που είναι η τιμή της. Η τιμή μιας έκφρασης προκύπτει όταν αντικαταστήσουμε τις μεταβλητές με τις τιμές τους και εκτελέσουμε τις πράξεις σύμφωνα με την προτεραιότητα των τελεστών.

Υπάρχουν 3 κατηγορίες εκφράσεων:

Αριθμητικές εκφράσεις: Υπολογίζουν αριθμητικές παραστάσεις με τη χρήση αριθμητικών τελεστών και παράγουν μια αριθμητική τιμή.

Λογικές εκφράσεις: Απεικονίζουν απλές παραστάσεις με τη χρήση σχεσιακών τελεστών που το αποτέλεσμά τους είναι μια λογική τιμή.

Σύνθετες εκφράσεις: Δημιουργούνται από το συνδυασμό 2 ή περισσότερων λογικών εκφράσεων και το αποτέλεσμά τους είναι μια λογική τιμή.

8. Τι είναι μια μεταβλητή. (και σελ 35)

Μια μεταβλητή χρησιμοποιείται για να αποθηκεύσουμε ένα δεδομένο. Αντιστοιχεί σε μια θέση μνήμης, στην οποία κάθε φορά μπορεί να αποθηκευτεί μια μόνο τρέχουσα τιμή και μπορεί να αντικατασταθεί με μια άλλη.

Κάθε μεταβλητή έχει ένα όνομα και μια τιμή. Ο τύπος της μεταβλητής είναι ίδιος με τον τύπο του δεδομένου το οποίο αποθηκεύεται στη μεταβλητή αυτή.

9. Τι ονομάζουμε «αφηρημένο τύπο δεδομένων». (ή σελ 44)

Όταν σχεδιάζεται ένας τύπος δεδομένων, αρχικά υπάρχει μόνο διανοητικά και ορίζεται μόνο με βάση τις λειτουργίες που επιτελεί χωρίς να μας απασχολεί ο τρόπος που θα αναπαρασταθούν τα δεδομένα αλλά ούτε και ο τρόπος που θα υλοποιηθούν οι λειτουργίες σε κώδικα. Μα ενδιαφέρει, δηλαδή, το τι θα υπολογιστεί και όχι το πώς. Με αυτό τον τρόπο ορίζεται ένας αφηρημένος τύπος δεδομένων.

10. Να περιγράψετε τη λειτουργία της εντολής εξόδου print.

Η εντολή print χρησιμοποιείται για να εμφανίσουμε τα αποτελέσματα του προγράμματός μας στην οθόνη. Είναι ένας από τους τρόπους που χαρακτηρίζουν την έξοδο του προγράμματός μας.

Για την εμφάνιση των αποτελεσμάτων γράφεται η εντολή print και μπορεί να ακολουθεί κάποιο από τα παρακάτω στοιχεία:

- **Μια ή περισσότερες μεταβλητές**, χωρισμένες με κόμμα μεταξύ τους, η τιμή των οποίων θα εμφανιστεί στην οθόνη.
- **Ένας αριθμός**, ο οποίος εμφανίζεται στην οθόνη.
- **Μια συμβολοσειρά**, η οποία εμφανίζεται στην οθόνη.
- **Μια έκφραση**, η οποία υπολογίζεται και το αποτέλεσμα εμφανίζεται στην οθόνη.
- **Συνδυασμός των παραπάνω**, χωρισμένα με κόμμα μεταξύ τους, τα οποία εμφανίζονται σε μία σειρά.

Μια εντολή print χωρίς κανένα στοιχείο από τα παραπάνω, εμφανίζει μια κενή γραμμή στην οθόνη.

11. Να περιγράψετε τη λειτουργία της εντολής εισόδου input και raw_input.

Οι εντολές **input** και **raw_input** χρησιμοποιούνται για να εισαχθεί μια τιμή από το πληκτρολόγιο, η οποία θα αποδοθεί σε μια μεταβλητή.

Κατά την εκτέλεση των εντολών αυτών, διακόπτεται προσωρινά το πρόγραμμα και περιμένει από το χρήστη να εισάγει μία τιμή από το πληκτρολόγιο. Η τιμή που θα δοθεί αποδίδεται αυτόματα στη μεταβλητή που βρίσκεται πριν από το =. Είναι ένας από τους τρόπους που χαρακτηρίζουν την είσοδο του προγράμματός μας. Το «μήνυμα» στις παρενθέσεις δεν είναι υποχρεωτικό να υπάρχει, αν υπάρχει, όμως εμφανίζεται στην οθόνη πριν πληκτρολογηθεί η τιμή.

Η διαφορά ανάμεσα στις δυο εντολές βρίσκεται στον τύπο δεδομένων που αναμένει το πρόγραμμα να εισαχθούν. Η εντολή raw_input() διαβάζει μόνο αλφαριθμητικά δεδομένα, ενώ η εντολή input() αριθμητικά ή λογικά δεδομένα.

12. Τι ονομάζουμε αλγοριθμική δομή.

Με τον όρο αλγοριθμική δομή εννοούμε τον τρόπο με τον οποίο εκτελούνται οι εντολές που υπάρχουν σε ένα αλγόριθμο (πρόγραμμα).

13. Να αναφέρετε ποιες βασικές αλγοριθμικές δομές χρησιμοποιούνται για την ανάπτυξη ενός προγράμματος.

Οι αλγοριθμικές δομές που χρησιμοποιούνται σε ένα αλγόριθμο είναι:

- Η δομή ακολουθίας
- Η δομή επιλογής
- Η δομή επανάληψης

14. Με ποια εντολή υλοποιείται η δομή επιλογής στην Python. (ή σελ 46)

Η δομή επιλογής υλοποιείται με την εντολή **if**, η οποία μπορεί να χρησιμοποιηθεί όταν:

- Υπάρχει μόνο μια επιλογή (Απλή δομή επιλογής) και έχει τη μορφή:

  if <συνθήκη>:
  εντολές

- Υπάρχουν δυο επιλογές (Σύνθετη δομή επιλογής) και έχει τη μορφή:
  if <συνθήκη>:
  εντολές
  else:
  εντολές

- Υπάρχουν περισσότερες από δυο επιλογές (Πολλαπλή δομή επιλογής) και έχει τη μορφή:
  if <συνθήκη>:
  εντολές
  elif <συνθήκη>:
  εντολές
  ……
  else:
  εντολές

15. Πότε χρησιμοποιείται η εμφωλευμένη δομή επιλογής.

Η εμφωλευμένη δομή επιλογής χρησιμοποιείται όταν οι περιπτώσεις επιλογής είναι περισσότερες από δύο.

16. Με ποιον τρόπο υλοποιείται η εμφωλευμένη επιλογή στην Python.

Η εμφωλευμένη δομή επιλογής υλοποιείται στην Python τοποθετώντας εντολές **if….else** μέσα σε άλλες εντολές **if…else**.

17. Πότε χρησιμοποιείται η εντολή for. Να γράψετε τον τρόπο χρήσης της εντολής for στην Python και να περιγράψετε τη λειτουργία της.

Η εντολή **for** χρησιμοποιείται όταν θέλουμε ένα σύνολο εντολών να εκτελεστεί προκαθορισμένες φορές, δηλαδή ο αριθμός των επαναλήψεων να είναι γνωστός πριν από την έναρξη της επανάληψης.

Η εντολή **for** εκτελείται τόσες φορές όσο είναι το πλήθος των αριθμών που παράγει η συνάρτηση range(). Η μεταβλητή της for στην πρώτη επανάληψη λαμβάνει τον πρώτο αριθμό από τη λίστα των αριθμών που παράγει η range() και σε κάθε νέα επανάληψη λαμβάνει τον επόμενο από τη λίστα των αριθμών αυτών.

18. Τι είναι η εμφωλευμένη δομή επανάληψης.

Η εμφωλευμένη δομή επανάληψης δημιουργείται όταν μια εντολή επανάληψης βρίσκεται μέσα σε μια άλλη. Πολλές φορές ονομάζεται και ως «εμφωλευμένοι βρόγχοι». Η εντολή επανάληψης που βρίσκεται μέσα στην άλλη δεν έχει σημασία. Μπορεί να μια for να βρίσκεται μέσα σε μία while ή να εμφωλεύεται οποιοσδήποτε άλλος συνδυασμός των εντολών επανάληψης.

19. Τι είναι ένα υποπρόγραμμα.

Ένα υποπρόγραμμα είναι ένα κομμάτι προγράμματος που έχει γραφεί ξεχωριστά από το υπόλοιπο πρόγραμμα και επιτελεί ένα αυτόνομο έργο.

20. Να γράψετε τον τρόπο χρήσης της εντολής return στην Python και να περιγράψετε τη λειτουργία της.

Η εντολή return χρησιμοποιείται για την επιστροφή τιμής από μια συνάρτηση. Η σύνταξη της εντολής είναι:

return <αποτέλεσμα>

Αν κατά τη ροή εκτέλεσης των εντολών μιας συνάρτησης, βρεθεί μια εντολή return, τότε διακόπτετε η εκτέλεση της συνάρτησης και ο έλεγχος ροής επιστρέφει από τη συνάρτηση στο σημείο του προγράμματος που αυτή κλήθηκε, χρησιμοποιώντας την τιμή του αποτελέσματος ως επιστρεφόμενη τιμή. Η εντολή return δεν είναι υποχρεωτικό να υπάρχει σε κάθε συνάρτηση.

21. Γιατί είναι σημαντική η δομή δεδομένων.

Η επιλογή της κατάλληλης δομής δεδομένων μπορεί να επιφέρει μεγάλα οφέλη στην ταχύτητα εκτέλεσης των υπολογισμών, άρα και στην ταχύτητα εκτέλεσης ενός προγράμματος.

22. Ποιες κατηγορίες δομών δεδομένων υπάρχουν.

Οι δομές δεδομένων χωρίζονται σε 2 μεγάλες κατηγορίες:

i. Τις Μη μεταβαλλόμενες (ή στατικές – immutable) δομές δεδομένων.
ii. Τις δυναμικές δομές δεδομένων (mutabe).

23. Ποια είναι τα χαρακτηριστικά των μη μεταβαλλόμενων (ή στατικών – immutable) δομών δεδομένων. (ή σελ 127)

Οι μη μεταβαλλόμενες δομές δεδομένων έχουν σταθερό μέγεθος που καθορίζεται κατά τη συγγραφή του προγράμματος και για την αναπαράσταση της δομής η μνήμη που δεσμεύεται είναι σταθερή. Κατά συνέπεια τα στοιχεία της δομής είναι μη μεταβαλλόμενα στοιχεία. Αυτό σημαίνει ότι δεν μπορούμε να αφαιρέσουμε ή να προσθέσουμε αντικείμενα σε αυτές, κατά τη διάρκεια εκτέλεσης του προγράμματος.

24. Ποια είναι τα χαρακτηριστικά των δυναμικών (mutabe) δομών δεδομένων. (ή σελ 130)

Οι δυναμικές δομές δεδομένων στηρίζονται στη λειτουργία της «δυναμικής παραχώρησης μνήμης», κατά την οποία το πρόγραμμα μπορεί να ζητήσει όση μνήμη απαιτείται για τη δημιουργία και επεξεργασία της δομής κατά την εκτέλεση του προγράμματος. Δεν έχουν σταθερό μέγεθος αλλά είναι δυνατό να προσθέτουμε και να αφαιρούμε στοιχεία από αυτές κατά τη διάρκεια εκτέλεσης του προγράμματος.

25. Ποιες στατικές δομές δεδομένων υπάρχουν στην Python.

Στην Python στατικές δομές είναι οι συμβολοσειρές και οι πλειάδες.

26. Ποιες δυναμικές δομές δεδομένων υπάρχουν στην Python.

Στην Python δυναμικές δομές είναι οι λίστες, η στοίβα, η ουρά, τα λεξικά, οι γράφοι και τα δέντρα.

27. Γιατί οι συμβολοσειρές ανήκουν στις στατικές δομές δεδομένων.

Οι συμβολοσειρές έχουν σταθερό μέγεθος. Δεν μπορούμε να προσθέσουμε ή να αφαιρέσουμε χαρακτήρες σε μια συμβολοσειρά από τη στιγμή που θα δημιουργηθεί, ούτε να τροποποιήσουμε το περιεχόμενό της. Μπορούμε μόνο να τη δημιουργήσουμε ξανά με τις αλλαγές που επιθυμούμε.

28. Πώς αποτυπώνεται μία συμβολοσειρά. (ή σελ 127)

Κάθε χαρακτήρας μιας συμβολοσειράς απαριθμείται με ένα αριθμό, με τον πρώτο χαρακτήρα να ξεκινάει με το 0. Η προσπέλαση σε κάθε χαρακτήρα (στοιχείο) της συμβολοσειράς γίνεται γράφοντας το όνομα της μεταβλητής και μέσα σε αγκύλες τον αριθμό της θέσης που αντιστοιχεί στον χαρακτήρα. Ο αριθμός μέσα στην αγκύλη ονομάζεται δείκτης και δηλώνει τη θέση του στοιχείου σε σχέση με τη συμβολοσειρά. Είναι δυνατό να έχουμε και αρνητικούς δείκτες οι οποίοι ξεκινούν από το τέλος της συμβολοσειράς με τον τελευταίο χαρακτήρα να λαμβάνει την τιμή -1.

Αν έχουμε μια μεταβλητή w στην οποία αποθηκεύουμε τη λέξη “ΠΑΝΕΛΛΑΔΙΚΕΣ”, τότε σχηματικά θα είχαμε την εξής αναπαράσταση (αποτύπωση):

```
w = “ΠΑΝΕΛΛΑΔΙΚΕΣ”
```

Χαρακτήρας	w[0]	w[1]	w[2]	w[3]	w[4]	w[5]	w[6]	w[7]	w[8]	w[9]	w[10]	w[11]
Δείκτης	0	1	2	3	4	5	6	7	8	9	10	11
Συμβολοσειρά	**Π**	**Α**	**Ν**	**Ε**	**Λ**	**Λ**	**Α**	**Δ**	**Ι**	**Κ**	**Ε**	**Σ**
Δείκτης	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1
Χαρακτήρας	w[-12]	w[-11]	w[-10]	w[-9]	w[-8]	w[-7]	w[-6]	w[-5]	w[-4]	w[-3]	w[-2]	w[-1]

<u>ΠΡΟΣΟΧΗ</u>: Ο δείκτης δηλώνει τη θέση του στοιχείου σε σχέση με τη συμβολοσειρά, η οποία θέση είναι διαφορετική από τη θέση του χαρακτήρα μέσα στη συμβολοσειρά. Ο χαρακτήρας “Δ” είναι ο 8^{ος} χαρακτήρας της λέξης “ΠΑΝΕΛΛΑΔΙΚΕΣ”, αλλά στη συμβολοσειρά w για να αναφερθούμε στον χαρακτήρα αυτό θα χρησιμοποιήσουμε το δείκτη 7 (ή το -5).

29. Γιατί η λίστα είναι μία δυναμική δομή δεδομένων. (ή σελ 130)

Η λίστα είναι μια δυναμική δομή δεδομένων γιατί δεν έχει σταθερό μέγεθος και μπορούμε να αφαιρούμε ή να προσθέτουμε στοιχεία σε αυτή όποτε χρειαζόμαστε (με τη χρήση των κατάλληλων μεθόδων).

30. Πώς αποτυπώνεται μία λίστα. (ή σελ 130)

Η λίστα είναι μια διατεταγμένη ακολουθία δεδομένων. Η αναπαράσταση μιας λίστας μοιάζει αρκετά με την συμβολοσειρά, με τη διαφορά ότι τα στοιχεία μιας λίστας μπορούν να είναι και διαφορετικού τύπου και για αυτό χαρακτηρίζονται ως αντικείμενα.

Κάθε αντικείμενο μιας λίστας χαρακτηρίζεται από ένα μοναδικό αύξοντα αριθμό, ο οποίος ορίζει τη θέση του μέσα στη λίστα, με το πρώτο αντικείμενο να ξεκινάει με το 0. Η προσπέλαση σε κάθε αντικείμενο της λίστας, όπως και στις συμβολοσειρές, γίνεται γράφοντας το όνομα της μεταβλητής και μέσα σε αγκύλες τον αριθμό της θέσης που αντιστοιχεί στο αντικείμενο. Ο αριθμός μέσα στην αγκύλη που δηλώνει τη θέση του στοιχείου ονομάζεται δείκτης. Είναι δυνατό να έχουμε και αρνητικούς

δείκτες οι οποίοι ξεκινούν από το τέλος της λίστας με το τελευταίο αντικείμενο να λαμβάνει την τιμή -1.

Αν έχουμε μια μεταβλητή nums στην οποία αποθηκεύουμε μια λίστα αριθμών, όπως παρακάτω, τότε σχηματικά θα είχαμε την εξής αναπαράσταση (αποτύπωση):

```
nums = [3,6,2,7,9,1]
```

Αντικείμενο	nums[0]	nums[1]	nums[2]	nums[3]	nums[4]	nums[5]
Δείκτης	0	1	2	3	4	5
Λίστα	**3**	**6**	**2**	**7**	**9**	**1**
Δείκτης	-6	-5	-4	-3	-2	-1
Αντικείμενο	nums[-6]	nums[-5]	nums[-4]	nums[-3]	nums[-2]	nums[-1]

<u>ΠΡΟΣΟΧΗ</u>: Ο δείκτης δηλώνει τη θέση του αντικειμένου, η οποία θέση είναι διαφορετική από τη θέση του στοιχείου μέσα στη λίστα. Ο αριθμός 9 είναι ο 5^{ος} αριθμός μέσα στη λίστα, αλλά στη λίστα nums για να αναφερθούμε στον αριθμό αυτό αυτό θα χρησιμοποιήσουμε το δείκτη 4 (ή το -2).

31. Τι είναι η άδεια λίστα.

Η άδεια λίστα είναι μια λίστα η οποία δεν περιέχει στοιχεία μέσα της και συμβολίζεται με 2 άδειες αγκύλες [ ]. Πχ lista=[ ]

Αν θέλουμε να ελέγξουμε αν μια λίστα είναι κενή υπάρχουν 2 τρόποι:

i. Να ελέγξουμε απευθείας αν η λίστα είναι κενή: `if L == [ ]:`
ii. Να ελέγξουμε αν το μέγεθος της λίστας είναι ίσο με το 0: `if len(L) == 0:`

BONUS ΕΡΩΤΗΣΕΙΣ

32. Να γράψετε μια συνάρτηση σε Python η οποία θα δέχεται μια λίστα, θα ελέγχει αν τα στοιχεία της είναι σε αύξουσα σειρά και θα επιστρέφει αντίστοιχα True ή False. (Δραστηριότητα 3 σελ 86)

Τρόπος Α

```
def check(lista):
    n=len(lista)
    taxinom=True
    for j in range(1,n-1):
        if lista[j]<lista[j-1]:
            taxinom=False
    return taxinom
```

Τρόπος Β

```
def check (lista):
    taxinom = True
    i = 0
    N = len(lista)-1
    while taxinom and i<N :
        if lista[i] > lista[i+1]:
          taxinom = False
        i = i +1
    return taxinom
```

33. Να αναπτύξετε τη βελτιωμένη έκδοση του αλγόριθμου ταξινόμησης ευθείας ανταλλαγής η οποία τερματίζει , όταν διαπιστώσει ότι η λίστα είναι ταξινομημένη, ώστε να αποφεύγονται οι περιττές συγκρίσεις. (Δραστηριότητα 4 σελ 86)

Έξυπνη Φυσαλίδα

```
def smartbubblesort(lista):
    n=len(lista)
    isSorted=False
    i=0
    while i<n and isSorted==False:   #Εναλλακτικά → not isSorted
        isSorted=True
        for j in range(n-1,i,-1):
            if lista[j]<lista[j-1]:
                lista[j],lista[j-1]=lista[j-1],lista[j]
                isSorted=False
        i=i+1
```