Τελεστές – Μεταβλητές – Τύποι δεδομένων – Δομή προγράμματος

1. Ο αριθμός 23 είναι ένα δεδομένο πραγματικού τύπου.
2. Η τιμή True αντιστοιχεί σε δεδομένο τύπου χαρακτήρα.
3. Η τιμή μιας μεταβλητής μπορεί να αλλάζει κατά τη διάρκεια της εκτέλεσης ενός προγράμματος.
4. Τα είδη των μεταβλητών που χρησιμοποιούμε είναι οι αριθμητικές, οι αλφαριθμητικές, οι λογικές και οι σταθερές.
5. Μια λογική έκφραση μπορεί να περιέχει αριθμητικές παραστάσεις, συγκριτικούς και λογικούς τελεστές καθώς και παρενθέσεις.
6. Όταν αριθμητικοί και συγκριτικοί τελεστές συνδυάζονται σε μια έκφραση, οι αριθμητικές πράξεις εκτελούνται πρώτες.
7. Η λογική έκφραση ((19-9)>=10) Η ((-13+3) > -8) and (8%2 <> 0) είναι False.
8. Το +, -, *, /, = είναι αριθμητικοί τελεστές και το >, < συγκριτικοί τελεστές.
9. Ο τελεστής / παράγει το πηλίκο της ακέραιος διαίρεσης δύο ακέραιων αριθμών.
10. Το όνομα μιας μεταβλητής μπορεί να μεταβάλλεται σε ένα πρόγραμμα.
11. Το αποτέλεσμα της σύγκρισης 'a' > 'b' παράγει πάντα μία λογική τιμή.
12. Ο τελεστής and, όταν εφαρμόζεται σε δύο λογικές εκφράσεις για να παράγει αποτέλεσμα True, αρκεί μία από τις δύο εκφράσεις να είναι True.
13. Ο τελεστής or αντιστοιχεί στην πράξη της διάζευξης δύο λογικών εκφράσεων.
14. Ο τελεστής της ισότητας « == » είναι ένας λογικός τελεστής.
15. Σε μία αριθμητική έκφραση ο τελεστής % έχει υψηλότερη προτεραιότητα από τον τελεστή /.
16. Μεταξύ των λογικών τελεστών, ο τελεστής and έχει την υψηλότερη προτεραιότητα.
17. Η τιμή 20 % 3 είναι πραγματικού τύπου.
18. Το όνομα float είναι ένα αποδεκτό όνομα μεταβλητής.
19. Σε μία έκφραση οι λογικοί τελεστές έχουν μικρότερη προτεραιότητα από τους αριθμητικούς τελεστές.
20. Οι λογικοί τελεστές and, or, not, της Python έχουν την ίδια προτεραιότητα στις πράξεις.
21. Ο integer τύπος δεδομένων ανήκει στους απλούς τύπους.
22. Ο λογικός τύπος δεδομένων αποτελείται από αδιαίρετα στοιχεία.
23. Ο τύπος δεδομένων «χαρακτήρας» ανήκει στους σύνθετους τύπους.
24. Ο τύπος δεδομένων «αλφαριθμητικός-string» είναι απλός.
25. Η τιμή «ΔΕΥΤΕΡΑ» είναι ένα δεδομένο απλού τύπου.
26. Η τιμή 13.34 είναι ένα δεδομένο σύνθετου τύπου γιατί περιέχει δύο αριθμούς.
27. Το «Δέντρο» και ο «Γράφος» ανήκουν στους απλούς τύπους δεδομένων.
28. Η δομή δεδομένων «Πίνακας» ανήκει στους σύνθετους τύπους δεδομένων.
29. Τα «Αρχεία» είναι ένας σύνθετος τύπος δεδομένων.
30. Τα «Σύνολα» και οι «Εγγραφές» είναι σύνθετοι τύποι δεδομένων.
31. Η συνάρτηση int(x) επιστρέφει τον ακέραιο αριθμό που αντιστοιχεί στον χαρακτήρα x.
32. Η συνάρτηση float() μετατρέπει ακεραίους και συμβολοσειρές σε δεκαδικούς αριθμούς.
33. Όταν η συνάρτηση str() δέχεται μία οποιαδήποτε τιμή, τότε τη μετατρέπει σε συμβολοσειρά.
34. Η συνάρτηση abs() δέχεται πάντα χαρακτήρα και επιστρέφει τον επόμενο αριθμό που αντιστοιχεί σε αυτόν.

35. Η συνάρτηση pow(x,y) επιστρέφει τη δύναμη y^x.
36. Η συνάρτηση divmod(x,y) επιστρέφει το ακέραιο πηλίκο και το ακέραιο υπόλοιπο της διαίρεσης x/y.
37. Η εντολή εκχώρηση της Python συμβολίζεται με το σύμβολο « == ».
38. Αριστερά από τον τελεστή εκχώρησης μπορεί να υπάρχει και μία έκφραση.
39. Η εντολή type χρησιμοποιείται για να ελέγξουμε τον τύπο ενός δεδομένου.
40. Η εντολή εκχώρησης μπορεί να χρησιμοποιηθεί πολλές φορές σε ένα πρόγραμμα Python.
41. Το «type 'str'» αναφέρεται σε ένα δεδομένο ακεραίου τύπου.
42. Ένα πρόγραμμα γραμμένο σε Python παράγει έξοδο με την εντολή print.
43. Σε ένα πρόγραμμα Python η εντολή input εμφανίζεται υποχρεωτικά μία μόνο φορά.
44. Η λειτουργία της εντολής input είναι ακριβώς ίδια με αυτήν της εντολής raw_input.
45. Κάθε μεταβλητή παίρνει τιμή μόνο με την εντολή input.
46. Κατά την εκτέλεση του προγράμματος η εντολή input διακόπτει την εκτέλεσή του και περιμένει την εισαγωγή τιμών από το πληκτρολόγιο.
47. Ο χαρακτήρας « ! » χρησιμοποιείται για να εισάγουμε σχόλια σε ένα πρόγραμμα python.
48. Η διαίρεση « / » έχει μεγαλύτερη προτεραιότητα από το ακέραιο υπόλοιπο « % » και το ακέραιο πηλίκο « // ».
49. Η Λογική έκφραση "ΣΟΦΙΑ" > "ΣΟΦΟΣ" είναι True.
50. Το όνομα μια μεταβλητής πρέπει να ξεκινάει πάντα με γράμμα.
51. Ο αριθμητικός τελεστής « + » έχει μεγαλύτερη προτεραιότητα από το συγκριτικό « == ».
52. Σε μια εντολή εκχώρησης δεν μπορεί να χρησιμοποιηθεί η ίδια μεταβλητή τόσο στο αριστερό όσο και στο δεξιό μέλος της.
53. Το αποτέλεσμα της έκφρασης X*Y/2 είναι ίδιο με το αποτέλεσμα της έκφρασης X* (Y/2).
54. Μια λογική έκφραση μπορεί να περιέχει αριθμητικές παραστάσεις, συγκριτικούς και λογικούς τελεστές καθώς και παρενθέσεις.
55. Όταν δύο λογικές συνθήκες είναι αληθείς, τότε και η σύζευξή τους είναι True.
56. Η διάζευξη δύο λογικών συνθηκών είναι True όταν μία τουλάχιστον από τις δύο συνθήκες είναι True.
57. Όταν δύο λογικές συνθήκες έχουν την ίδια τιμή, τότε η διάζευξή τους είναι πάντα True.
58. Όταν αριθμητικοί και συγκριτικοί τελεστές συνδυάζονται σε μια έκφραση, οι αριθμητικές πράξεις εκτελούνται πρώτες.
59. Αν το A έχει την τιμή 10 και το B έχει την τιμή 20, τότε η λογική έκφραση « (A > 8 and B < 20) or (A > 10 or B == 10) » είναι True.
60. Αν το A έχει την τιμή 5 και το B την τιμή 6, τότε η λογική έκφραση « (A > 5 or A < 3) and (B > 5) » είναι False.
61. Η συνθήκη « x % y == 0 » ελέγχει εάν το x είναι πολλαπλάσιο του y.
62. Η συνθήκη « abs(x) == x » ελέγχει εάν το x είναι ακέραιος.
63. Η συνθήκη « x % 2 == 0 » ελέγχει εάν το x είναι άρτιος.
64. Ο τελεστής and αντιστοιχεί στη λογική πράξη της σύζευξης.
65. Σε μια λογική έκφραση, οι συγκριτικοί τελεστές έχουν χαμηλότερη ιεραρχία από τους λογικούς τελεστές.

66. Αν κατά την εκτέλεση της εντολής « x=input() » δεν εισαχθούν δεδομένα από το πληκτρολόγιο, τότε το πρόγραμμα τερματίζεται.
67. Η πρώτη γραμμή ενός προγράμματος Python είναι υποχρεωτικά ο τίτλος του προγράμματος.
68. Οι δηλώσεις των μεταβλητών ενός προγράμματος Python γίνονται μετά τον τίτλο του προγράμματος.
69. Τα σχόλια είναι υποχρεωτικά σε ένα πρόγραμμα Python.
70. Τα σχόλια γράφονται από τον προγραμματιστή για να βοηθήσουν στην κατανόηση τμημάτων εντολών.
71. Η εντολή «print x» έχει το ίδιο αποτέλεσμα με την εντολή «Print x».
72. Η εντολή «print 'x'» εμφανίζει το περιεχόμενο της μεταβλητής x.
73. Η μαθηματική έκφραση « A < X < B » κωδικοποιείται ως « (X > A) and (X < B) ».
74. Για τη διάταξη των αλφαριθμητικών στοιχείων χρησιμοποιείται η αλφαβητική σειρά.

Δομή ακολουθίας

1. Στη δομή ακολουθίας εκτελούνται υποχρεωτικά όλες οι εντολές.
2. Ένα πρόγραμμα σε Python υποχρεωτικά αρχίζει έχοντας ως σχόλια την πρώτη του γραμμή.
3. Ένα σύνολο διαδοχικών εντολών που εκτελούνται με τη σειρά που είναι γραμμένες αποτελούν τη δομή ακολουθίας.
4. Σε ένα πρόγραμμα μπορεί να υπάρχουν εντολές που δεν είναι σωστά διατυπωμένες και δεν εκτελούνται.
5. Στη δομή ακολουθίας υποχρεωτικά πριν από την εντολή input πρέπει να υπάρχει μία εντολή print.
6. Η μοναδική εντολή εισόδου που χρησιμοποιείται στη δομή ακολουθίας είναι η εντολή input.
7. Η εντολή εκχώρησης μπορεί να χρησιμοποιηθεί πολλές φορές σε ένα διάγραμμα ροής.
8. Κατά την εκτέλεση ενός προγράμματος σε Python που χρησιμοποιεί μόνο τη δομή ακολουθίας, ορισμένες εντολές μπορεί να μην εκτελεστούν ποτέ.
9. Ένα σύνθετο πρόβλημα που απαιτεί λήψη αποφάσεων και επαναλήψεων λύνεται αλγοριθμικά με τη δομή ακολουθίας.
10. Η δομή ακολουθίας χρησιμοποιείται για την αντιμετώπιση απλών προβλημάτων, στα οποία είναι δεδομένη η σειρά εκτέλεσης ενός συνόλου ενεργειών.
11. Σε ένα πρόγραμμα Python μπορούμε να χρησιμοποιήσουμε μεταβλητές χωρίς να εμφανίσουμε το περιεχόμενό τους.
12. Στη δομή ακολουθίας μια συγκεκριμένη εντολή μπορεί να εκτελεστεί πολλές φορές.

Δομή επιλογής

1. Η δομή επιλογής είναι μία από τις βασικές αλγοριθμικές δομές.
2. Με τις εντολές επιλογής μπορούμε να αλλάξουμε τη διαδοχική σειρά εκτέλεση των εντολών σε ένα πρόγραμμα.
3. Αν θέλουμε να βρούμε τον μεγαλύτερο από τρεις αριθμούς, υποχρεωτικά πρέπει να χρησιμοποιήσουμε τη δομή επιλογής.
4. Στην εντολή επιλογής «if Λογική έκφραση:» η λογική έκφραση μπορεί να λάβει και άλλες τιμές εκτός από τις τιμές True και False.
5. Στην εντολή επιλογής «if Λογική έκφραση: εντολή-1» αν η λογική έκφραση είναι False, τότε εκτελείται η εντολή-1.
6. Στη δομή επιλογής, μπορεί μία ή περισσότερες εντολές να μην εκτελεστούν ποτέ.
7. Για την εύρεση της μέσης θερμοκρασίας πέντε θερμοκρασιών πρέπει να χρησιμοποιηθεί η δομή επιλογής.
8. Αν A = 2, τότε εκτελείται η εντολή-1 στην επιλογή «if A > 2: εντολή-1 else: εντολή-2».
9. Στην εντολή «if x % y == 0 : εντολή-1» η εντολή-1 εκτελείται όταν το x είναι πολλαπλάσιο του y.
10. Στην εντολή επιλογής «if Λογική έκφραση: εντολή-1 else: εντολή-2» η εντολή-2 εκτελείται, όταν η λογική έκφραση είναι False.
11. Έστω η μεταβλητή y είναι λογική. Η εντολή «print x» της δομής επιλογής «if y or not(y): print x» εκτελείται πάντα.
12. Η δομή της πολλαπλής επιλογής περιλαμβάνει τον έλεγχο μιας ίδιας συνθήκης πολλές φορές.
13. Η πολλαπλή επιλογή χρησιμοποιείται σε προβλήματα, στα οποία εκτελούνται πάντα οι ίδιες εντολές ανάλογα με την τιμή που λαμβάνει μια συνθήκη.
14. Στην εντολή επιλογής «elif» η περίπτωση «else», είναι δυνατόν να υπάρχει δύο φορές.
15. Στην εντολή επιλογής «elif» μπορεί μία ή περισσότερες εντολές να μην εκτελεστούν ποτέ.
16. Η κάθε συνθήκη που ελέγχει η εντολή «elif», είναι μια λογική έκφραση.
17. Η εντολή «elif» μπορεί να εξετάσει και σύνθετες συνθήκες.
18. Όταν η πρώτη και η τρίτη συνθήκη της εντολής «elif» είναι αληθείς, τότε εκτελούνται οι εντολές που αντιστοιχούν στην τρίτη συνθήκη.
19. Στην εντολή «elif» υπάρχει περιορισμός στον αριθμό των συνθηκών που ελέγχουμε.
20. Οι εντολές που υπάρχουν στην περίπτωση «else» της πολλαπλής επιλογής, εκτελούνται, όταν όλες οι συνθήκες που εξετάζονται είναι ψευδείς.
21. Όταν μία εντολή «if» υπάρχει μέσα σε μία άλλη εντολή «if», τότε αυτή εκτελείται, όταν η συνθήκη της εξωτερικής «if» είναι ψευδής.
22. 0 αριθμός των εντολών «if» που χρησιμοποιούμε για να δημιουργήσουμε εμφωλευμένες if, είναι συγκεκριμένος και δεν μπορούμε να τον υπερβούμε.
23. Στην εμφωλευμένη δομή επιλογής, κάποιες εντολές που υπάρχουν στο εσωτερικό αυτών μπορεί να μην εκτελεστούν ποτέ.
24. Στις εμφωλευμένες εντολές «if» εκτελούνται μόνο οι εντολές που αντιστοιχούν, στις περιπτώσεις που οι συνθήκες των εντολών «if» είναι αληθείς.
25. Χρησιμοποιώντας εμφωλευμένες εντολές «if» μπορούμε να έχουμε περισσότερες από δύο επιλογές.
26. Μία εντολή «if» δεν μπορεί να τοποθετηθεί στο εσωτερικό μιας εντολής «if.. .else».

27. Την «if λογική συνθήκη-1 and λογική συνθήκη-2:» μπορούμε να την αντικαταστήσουμε με τις «if λογική συνθήκη-1:» και «if λογική συνθήκη-2:», με τη δεύτερη να υπάρχει στο εσωτερικό της πρώτης.
28. Όταν πρέπει να εκτελεστούν κάποιες εντολές υπό κάποια συνθήκη, χρησιμοποιείται η δομή επιλογής.
29. Στη δομή επιλογής υπάρχει περίπτωση κάποιες εντολές να μην εκτελεστούν ποτέ.
30. Η συνθήκη που ελέγχεται σε μια δομή επιλογής μπορεί να πάρει περισσότερες από δύο διαφορετικές τιμές.
31. Κάθε εντολή if περιέχει και το else.
32. Η δομή επιλογής αντικαθιστά τη δομή ακολουθίας ελαττώνοντας αισθητά το πλήθος των εντολών ενός αλγορίθμου
33. Η συνθήκη που ελέγχεται σε μια δομή επιλογής μπορεί να πάρει περισσότερες από δύο διαφορετικές τιμές.
34. Μέσα σε μια δομή επιλογής δεν μπορούν να τοποθετηθούν σχόλια.
35. Για την εύρεση του μεγίστου μεταξύ δύο αριθμών χρησιμοποιούμε απαραίτητα τη δομή επιλογής.

Δομή επανάληψης

1. Η συνάρτηση «range» παράγει μια λίστα αριθμών.
2. Η συνάρτηση «range» μπορεί να παράγει και πραγματικούς αριθμούς.
3. Όταν στο εσωτερικό των παρενθέσεων της «range» υπάρχουν μόνο δύο αριθμοί, τότε το βήμα μεταβολής είναι μηδέν.
4. Η συνάρτηση «range(5,1)» παράγει τη λίστα των αριθμών [1,2,3,4].
5. Όταν Α = Β, η συνάρτηση range(Α, Β) δεν παράγει αριθμούς.
6. Η range(-5, -1) και η range(-5, -1, 1) παράγουν την ίδια λίστα αριθμών.
7. Η συνάρτηση range(2,10, -1) παράγει μια κενή λίστα αριθμών.
8. Με τη συνάρτηση «range» μπορούμε να παράγουμε και μία λίστα αρνητικών αριθμών.
9. Οι συναρτήσεις «range(6)» και «range(1,6)» παράγουν την ίδια λίστα αριθμών.
10. Η δομή επανάληψης χρησιμοποιεί ένα σύνολο εντολών που εκτελούνται μία μόνο φορά.
11. Η εντολή «for» μπορεί να χρησιμοποιηθεί, όταν έχουμε άγνωστο αριθμό επαναλήψεων.
12. Η εντολή «for i in range(2,2)» εκτελείται 2 φορές.
13. Στο εσωτερικό της «for i in range(Α, Β, C)» πρέπει να υπάρχει μία εντολή που θα μεταβάλλει την τιμή του «i», για να τελειώνει γρηγορότερα η επανάληψη.
14. Στην εντολή «for i in range(Α, Β)» η μεταβλητή i αυξάνεται πάντα κατά ένα, όταν τα Α και Β είναι ακέραιοι αριθμοί και Α<Β.
15. Για να εκτελεστεί η εντολή «for i in range(1,10, C)» πρέπει C>0.
16. Η εντολή «for» εκτελείται τουλάχιστον μία φορά.
17. Στην εντολή «for i in range(Α, Β, C)» πρέπει η μεταβλητή C να έχει πάντα θετικές τιμές.
18. Η εντολή «for i in range(Α, Β, C)» εκτελείται μία φορά, όταν Α=Β.
19. Σε κάθε πρόβλημα που στη λύση του, απαιτείται η χρήση της δομής επανάληψης μπορεί να χρησιμοποιηθεί η εντολή for.
20. Στο εσωτερικό της εντολής «while» πρέπει υποχρεωτικά να υπάρχει η δομή επιλογής, ώστε να ελέγχει τη συνθήκη της επανάληψης.
21. Στην εντολή «while συνθήκη:» η συνθήκη μπορεί να λάβει και άλλες τιμές εκτός από τις τιμές True και False.
22. Για το διάβασμα 100 αριθμών μπορεί να χρησιμοποιηθεί η εντολή «while».
23. Η εντολή «while» είναι η μοναδική εντολή που χρησιμοποιεί η Python για υλοποίηση μιας επαναληπτικής διαδικασίας.
24. Η εντολή «while» μπορεί να μην εκτελεστεί ποτέ.
25. Οι εντολές που περιλαμβάνονται μέσα στη «while» θα εκτελεστούν τουλάχιστον μία φορά.
26. Κάθε δομή επανάληψης «while» είναι πάντα δυνατό να ξαναγραφεί με χρήση της δομής επανάληψης «for».
27. Η εντολή «while συνθήκη» εκτελείται συνεχώς, μέχρι η συνθήκη να γίνει False.
28. Για την εύρεση του μέσου όρου πέντε αριθμών υποχρεωτικά πρέπει να χρησιμοποιηθεί η εντολή «while».
29. Ο αριθμός των επαναλήψεων που θα πραγματοποιήσει η εντολή «while», εξαρτάται από τη συνθήκη που ελέγχει.

30. Εμφωλευμένες εντολές επανάληψης σχηματίζονται μόνο με εντολές while.
31. Στους εμφωλευμένους βρόγχους, ο βρόγχος που ξεκινάει τελευταίος, πρέπει να ολοκληρώνεται τελευταίος.
32. Δεν είναι καλή πρακτική να χρησιμοποιείται η ίδια μεταβλητή ως μετρητής δυο ή περισσότερων βρόγχων που ο ένας βρίσκεται στο εσωτερικό του άλλου.
33. Για να σχηματιστούν εμφωλευμένες εντολές επανάληψης, πρέπει η εσωτερική εντολή επανάληψης να βρίσκετε ολόκληρη μέσα στην εξωτερική.
34. Σε δυο εμφωλευμένες εντολές επανάληψης, η εσωτερική εντολή επανάληψης μπορεί να μην εκτελεστεί ποτέ.
35. Εμφωλευμένες εντολές επανάληψης σχηματίζονται με οποιοδήποτε συνδυασμό «while» και «for».
36. Μπορούν να χρησιμοποιηθούν μέχρι δυο το πολύ εντολές επανάληψης η μία μέσα στην άλλη.
37. Οι δομές επανάληψης χρησιμοποιούνται στις περιπτώσεις που μια ακολουθία εντολών πρέπει να εφαρμοστεί σε ένα σύνολο περιπτώσεων οι οποίες έχουν κάτι κοινό.
38. Οι επαναληπτικές δομές χρησιμοποιούνται στην περίπτωση που μια ομάδα εντολών πρέπει να εκτελεστεί πολλές φορές.
39. Βρόγχος ονομάζεται το μπλοκ των εντολών που περιέχονται σε μια δομή επιλογής ή σε μια δομή επανάληψης.
40. Εντός μιας δομής επιλογής δεν μπορεί να περιέχεται δομή επανάληψης.
41. Εντός μιας δομής επανάληψης δεν μπορεί να περιέχεται μια δομή επιλογής.
42. Ένα τμήμα αλγορίθμου που εκτελείται επαναληπτικά αποτελείται βρόγχος.
43. Μια δομή επανάληψης πρέπει να φροντίζει για τη μεταβολή της τιμής της συνθήκης ελέγχου έτσι, ώστε κάποτε να τερματίζεται η επανάληψη.
44. Η δομή while χρησιμοποιείται μόνο όταν γνωρίζουμε το πλήθος των επαναλήψεων.
45. Στην επαναληπτική δομή while δεν γνωρίζουμε εκ των προτέρων το πλήθος των επαναλήψεων.
46. Στη δομή while η ομάδα εντολών εκτελείται κατ' επανάληψη μέχρι η συνθήκη να γίνει False.
47. Με χρήση της δομής while επιτυγχάνεται η επανάληψη μιας διαδικασίας με βάση κάποια συνθήκη.
48. Στη δομή επανάληψης for η μεταβλητή πρέπει πάντοτε να ονομάζεται i.
49. Η δομή while τερματίζει τις επαναλήψεις της όταν η συνθήκη ελέγχου γίνεται False.
50. Όταν το πλήθος των επαναλήψεων είναι γνωστό, δεν μπορεί να χρησιμοποιηθεί η δομή επανάληψης while.
51. Η εντολή επανάληψης while εκτελείται τουλάχιστον μια φορά.
52. Στη δομή επανάληψης while οι μεταβλητές που συμμετέχουν στη συνθήκη ελέγχου πρέπει να πάρουν τιμή πριν από το βρόγχο.
53. Όταν σε μια δομή επανάληψης for παραλείπεται το βήμα, τότε εννοείται πως το βήμα είναι 1.
54. Στη δομή επανάληψης for δεν είναι δυνατόν η αρχική τιμή να είναι μεγαλύτερη από την τελική.
55. Η δομή επανάληψης for πρέπει πάντοτε να έχει ως βήμα ένα θετικό αριθμό.
56. Αν το βήμα μιας δομής for είναι αρνητικός αριθμός, τότε δεν εκτελείται καμία επανάληψη.
57. Σε μια δομή επανάληψης for πρέπει υποχρεωτικά να αναφέρεται η τιμή του βήματος.
58. Στη δομή for, αν το βήμα είναι γνωστό, τότε μπορεί να παραλειφθεί.
59. Στη δομή επανάληψης for το βήμα δε μπορεί να είναι μηδέν.

60. Οι εντολές του βρόγχου for εκτελούνται τουλάχιστον μια φορά.
61. Στην εντολή for ο βρόγχος επαναλαμβάνεται για προκαθορισμένο αριθμό επαναλήψεων.
62. Στη δομή επανάληψης for πρέπει η τιμή του μετρητή να μεταβάλλεται εντός του βρόγχου.
63. Κάθε πρόβλημα που απαιτεί τη χρήση δομής επανάληψης μπορεί να επιλυθεί με τη χρήση της δομής for.
64. Ο βρόγχος «for i in range(5,5):» δεν εκτελείται καμία φορά.
65. Εντός της δομής for δεν επιτρέπεται η τροποποίηση της τιμής του μετρητή.
66. Όταν το πλήθος των επαναλήψεων είναι γνωστό, δε μπορεί να χρησιμοποιηθεί η δομή επανάληψης while.
67. Στην επαναληπτική δομή while δεν γνωρίζουμε εκ των προτέρων το πλήθος των επαναλήψεων.
68. Στη δομή επανάληψης while ο βρόχος εκτελείται κατ' επανάληψη μέχρι η συνθήκη να γίνει False.
69. Κάθε πρόβλημα που απαιτεί τη χρήση της δομής επανάληψης μπορεί να επιλυθεί με τη χρήση της δομής while.

Υποπρογράμματα – Συναρτήσεις - Αρθρώματα

1. Ένα πρόγραμμα μπορεί να χωριστεί σε μικρότερα αυτόνομα κομμάτια προγράμματος.
2. Ένα υποπρόγραμμα μπορούμεν να το χρησιμοποιήσουμε μια μόνο φορά μέσα σε ένα πρόγραμμα.
3. Η είσοδος σε ένα υποπρόγραμμα γίνεται μόνο από ένα σημείο.
4. Με τον τμηματικό προγραμματισμό μπορεί σε μια χρονική στιγμή να εκτελούνται δυο υποπρογράμματα.
5. Απαγορεύεται ένα υποπρόγραμμα να καλεί ένα άλλο υποπρόγραμμα.
6. Σε ένα υποπρόγραμμα μπορεί να υπάρχουν εντολές που δεν είναι σωστά διατυπωμένες και δεν εκτελούνται.
7. Ένα υποπρόγραμμα μπορεί να επιστρέφει μια τιμή στο σημείο από το οποίο κλήθηκε.
8. Μέσα σε ένα υποπρόγραμμα μπορούμε να υλοποιήσουμε τη δομή επανάληψης ή το δομή επιλογής.
9. Κάθε πρόγραμμα υποχρεωτικά έχει και υποπρόγραμμα.
10. Τη στιγμή εκτέλεσης ενός υποπρογράμματος, υπάρχει προσωρινή διακοπή εκτέλεσης του κυρίου προγράμματος.
11. Ο ορισμός μιας συνάρτησης περιλαμβάνει, τη λέξη def, το όνομά της, τις παραμέτρους εισόδου και την άνω κάτω τελεία.
12. Ο ορισμός των συναρτήσεων πάντα γίνεται στο τέλος του προγράμματος.
13. Μια συνάρτηση μπορεί να κληθεί από μια άλλη συνάρτηση.
14. Η επιστρεφόμενη τιμή μιας συνάρτησης μπορεί να χρησιμοποιηθεί σε μια έκφραση.
15. Ο τρόπος κλήση μιας συνάρτησης στο κύριο πρόγραμμα, είναι διαφορετικός από τον τρόπο κλήσης στο εσωτερικό μιας άλλης συνάρτηση.
16. Για να χρησιμοποιήσουμε μέσα στο πρόγραμμα, μία συνάρτηση ορισμένη από τον προγραμματιστή, πρέπει υποχρεωτικά να την ορίσουμε.
17. Μέσα σε μια συνάρτηση ορισμένη από τον προγραμματιστή μπορεί να υπάρχουν δύο διαφορετικές μεταβλητές με το ίδιο όνομα.
18. Υποχρεωτικά όλες οι συναρτήσεις επιστρέφουν τουλάχιστον μια τιμή.
19. Κατά την κλήση μιας συνάρτησης, τα ορίσματα και οι παράμετροι μπορούν να έχουν το ίδιο όνομα.
20. Όταν μια συνάρτηση καλείται από διαφορετικά σημεία ενός προγράμματος, τα ορίσματα πρέπει να είναι ίδια.
21. Μεταβλητές με απεριόριστη εμβέλεια, είναι ορατές σε οποιοδήποτε τμήμα προγράμματος.
22. Οι τοπικές μεταβλητές έχουν περιορισμένη εμβέλεια.
23. Οι παράμετροι μιας συνάρτησης στην Python είναι καθολικές μεταβλητές.

24. Μεταβλητές με απεριόριστη εμβέλεια, είναι ορατές σε οποιοδήποτε τμήμα του προγράμματος.
25. Οι τοπικές μεταβλητές έχουν περιορισμένη εμβέλεια.
26. Οι παράμετροι μιας συνάρτησης στην Python είναι καθολικές μεταβλητές.
27. Όταν μια συνάρτηση καλείται από διαφορετικά σημεία ενός προγράμματος τα ορίσματα που δέχεται είναι πάντα καθολικές μεταβλητές.
28. Στην περιορισμένη εμβέλεια, οι μεταβλητές χρησιμοποιούνται μόνο στο τμήμα προγράμματος που ορίζονται.
29. Σε μια συνάρτηση, είναι δυνατόν να υπάρχουν δύο τοπικές μεταβλητές με το ίδιο όνομα.
30. Μια καθολική μεταβλητή μπορεί να χρησιμοποιηθεί στο εσωτερικό μιας συνάρτησης.
31. Η αλλαγή τιμής μιας καθολικής μεταβλητής στο σώμα της συνάρτησης, γίνεται με τη χρήση της εντολή global στη μεταβλητή.
32. Οι τοπικές μεταβλητές μιας συνάρτησης διατηρούνται και εκτός συνάρτησης.
33. Κάθε μεταβλητή έχει την εμβέλεια του τμήματος που δηλώνεται.
34. Σε ένα πρόγραμμα μπορούμε να εισάγουμε ένα μόνο άρθρωμα.
35. Η ομαδοποίηση σχετικού κώδικα σε ένα άρθρωμα καθιστά τον κώδικα ευκολότερο στη χρήση του.
36. Σε ένα άρθρωμα μπορούν να υπάρχουν και μεταβλητές.
37. Η χρήση μιας συνάρτησης ενός αρθρώματος σ' ένα πρόγραμμα, προϋποθέτει την εισαγωγή του αρθρώματος στο πρόγραμμα.
38. Η χρήση μιας συνάρτησης ενός αρθρώματος που έχει εισαχθεί στο πρόγραμμα με την εντολή import, γίνεται υποχρεωτικά με «άρθρωμα τελεία όνομα συνάρτησης».
39. Όταν εισάγουμε ένα άρθρωμα σ' ένα πρόγραμμα πρέπει υποχρεωτικά να χρησιμοποιήσουμε όλες τις συναρτήσεις του.
40. Οι ενσωματωμένες συναρτήσεις της Python, πρέπει να εισαχθούν με την εντολή import, πριν χρησιμοποιηθούν σε ένα πρόγραμμα.
41. Οι βιβλιοθήκες παρέχουν στους προγραμματιστές χρήσιμα εργαλεία για την ανάπτυξη προγραμμάτων.
42. Στο εσωτερικό μιας, κατασκευασμένης από τον προγραμματιστή, συνάρτησης μπορεί να χρησιμοποιηθεί μια συνάρτηση από εξωτερικό άρθρωμα.
43. Υπάρχει δυνατότητα να εισαχθούν ορισμένες από τις συναρτήσεις ενός αρθρώματος σε ένα πρόγραμμα.
44. Τα ονόματα των παραμέτρων πρέπει πάντα να είναι διαφορετικά από τα ονόματα των ορισμάτων.
45. Οι συναρτήσεις μπορούν να εκτελέσουν οποιαδήποτε λειτουργία από αυτές που μπορεί να εκτελέσει ένα πρόγραμμα.
46. Ένα υποπρόγραμμα δεν μπορεί να κληθεί περισσότερες από δυο φορές από το κυρίως πρόγραμμα.
47. Η λίστα παραμέτρων ενός υποπρογράμματος μπορεί να είναι κενή.

Δομές Δεδομένων – Συμβολοσειρές

1. Η επιλογή της κατάλληλη δομής δεδομένων παίζει σημαντικό ρόλο στην ανάπτυξη αλγορίθμου για την επίλυση ενός προβλήματος.
2. Σε μία στατική δομή, το ακριβές μέγεθος της, καθορίζεται κατά τη στιγμή του προγραμματισμού.
3. Οι δυναμικές δομές δεδομένων μπορούν να χρησιμοποιήσουν όση μνήμη απαιτείται για τη δημιουργία της δομής κατά την εκτέλεση του προγράμματος.
4. Το μέγεθος μιας δυναμικής δομής είναι γνωστό κατά τη φάση της μεταγλώττισης του προγράμματος.
5. Οι συμβολοσειρές ανήκουν στις στατικές δομές ενώ οι λίστες στις δυναμικές δομές.
6. Η λίστα είναι μια μη μεταβαλλόμενη δομή δεδομένων.
7. Η στοίβα και η ουρά μπορούν να υλοποιηθούν με τη χρήση ενσωματωμένων δομών δεδομένων της Python.
8. Οι συμβολοσειρές ως δομή δεδομένων, έχουν μεταβαλλόμενο μέγεθος και σταθερό περιεχόμενο.
9. Με το s[1] αναφερόμαστε στον πρώτο χαρακτήρα της συμβολοσειράς s.
10. Το αποτέλεσμα της εφαρμογής της συνάρτησης len στη συμβολοσειρά 'epal' είναι τέσσερα.
11. Με τη βοήθεια του τελεστή in μπορούμε να πάρουμε ένα τμήμα μιας συμβολοσειράς.
12. Το αποτέλεσμα της πράξης «'m' not in 'Nantia'» είναι True.
13. Μια συμβολοσειρά αποτελείται μόνο από γράμματα λατινικού αλφαβήτου.
14. Χρησιμοποιώντας τον τελεστή «-» μπορούμε να μεταβάλλουμε ένα μέρος μιας συμβολοσειράς.
15. Το s[2] και s[-2] αναφέρονται στον ίδιο χαρακτήρα της συμβολοσειράς s.

Δομές Δεδομένων – Λίστες

1. Οι συμβολοσειρές ανήκουν στις στατικές δομές ενώ οι λίστες στις δυναμικές δομές.
2. Στη δομή δεδομένων «λίστα», το ακριβές μέγεθος της καθορίζεται κατά τη στιγμή του προγραμματισμού.
3. Οι λίστες μπορούν να χρησιμοποιήσουν όση μνήμη απαιτείται για τη δημιουργία της δομής κατά την εκτέλεση του προγράμματος.
4. Μία λίστα δημιουργείται αν γράψουμε ένα σύνολο αντικειμένων μέσα σε παρενθέσεις χωρισμένα μεταξύ τους με κόμμα.
5. Μία λίστα μπορούμε να θεωρήσουμε ότι είναι ένα σύνολο από αντικείμενα.
6. Το κάθε στοιχείο της λίστας υποχρεωτικά εμφανίζεται μόνο μία φορά μέσα σε αυτήν.
7. Σε μία λίστα μπορούν να υπάρχουν και στοιχεία διαφορετικού τύπου.
8. Σε ένα πρόγραμμα μπορεί να υπάρχει μόνο μια λίστα δεδομένων.
9. Σε ένα πρόγραμμα, μπορούμε να τροποποιήσουμε το περιεχόμενο κάθε στοιχείου μιας λίστας, αλλά όχι το μέγεθος της.
10. Ο αριθμός που δείχνει τη θέση του στοιχείου μέσα σε μια λίστα μπορεί να λάβει και πραγματικές τιμές.
11. Γράφοντας L[1] αναφερόμαστε στο πρώτο στοιχείο της λίστας L.
12. Το L[0] και το L[-1] αναφέρονται στο ίδιο στοιχείο της λίστας.
13. Για να προσθέσουμε το στοιχείο «A» στο τέλος της λίστας let γράφουμε let = let +['A']
14. Η εντολή «print L[2 : 2]» εμφανίζει το δεύτερο στοιχείο της λίστας L.
15. Με την εντολή «L + 2» η λίστα L επαναλαμβάνεται δύο φορές.
16. Ο τελεστής in επιστρέφει True όταν ένα στοιχείο υπάρχει σε μία λίστα.
17. Το αποτέλεσμα της εφαρμογής της συνάρτησης len στη λίστα L[1,2,3] είναι τρία.
18. Με τη βοήθεια του τελεστή «*» μπορούμε να πάρουμε ένα τμήμα μιας λίστας.
19. Χρησιμοποιώντας τον τελεστή «-» μπορούμε να αφαιρέσουμε ένα στοιχείο μιας λίστας
20. Το αποτέλεσμα της πράξης «'m' not in L['a', 'b', 'c']» είναι True.
21. Σε μία λίστα δεν είναι δυνατό να αφαιρέσουμε στοιχεία από οποιαδήποτε θέση.
22. Σε μία λίστα μπορούμε να προσθέσουμε ένα στοιχείο σε οποιαδήποτε θέση μέσα σε αυτήν.
23. Με τη μέθοδο append(0) μπορούμε να προσθέσουμε ένα (Στοιχείο στην αρχή μιας λίστας.
24. Η μέθοδος insert()μπορεί να εφαρμοστεί μία μόνο φορά στην ίδια λίστα.
25. Για να αφαιρέσουμε ένα στοιχείο από μια λίστα, μπορούμε να χρησιμοποιήσουμε τη μέθοδο pop().
26. Για να διασχίσουμε/σαρώσουμε μια λίστα θα πρέπει υποχρεωτικά να γνωρίζουμε το πλήθος των στοιχείων της.
27. Εκτός από το άθροισμα όλων των στοιχείων μιας λίστας, μπορούμε να προσθέσουμε και ορισμένα από αυτά.
28. Η εύρεση του μέσου όρου μιας λίστας απαιτεί τη διαίρεση του αθροίσματος των στοιχείων της με το μέγεθος της λίστας.
29. Μια συνάρτηση δεν μπορεί να επιστρέφει μια λίστα, διότι η λίστα περιέχει πολλές διαφορετικές τιμές.
30. Μία λίστα μπορεί να χρησιμοποιηθεί ως παράμετρος σε μία συνάρτηση, μόνο αν περιέχει αριθμούς.

31. Ο διαχωρισμός μιας λίστας πραγματοποιείται πάντοτε με βάση κάποια κριτήρια.
32. Μία λίστα μπορούμε να τη διαχωρίσουμε μόνο σε δύο λίστες.
33. Η συγχώνευση δύο η περισσοτέρων λιστών μπορεί να γίνει και με τον τελεστή «*».
34. Δύο ταξινομημένες λίστες μπορούν να συγχωνευτούν σε μια νέα λίστα επίσης ταξινομημένη.
35. Συγχώνευση ταξινομημένων λιστών μπορεί να γίνει μόνο αν αυτές είναι ταξινομημένες κατά αύξουσα σειρά.
36. Οι εντολές για την εύρεση του μικρότερου στοιχείου μιας λίστας μπορούν να χρησιμοποιηθούν και για την εύρεση του μεγαλύτερου στοιχείου της ίδιας λίστας.
37. Η ίδια λίστα μπορεί να αποθηκεύσει και ακεραίους αριθμούς και χαρακτήρες.
38. Η ανάγνωση των στοιχείων μιας λίστας μπορεί να γίνει και από μία συνάρτηση.
39. Μια λίστα μπορεί να αποθηκεύσει ακέραιες και λογικές τιμές.
40. Εφόσον το πλήθος των στοιχείων μιας λίστας είναι γνωστό, δεν μπορεί να χρησιμοποιηθεί η δομή while για την προσπέλαση των στοιχείων του.
41. Για να προσπελάσουμε μαζικά τα στοιχεία μιας λίστας χρησιμοποιούμε επαναληπτική δομή.
42. Για τον υπολογισμό του μέσου όρου μιας λίστας αριθμών πρέπει να προσπελαστεί ολόκληρη η λίστα.
43. Ο δείκτης μιας λίστας πρέπει πάντοτε να ονομάζεται i.
44. Οι δυνατοί τύποι δεδομένων των στοιχείων μιας λίστας είναι ίδιοι με αυτούς μιας μεταβλητής.
45. Οι παρακάτω αντιμεταθέσεις είναι ισοδύναμες :

```
tmp = A[j]          tmp = A[j-1]
A[j] = A[j-1]       A[j-1] = A[j]
A[j-1] = tmp        A[j] = tmp
```

Δομές Δεδομένων – Ουρά - Στοίβα

1. Οι λειτουργίες «ώθηση» και «απώθηση» είναι οι βασικές και κύριες λειτουργίες σε μία στοίβα.
2. Κάθε στοιχείο που εισάγεται πρώτο σε μία στοίβα είναι και αυτό που εξάγεται πρώτο.
3. Η μέθοδος επεξεργασίας LIFO περιγράφει τη διαδικασία εκείνη κατά την οποία κάθε δεδομένο που τοποθετείται τελευταίο, εξάγεται πρώτο.
4. Όταν η ώθηση ενός στοιχείου σε μια στοίβα γίνεται στο κάτω μέρος της, η απώθηση μπορεί να γίνει από το πάνω μέρος της.
5. Σε μία στοίβα μπορούμε να εξαγάγουμε το τελευταίο εισαχθέν στοιχείο, χωρίς να έχει εξαχθεί το προτελευταίο στοιχείο.
6. Μία στοίβα μπορούμε να την υλοποιήσουμε με τη βοήθεια μιας λίστας.
7. Η χρήση της λίστας για την υλοποίηση μιας στοίβας μας επιτρέπει την εκτέλεση της ώθησης και της απώθησης μόνο από το ένα άκρο της.
8. Όταν εισάγουμε ένα στοιχείο σε μία λίστα που υλοποιεί τη δομή στοίβα πρώτα ελέγχουμε αν η στοίβα είναι άδεια.
9. Εκτελώντας τη λειτουργία της εξαγωγής αφαιρούμε το στοιχείο που βρίσκεται στην κορυφή μιας στοίβας.
10. Η διεπαφή/διασύνδεση ορίζει τι μπορεί να κάνει μια δομή κώδικα και όχι τον τρόπο με τον οποίο το κάνει.
11. Οι λειτουργίες «Εισαγωγή» και « Εξαγωγή» είναι οι βασικές και κύριες λειτουργίες σε μία ουρά.
12. Κάθε στοιχείο που εισάγεται τελευταίο σε μία ουρά είναι και αυτό που εξάγεται πρώτο.
13. Η μέθοδος επεξεργασίας FIFO περιγράφει τη διαδικασία εκείνη κατά την οποία κάθε δεδομένο που τοποθετείται τελευταίο, εξάγεται πρώτο.
14. Η μέθοδος επεξεργασίας «πρώτο μέσα πρώτο έξω (FIFO)» εφαρμόζεται στη δομή δεδομένων ουρά.
15. Στην ουρά, με τη λειτουργία της εισαγωγής μπορούν να εισαχθούν περισσότερα από ένα δεδομένα μαζί.
16. Μία ουρά μπορούμε να την υλοποιήσουμε με τη βοήθεια μιας λίστας.
17. Αν η εισαγωγή στοιχείου σε μια ουρά, γίνεται από το ένα άκρο μιας λίστας που υλοποιεί την ουρά, η εξαγωγή γίνεται από το άλλο άκρο της λίστας.
18. Όταν εισάγουμε ένα στοιχείο σε μία λίστα που υλοποιεί τη δομή ουρά, πρώτα ελέγχουμε αν η ουρά είναι άδεια.
19. Μια ουρά διατηρεί τα δεδομένα ταξινομημένα ως προς τη σειρά άφιξής τους.
20. Σε μια ουρά μπορούμε να προσθέσουμε στοιχεία στο μέσο της.
21. Η εισαγωγή και η εξαγωγή στοιχείων από μια ουρά πραγματοποιούνται από το ίδιο άκρο κάθε φορά.
22. Η στοίβα και η ουρά είναι δυναμικές δομές δεδομένων.

Ταξινόμηση - Αναζήτηση

1. Όταν ταξινομούμε ένα σύνολο δεδομένων κατά φθίνουσα σειρά, τότε αυτά τοποθετούνται από το μικρότερο προς το μεγαλύτερο.
2. Η λειτουργία της ταξινόμησης μπορεί να εφαρμοστεί και σε μία λίστα αλφαριθμητικών.
3. Η ταξινόμηση «φυσαλίδα» είναι ο πιο απλός αλλά και ο πιο γρήγορος αλγόριθμος ταξινόμησης.
4. Η ταξινόμηση του περιεχομένου μιας λίστας με τη μέθοδο της «ευθείας εισαγωγής» βασίζεται στην αρχή της σύγκρισης και αντιμετάθεσης ζευγών γειτονικών στοιχείων της λίστας.
5. Το πλήθος των συγκρίσεων και αντιμεταθέσεων που πραγματοποιούνται στην «ταξινόμηση με εισαγωγή», εξαρτάται από τις θέσεις των στοιχείων της λίστας.
6. Η «(ταξινόμηση με επιλογή», βασίζεται στην επιλογή του μικρότερου ή του μεγαλύτερου στοιχείου από αυτά που δεν έχουν ταξινομηθεί και την τοποθέτησή του στην αρχή της λίστας.
7. Η μέθοδος «ταξινόμηση με εισαγωγή» μπορεί να εφαρμοστεί και σε λίστα αλφαριθμητικών.
8. Η «ταξινόμηση με επιλογή» και η «ταξινόμηση ευθείας ανταλλαγής» είναι κατάλληλοι για λίστες που είναι σχεδόν ταξινομημένες.
9. Με τη μέθοδο «ταξινόμηση με εισαγωγή» μπορούμε να διατάξουμε τα στοιχεία μιας λίστας μόνο από το μεγαλύτερο προς το μικρότερο.
10. Η δυαδική αναζήτηση όταν χρησιμοποιείται σε μη ταξινομημένους πίνακες είναι πιο αργή από τη σειριακή αναζήτηση.
11. Ο αλγόριθμος της σειριακής αναζήτησης χρησιμοποιείται αποκλειστικά σε ταξινομημένους πίνακες.
12. Η πιο απλή και λιγότερη αποτελεσματική μέθοδος αναζήτησης είναι η σειριακή αναζήτηση.
13. Χρησιμοποιώντας τον αλγόριθμο της δυαδικής αναζήτησης μπορούμε να ψάξουμε ένα αλφαριθμητικό δεδομένο σε μία λίστα αλφαριθμητικών.
14. Η δυαδική αναζήτηση βρίσκει πιο γρήγορα το ζητούμενο από ότι η σειριακή αναζήτηση.
15. Κατά την εφαρμογή της δυαδικής αναζήτησης σε μια ταξινομημένη λίστα προσπελαύνονται υποχρεωτικά όλα τα στοιχεία της λίστας.
16. Σε μία λίστα που περιέχει τα ονόματα 100 μαθητών σε τυχαία σειρά, μπορούμε να χρησιμοποιήσουμε τη δυαδική αναζήτηση για να εντοπίσουμε το όνομα ενός μαθητή.
17. Η δυαδική αναζήτηση είναι πάντα ο καλύτερος τρόπος αναζήτησης.
18. Η δυαδική αναζήτηση εκμεταλλεύεται τη διάταξη των στοιχείων ενός συνόλου δεδομένων για τη γρήγορη εύρεση ενός στοιχείου.
19. Σε ένα ταξινομημένο πίνακα η δυαδική αναζήτηση είναι κατά κανόνα ταχύτερη από την σειριακή αναζήτηση.
20. Σε ταξινομημένη λίστα μπορεί να χρησιμοποιηθεί είτε η σειριακή είτε η δυαδική αναζήτηση.
21. Η δυαδική αναζήτηση χρησιμοποιείται αποκλειστικά σε ταξινομημένες λίστες.
22. Η δυαδική αναζήτηση είναι αποδοτικότερη από τη σειριακή αναζήτηση.

23. Η μέθοδος της ταξινόμησης ευθείας ανταλλαγής βασίζεται στην αρχή της σύγκρισης και ανταλλαγής ζευγών γειτονικών στοιχείων μέχρις ότου διαταχθούν όλα τα στοιχεία.
24. Η ταξινόμηση έχει ως στόχο να διατάξει τα στοιχεία ενός πίνακα σε αύξουσα ή φθίνουσα σειρά.
25. Ο αλγόριθμος της φυσαλίδας δεν μπορεί να χρησιμοποιηθεί σε λίστα χαρακτήρων.

Αρχεία

1. Τα αρχεία έχουν την ιδιότητα να διατηρούν τα δεδομένα που υπάρχουν σε αυτά και μετά το τέλος ενός προγράμματος.
2. Το διάβασμα του περιεχομένου ενός αρχείου σε πρόγραμμα, μπορεί να γίνει και χωρίς να προηγηθεί το άνοιγμα του.
3. Ένα αρχείο μπορεί να ανοιχτεί μόνο για ανάγνωση του περιεχομένου του και για εγγραφή δεδομένων σε αυτό.
4. Για να κλείσουμε ένα αρχείο, πρέπει πρώτα να έχει ανοίξει.
5. Εγγραφή δεδομένων σ' ένα αρχείο μπορούμε να κάνουμε μόνο μία φορά ενώ επέκταση μπορούμε πολλές φορές.
6. Οι μέθοδοι read() και realine() έχουν το ίδιο ακριβώς αποτέλεσμα στο διάβασμα αλφαριθμητικών δεδομένων.
7. Με την κατάλληλη μέθοδο μπορούμε να αλλάξουμε την τρέχουσα θέση μας μέσα σε ένα αρχείο.
8. Ένα αρχείο δεδομένων μπορούμε να δημιουργήσουμε και εκτός περιβάλλον μιας γλώσσας προγραμματισμού με ένα απλό συντάκτη.

Αντικειμενοστρεφής προγραμματισμός

1. Ο αντικειμενοστρεφής προγραμματισμός εστιάζει στις έννοιες στις οποίες αναθέτει χαρακτηριστικά και όχι στις διαδικασίες.
2. Η Python είναι μια αντικειμενοστρεφής γλώσσα προγραμματισμού.
3. Κάθε αντικείμενο έχει μόνο μια ιδιότητα.
4. Οι μέθοδοι επιτρέπουν στα αντικείμενα να κάνουν διάφορες ενέργειες που ελέγχουν τις ιδιότητες του αντικειμένου.
5. Μια από τις ενέργειες ενός κατασκευαστή είναι η καταστροφή του αντικειμένου και η απελευθέρωση μνήμης.
6. Μια κλάση αποτελείται μόνο από μεθόδους.
7. Στην επικεφαλίδα μιας κλάση υπάρχει η δεσμευμένη λέξη def.
8. Η αρχικοποίηση των τιμών ενός αντικειμένου γίνεται στη μέθοδο init.
9. Το στιγμιότυπο μιας κλάσης δημιουργείται από την κλάση με συγκεκριμένες τιμές που ορίζονται κατά τη κλήση της κλάσης.
10. Οι μεταβλητές κλάσης είναι διαθέσιμες και κοινές σε όλα τα αντικείμενα της κλάσης.
11. Μια στατική μέθοδος λειτουργεί στο επίπεδο της κλάσης και όχι στο επίπεδο του στιγμιότυπου.
12. Μια μέθοδος κλάσης δεν είναι κληρονομήσιμη.
13. Από μια κλάση μπορούμε να δημιουργήσουμε ένα μόνο στιγμιότυπο.
14. Η self ως παράμετρος σε μια μέθοδο κλάσης, εξασφαλίζει ότι η μέθοδος αναφέρεται στο ίδιο το αντικείμενο και όχι στην κλάση.
15. Μια στατική μέθοδος και μια μέθοδος κλάσης έχουν την ίδια λειτουργία.