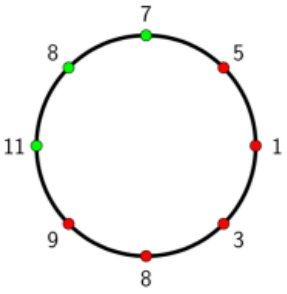
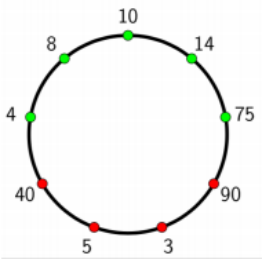
1. Εμφάνιση των 4ψήφιων ακέραιων με τα ψηφία τους **ανά δύο ίσα**. π.χ. 5757 (τροποποίηση για 6ψήφιους) (*ana2isa.psc*)
2. Εμφάνιση των **δίσεκτων** (πολλαπλάσια του 4 και όχι του 100 ή πολλαπλάσια του 400) ετών του 21ου αιώνα (*disekta21oyAiwna.psc*)
3. Ισχύει ότι **N^2 = άθροισμα των N πρώτων περιττών** π.χ. 4^2 = 1+3+5+7. Εισαγωγή ακέραιου Ν και επαλήθευση του παραπάνω (*epalitheyshNtetragwno.psc*)
4. Εμφάνιση των **3ψήφιων** ακέραιων xyz: x<y<z και πόσοι είναι συνολικά (*tripshfioiMeSxeshMikroteroy.psc*)
5. Εμφάνιση των 3ψήφιων ακέραιων xyz που είναι «**ισορροπημένοι**»: max{x,y,z} = (x+y+z)/2 π.χ. 123: 3 = (1+2+3)/2 και πόσοι είναι συνολικά (126) (*isorrophmenoi.psc*)
6. Εισαγωγή πραγματικού x και εμφάνιση του **κλάσματος** M/N (M,N ακέραιοι στο [1,100]) που προσεγγίζει καλύτερα τον x (*proseggishMeKlasma.psc*)
7. Εμφάνιση των 100 πρώτων τιμών της **ακολουθίας**: 1,2,4,7,11,16,22,…4951 (*akoloythia100oroi.psc*)
8. Εισαγωγή **κεφαλαίου** κατάθεσης (€), ετών κατάθεσης και % επιτοκίου. Εμφάνιση της πορείας του κεφαλαίου χρόνο με το χρόνο
9. Εισαγωγή ακέραιου και εμφάνιση του ακέραιου με τα **ψηφία του αρχικού αυξημένα κατά 1** (το 9 να γίνεται 0). Π.χ. 12389 ➔23490 (*akeraios_pshfia+1.psc*)
10. Εισαγωγή 10 βαθμών στην 20θμια (0-20) με έλεγχο εγκυρότητας και εύρεση: του μέσου όρου τους και του **πλήθους των άκυρων** βαθμών που δόθηκαν (*plithos_akyrwn_bathmwn.psc*)
11. Εμφάνιση όλων των θετικών 5ψήφιων ακέραιων που είναι **συμμετρικοί** (π.χ. 12321) (*symmetrikoi5pshfioi.psc*)
12. 

Εισαγωγή 2 θετικών ακέραιων και έλεγχος αν είναι **φίλιοι**: το άθροισμα των γνήσιων διαιρετών του καθενός (δηλ. εκτός του εαυτού τους) ισούται με τον άλλο. (*filioi_arithmoi.psc*)
13. Εισαγωγή 2 θετικών ακέραιων και εμφάνιση του **ελάχιστου κοινού πολλαπλάσιού** τους (π.χ. x=15, y=100, ekp = 300) (*EKP.psc*)
14. (30ος ΠΑΝΕΛΛΗΝΙΟΣ ΔΙΑΓΩΝΙΣΜΟΣ ΠΛΗΡΟΦΟΡΙΚΗΣ 2018 ΘΕΜΑ Β΄ ΦΑΣΗΣ ΓΥΜΝΑΣΙΟΥ). Δύο αδέλφια, ο Κόκκινος και η Πράσινη, θέλουν να επισκεφθούν τα σπίτια που βρίσκονται κατά μήκος ενός **κυκλικού δρόμου** για να πουν τα κάλαντα. Σε κάθε σπίτι, τα παιδιά ξέρουν από πριν πόσα χρήματα συνηθίζουν να τους δίνουν οι κάτοικοι. Για οικονομία χρόνου, αποφασίζουν να μοιράσουν τα σπίτια: ο Κόκκινος θα πάει σε κάποια σπίτια που είναι διαδοχικά, πάνω στον κύκλο, και η Πράσινη θα πάει στα υπόλοιπα, που προφανώς είναι επίσης διαδοχικά. Για να είναι δίκαιη η μοιρασιά, θέλουν να μοιράσουν τα σπίτια με τέτοιο τρόπο ώστε τα συνολικά χρήματα που θα πάρει κάθε παιδί να διαφέρουν μεταξύ τους όσο το δυνατόν λιγότερο. Να αναπτύξετε ένα πρόγραμμα το οποίο, αφού διαβάσει το πλήθος των σπιτιών (*το πολύ 30*) και τα ποσά των φιλοδωρημάτων σε κάθε σπίτι, θα βρίσκει την ελάχιστη δυνατή τιμή της διαφοράς των συνολικών ποσών που θα πάρουν ο Κόκκινος και η Πράσινη *καθώς και τα χωριά που θα επισκεφθεί ο*





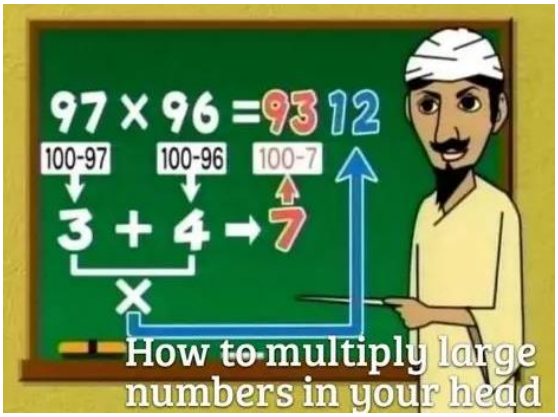
καθένας. Π.χ. με διαφορά ποσών = 0€, με διαφορά ποσών = 27€. (*kalanta.psc*)

15. Πρόγραμμα που διαβάζει έναν πραγματικό αριθμό x και έναν ακέραιο αριθμό n και **στρογγυλοποιεί** τον x κατά n δεκαδικά ψηφία. Π.χ. x = 12.345, n = 2 ➔x = 12.34 (*stroggylopoihsh_dekadikwn.psc*)
16. Πάρτε οποιοδήποτε θετικό ακέραιο n. Αν ο n είναι άρτιος, διαιρέστε τον δια το 2, για να πάρετε το n/2. Εάν ο n είναι περιττός, πολλαπλασιάστε τον επί 3 και προσθέστε 1 για να πάρετε το 3n+1. Κατά την εικασία του **Κόλατζ**, από όποιο αριθμό κι αν ξεκινήσετε, θα καταλήξετε πάντα στο ένα. Μετρήστε τον αριθμό των βημάτων που απαιτήθηκαν. (*colatz.psc*)
17. Εμφάνιση όλων των ακέραιων z: **z = x^2 + y^2**, για κάθε ακέραιο x στο [2,10] και για κάθε ακέραιο y στο [x, 10]: 8=2^2+2^2, 13=2^2+3^2, 20=2^2+4^2, 29=2^2+5^2, 40=2^2+6^2, 53=2^2+7^2, 68=2^2+8^2, 85=2^2+9^2, 104=2^2+10^2, 18=3^2+3^2, 25=3^2+4^2, 34=3^2+5^2, 45=3^2+6^2, 58=3^2+7^2, 73=3^2+8^2, 90=3^2+9^2, 109=3^2+10^2, 32=4^2+4^2, 41=4^2+5^2, 52=4^2+6^2, 65=4^2+7^2, 80=4^2+8^2, 97=4^2+9^2, 116=4^2+10^2, 50=5^2+5^2, 61=5^2+6^2, 74=5^2+7^2, 89=5^2+8^2, 106=5^2+9^2,125=5^2+10^2, 72=6^2+6^2, 85=6^2+7^2, 100=6^2+8^2, 117=6^2+9^2, 136=6^2+10^2, 98=7^2+7^2, 113=7^2+8^2, 130=7^2+9^2, 149=7^2+10^2, 128=8^2+8^2, 145=8^2+9^2, 164=8^2+10^2, 162=9^2+9^2, 181=9^2+10^2, 200=10^2+10^2 (*athroisma_tetragwnwn.psc*)
18. Εμφάνιση των 100 πρώτων ακέραιων x (>1) που αποτελούν **άθροισμα των τετραγώνων** δύο θετικών ακέραιων: $x = y^2 + z^2$.

```
2 = 1 ^2+ 1 ^2
5 = 1 ^2+ 2 ^2
8 = 2 ^2+ 2 ^2
10 = 1 ^2+ 3 ^2
13 = 2 ^2+ 3 ^2
17 = 1 ^2+ 4 ^2
18 = 3 ^2+ 3 ^2
20 = 2 ^2+ 4 ^2
25 = 3 ^2+ 4 ^2
26 = 1 ^2+ 5 ^2
29 = 2 ^2+ 5 ^2
32 = 4 ^2+ 4 ^2
34 = 3 ^2+ 5 ^2
37 = 1 ^2+ 6 ^2
40 = 2 ^2+ 6 ^2
```

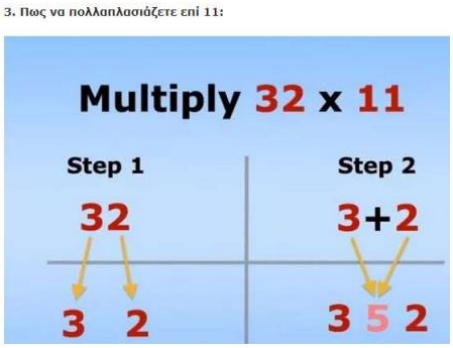
(*athroismata2tetragwnwn.psc*)

19. Πρόγραμμα που επαληθεύει το παρακάτω **τρικ**:



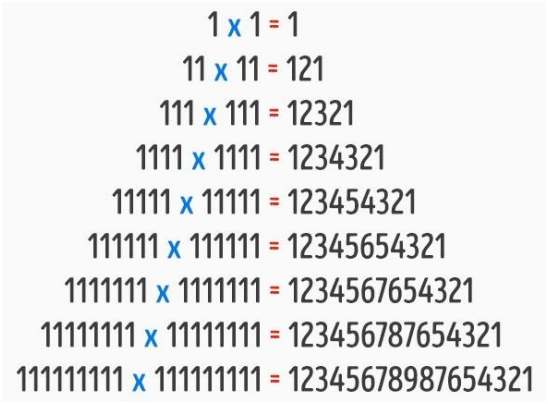
(*pollaplasiasmos_dipshfiwn.psc*)

20. Πρόγραμμα που επαληθεύει το παρακάτω **τρικ**:



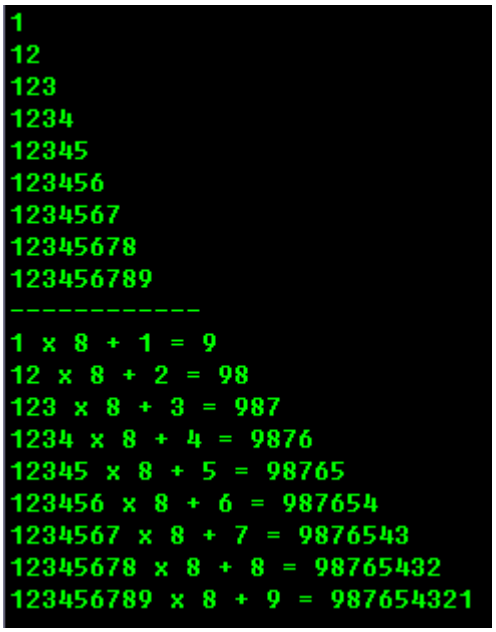
(*pollaplasiasmos_dipshfioy_me_to_11.psc*)

21. Πολλαπλασιάζουμε έναν οποιοδήποτε θετικό άρτιο αριθμό με το 6. Το αποτέλεσμα και ο αρχικός αριθμός, έχουν πάντα **το ίδιο τελευταίο ψηφίο**. Πρόγραμμα που επαληθεύει το παραπάνω για τους άρτιους στο [2, 100]. (*pollaplasiasmos_me_to_6.psc*)
22. Αν διαιρέσουμε οποιονδήποτε θετικό **τριψήφιο ακέραιο με όλα τα ψηφία του ίσα** (π.χ. 888) με το άθροισμα των ψηφίων του, το αποτέλεσμα είναι πάντα το 37. Πρόγραμμα που επαληθεύει το παραπάνω για όλους τους αριθμούς (111, 222, …, 999). (*tripshfioi_xxx.psc*)
23. Πρόγραμμα που **εμφανίζει** το παρακάτω:



(*pollaplasiasmos_111.psc*)

24. Πρόγραμμα που **εμφανίζει** τα παρακάτω:



(*p123456789.psc*)

25. Όλα τα **πολλαπλάσια του 3** έχουν άθροισμα ψηφίων που και αυτό είναι πολλαπλάσιο του 3. Πρόγραμμα που το αποδεικνύει για τα 1000 πρώτα πολλαπλάσια του 3. (*pollaplasiaToy3.psc*)

```
Το 2976  έχει άθροισμα ψηφίων:  24 (πολ/σιο του 3)
Το 2979  έχει άθροισμα ψηφίων:  27 (πολ/σιο του 3)
Το 2982  έχει άθροισμα ψηφίων:  21 (πολ/σιο του 3)
Το 2985  έχει άθροισμα ψηφίων:  24 (πολ/σιο του 3)
Το 2988  έχει άθροισμα ψηφίων:  27 (πολ/σιο του 3)
Το 2991  έχει άθροισμα ψηφίων:  21 (πολ/σιο του 3)
Το 2994  έχει άθροισμα ψηφίων:  24 (πολ/σιο του 3)
Το 2997  έχει άθροισμα ψηφίων:  27 (πολ/σιο του 3)
Το 3000  έχει άθροισμα ψηφίων:  3 (πολ/σιο του 3)
```

26. Εισαγωγή χαρακτήρα προς χαρακτήρα των συμβόλων μιας αριθμητικής έκφρασης μέχρι να δοθεί ο χαρακτήρας '#' ή όταν δοθεί δεξιά παρένθεση ')' που δεν έχει αντίστοιχη αριστερή. Να ελεγχθεί εάν η έκφραση έχει **ισορροπημένες παρενθέσεις** καθώς και το πλήθος των αριστερών και δεξιών παρενθέσεων που περιέχει (*parentheseis.psc*). Π.χ.

```
(
1
+
2
)
*
3
#
Ισορροπημένες
αριστερές παρενθέσεις:  1 δεξιές παρενθέσεις:  1
```

```
(
1
+
2
)
)
Μη ισορροπημένες
αριστερές παρενθέσεις:  1 δεξιές παρενθέσεις:  2
```

```
(
1
+
2
#
Μη ισορροπημένες
αριστερές παρενθέσεις:  1 δεξιές παρενθέσεις:  0
```

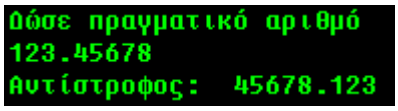
27. (Άσκηση 2.18 Βιβλίο Πληροφορικής Κύπρου) Η Αριάδνη έχει **κρύψει τον κωδικό** της μέσα σε ένα κείμενο. Δεν είναι σίγουρη όμως αν το έκανε σωστά. Να δημιουργήσετε πρόγραμμα το οποίο να δέχεται χαρακτήρα προς χαρακτήρα τον κωδικό (4 χαρακτήρες) και το κείμενο (10 χαρακτήρες). Αν ο κωδικός υπάρχει μέσα στο κείμενο να εμφανίζει τις θέσεις των χαρακτήρων του κειμένου που εμφανίζονται οι χαρακτήρες του κωδικού. Διαφορετικά, να εμφανίζει κατάλληλο μήνυμα. Οι χαρακτήρες του κωδικού πρέπει να εμφανίζονται μέσα στη συμβολοσειρά με τη σωστή σειρά, αλλά όχι κατά ανάγκη ο ένας δίπλα από τον άλλο (*krymmenos_kwdikos.psc*). π.χ. κωδικός='STOP', κείμενο='ABDSTKLOTP'➔

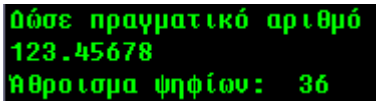
```
Βρέθηκε:
4
5
8
10
```

28. (Άσκηση 6.19 Βιβλίο Πληροφορικής Κύπρου) Να δημιουργήσετε πρόγραμμα το οποίο να δέχεται 10 ακέραιους αριθμούς και να τους **ταξινομεί** σε αύξουσα σειρά, **με βάση το ψηφίο των μονάδων τους**. Αν το ψηφίο των μονάδων των δύο αριθμών είναι το ίδιο, τότε να εμφανίζεται πρώτος ο μικρότερος από τους δύο αριθμούς (*sortByLastDigit.psc*). π.χ. 890 11 42 562 9342 25 455 78 9 109

29. (Άσκηση 6.25 Βιβλίο Πληροφορικής Κύπρου) Να δημιουργήσετε πρόγραμμα το οποίο δέχεται **5 κλάσματα** με τη μορφή δύο θετικών ακεραίων και να τα εμφανίζει κατά αύξουσα σειρά. Σε περίπτωση που δύο κλάσματα είναι ίσα, δίνεται προτεραιότητα σε εκείνο με τον μικρότερο αριθμητή (*sortKlasmata.psc*). π.χ.

30. Πρόγραμμα που διαβάζει έναν πραγματικό αριθμό και εμφανίζει τον πραγματικό αριθμό που προκύπτει **αν αντιστραφεί το ακέραιο με το δεκαδικό μέρος** του αρχικού (*antistrofh_akeraioy_dekadikoy_meroys.psc*).

Π.χ. 

31. Πρόγραμμα που διαβάζει έναν πραγματικό αριθμό και εμφανίζει το **άθροισμα των ψηφίων** του (*athroisma_pshfiwn_pragmatikoy.psc*). π.χ. 

32. Δίνεται το παρακάτω τμήμα κώδικα:

```
Διάβασε x
Αν (x mod 2 = 0) τότε
        Γράψε x
Αλλιώς
        Γράψε -x
ΤέλοςΑν
```

Συμπληρώστε το κενό ώστε να προκύψει ισοδύναμος κώδικας:

```
Διάβασε x
Γράψε ........
```

(*grifos_kenoy01.psc*)

33. Καταχωρείστε σε πίνακα Ν[9] τα 9 (χαρτο)νομίσματα του €: 500, 200, ..., 5, 2, 1. Εύρεση **ελάχιστου αριθμού κερμάτων και χαρτονομισμάτων** που ισοδυναμούν με ένα ποσό σε € (*minimum_coins.psc*). Π.χ.

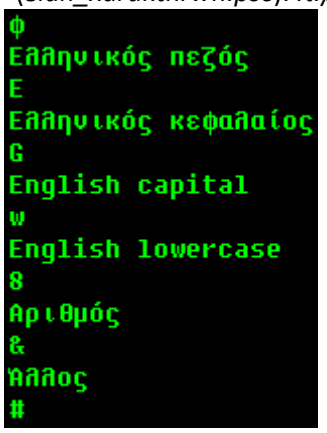
34. Να γραφεί πρόγραμμα που διαβάζει μία ακολουθία χαρακτήρων με τις τιμές W/B και την εμφανίζει συμπιεσμένη με βάση τον **αλγόριθμο συμπίεσης** Run Length Encoding (RLE). Να σταματάει όταν δοθεί ο χαρακτήρας '#' (*RLE.psc*)

π.χ. για είσοδο:
WWWWWWWWWWWWBWWWWWWWWWWWWBBBWWWWWWWWWWWWWWWWWWWWWBWWWWWWWWWWWWW#
έξοδος:
12W
1B
12W
3B
24W

1B
14W

35. Να γραφεί πρόγραμμα που διαβάζει μία ακολουθία χαρακτήρων (έναν προς έναν) και εμφανίζει το **είδος του χαρακτήρα**:
    - πεζός Ελληνικός (α-ω),
    - κεφαλαίος Ελληνικός (Α-Ω),
    - πεζός Αγγλικός (a-z),
    - κεφαλαίος Αγγλικός (A-Z),
    - αριθμός (0-9),
    - άλλος.

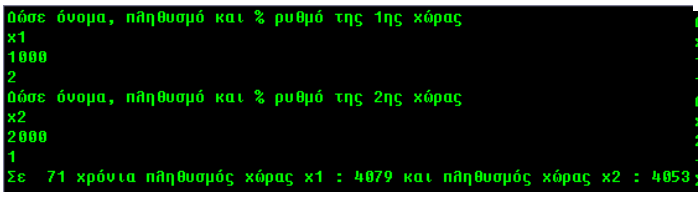
    Να σταματάει όταν δοθεί ο χαρακτήρας: '#' (*eidh_xarakthrwn.psc*). π.χ.

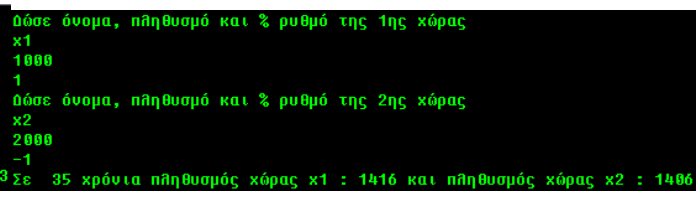


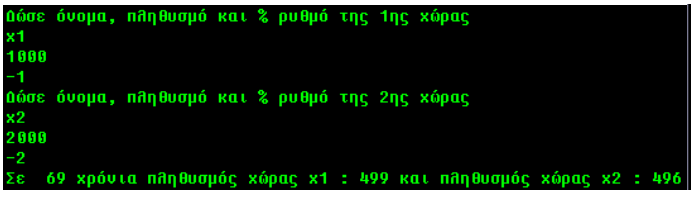
36. Να γραφεί πρόγραμμα που διαβάζει την τρέχουσα **ώρα** σε μορφή: ώρες(0-23)-λεπτά(0-59) και εμφάνιση της **λεκτικής περιγραφής** της ως εξής (*time.psc*)

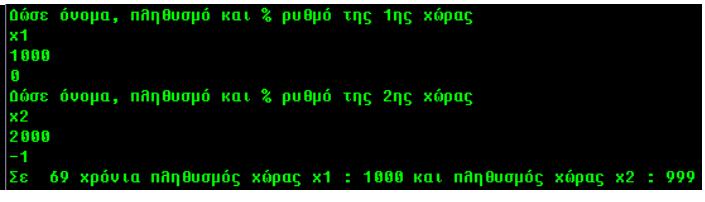
    00:10 ➔ 12 και 10, 01:30 ➔ 1 και μισή
    02:40 ➔ 3 παρά 20, 03:00 ➔ 3 ακριβώς
    13:05 ➔ 1 και 5, 04:15 ➔ 4 και τέταρτο
    20:45 ➔ 9 παρά τέταρτο 00:45 ➔1 παρά τέταρτο

37. Να γραφεί πρόγραμμα που διαβάζει για **2 χώρες**: τα ονόματα, τους σημερινούς πληθυσμούς και τους % ετήσιους ρυθμούς αύξησής τους. Να εμφανίζει σε πόσα χρόνια η μικρότερη σε πληθυσμό χώρα θα ξεπεράσει τη μεγαλύτερη ή το μήνυμα "δεν θα συμβεί ποτέ" (*xwres_plithismoi2.psc*). π.χ.









```
Δώσε όνομα, πληθυσμό και % ρυθμό της 1ης χώρας
x1
1000
1
Δώσε όνομα, πληθυσμό και % ρυθμό της 2ης χώρας
x2
2000
2
Δεν θα συμβεί ποτέ
```

```
Δώσε όνομα, πληθυσμό και % ρυθμό της 1ης χώρας
x1
1000
-2
Δώσε όνομα, πληθυσμό και % ρυθμό της 2ης χώρας
x2
2000
-1
Δεν θα συμβεί ποτέ
```

38. Να γραφεί πρόγραμμα που εμφανίζει όλες (13.983.816) τις πιθανές **τυχερές εξάδες κλήρωσης του ΛΟΤΤΟ**. Το ΛΟΤΤΟ κληρώνει 6 αριθμούς στο διάστημα 1-49. (*lotto_syndyasmoi.psc*)
39. Να γραφεί πρόγραμμα που διαβάζει έναν πραγματικό αριθμό και υπολογίζει και εμφανίζει το **ακέραιο μέρος** του χωρίς την χρήση της εντολής A_M( ). (*akeraio_meros.psc*)

40. Ένας Ινδός μαθηματικός υπολόγισε την **τιμή του π** σύμφωνα με τη σχέση:

$$\pi = \sqrt{12}\left(1 - \frac{1}{3\cdot 3} + \frac{1}{5\cdot 3^2} - \frac{1}{7\cdot 3^3} + \frac{1}{9\cdot 3^4} - \frac{1}{11\cdot 3^5} + \ldots\right)$$

Να γραφεί πρόγραμμα που διαβάζει έναν ακέραιο n και υπολογίζει την τιμή του π έως και τον όρο

$$\frac{1}{(2\cdot n+1)\cdot 3^n}$$

Η τιμή του π με 20 δεκαδικά ψηφία είναι: 3.14159265358979323846. (*ypologismos_pi.psc*) Παραδείγματα: n = 10➔π = 3.14159330450308,n = 20➔π = 3.14159265359563, n = 30 ➔π = 3.14159265358979

41. Να γραφεί πρόγραμμα που διαβάζει έναν ακέραιο αριθμό και εμφανίζει αν είναι **άρτιος ή περιττός** χωρίς να κάνει χρήση των αριθμητικών τελεστών της διαίρεσης (div, mod και /). (*artios_perittos.psc, artios_perittos2.psc*)
42. Να γραφεί πρόγραμμα που διαβάζει ένα θετικό ακέραιο αριθμό και κάνει **καταμέτρηση των ψηφίων** του π.χ. ως εξής (*syxnothtes_pshfiwn_arithmoy.psc*):

```
457265200099
Ψηφίο: 0 Πλήθος: 3
Ψηφίο: 2 Πλήθος: 2
Ψηφίο: 4 Πλήθος: 1
Ψηφίο: 5 Πλήθος: 2
Ψηφίο: 6 Πλήθος: 1
Ψηφίο: 7 Πλήθος: 1
Ψηφίο: 9 Πλήθος: 2
```

43. Να γραφεί πρόγραμμα που διαβάζει ένα θετικό ακέραιο αριθμό και εμφανίζει τον αριθμό με **τα ψηφία** του αρχικού **σε φθίνουσα διάταξη**. π.χ. (*pshfia_arithmoy_se_fthinoysa.psc*):

```
3287463219902341
99876443332221100
```

44. Να γραφεί πρόγραμμα που διαβάζει έναν θετικό ακέραιο αριθμό (διασφάλιση εισόδου) και ελέγχει αν είναι **καρκινικός αριθμός** ή όχι εμφανίζοντας σχετικό μήνυμα. Καρκινικοί ονομάζονται οι αριθμοί που μπορούν να διαβαστούν και από τα δεξιά και από τα αριστερά π.χ. 232, 4554, 12321 κλπ. (*karkinikos_arithmos.psc*)
45. Ο αριθμός 202 ονομάζεται **super παλινδρομικός** επειδή και αυτός και το τετράγωνό του 202^2 = 40804 είναι παλινδρομικοί αριθμοί (δηλαδή διαβάζονται το ίδιο είτε από αριστερά είτε από δεξιά). Να γραφεί πρόγραμμα που θα εμφανίζει όλους τους super παλινδρομικούς αριθμούς από το 1 μέχρι το 1111 (*super_palindromikoi.psc*). Υπόδειξη: φτιάξτε τη *Συνάρτηση* palindromikos(x): Λογική

```
1 ( 1 )
2 ( 4 )
3 ( 9 )
11 ( 121 )
22 ( 484 )
101 ( 10201 )
202 ( 40804 )
212 ( 44944 )
111 ( 12321 )
121 ( 14641 )
1001 ( 1002001 )
1111 ( 1234321 )
```

46. Να γραφεί πρόγραμμα που διαβάζει έναν πίνακα ακεραίων μονοδιάστατο 10 θέσεων και υπολογίζει την **επικρατούσα τιμή**, δηλαδή την τιμή(ές) που εμφανίζεται τις περισσότερες φορές. (*epikratoysa_timh.psc*) π.χ. με είσοδο των τιμών: 7, 4, 3, 7, 3, 3, 6, 7, 1, 9➔Επικρατούσα: 3, Επικρατούσα: 7 (Υπόδειξη: ταξινομήστε τον πίνακα)
47. Να γραφεί πρόγραμμα που διαβάζει έναν πίνακα ακεραίων μονοδιάστατο 10 θέσεων και υπολογίζει τις **3 μεγαλύτερες διαφορετικές τιμές** (αν υπάρχουν). (*max_different.psc*) Παραδείγματα εκτέλεσης:

1 1 1 2 2 2 2 2 2 3 max= 3 max= 2 max= 1	1 1 1 2 2 2 2 2 2 2 max= 2 max= 1	1 1 1 1 1 1 1 1 1 1 max= 1	1 2 3 4 5 7 7 8 10 10 max= 10 max= 8 max= 7

48. Να γραφεί πρόγραμμα που διαβάζει έναν θετικό πραγματικό αριθμό και υπολογίζει την **τετραγωνική ρίζα** του χωρίς χρήση της συνάρτησης T_P(). (*proseggish_T_P.psc*)
49. Να γραφεί πρόγραμμα που παρακολουθεί **τα στατιστικά ενός παίκτη του μπάσκετ** κατά τη διάρκεια ενός παιγνιδιού. Το πρόγραμμα θα παίρνει τα εξής στοιχεία:
    '1' κάθε φορά που ο παίκτης επιτυγχάνει καλάθι με ελεύθερη βολή,
    '2' κάθε φορά που ο παίκτης επιτυγχάνει δίποντο,
    '3' κάθε φορά που ο παίκτης επιτυγχάνει τρίποντο και
    '4' κάθε φορά που ο παίκτης κάνει φάουλ.

    Πληκτρολογώντας '0' δείχνουμε στο πρόγραμμα ότι τελείωσε το παιχνίδι. Όταν ο παίκτης συμπληρώσει πέντε φάουλ, θα πρέπει να βγαίνει το μήνυμα: "ΒΓΑΙΝΕΙ ΕΞΩ ΜΕ 5 ΦΑΟΥΛ". Όταν τελειώσει η συμμετοχή του παίκτη –είτε λόγω αποβολής είτε λόγω τέλους του παιχνιδιού– θα γράφεται στην οθόνη η στατιστική του. (*basket_statistika.psc*). **Προέκταση**: οι επιλογές 1-3 να δίνονται για κάθε βολή/δίποντο/τρίποντο και να εισάγεται επιπλέον (1/0) κατά πόσο ήταν εύστοχη/άστοχη προσπάθεια, ώστε τελικά να βγαίνουν και τα αντίστοιχα % ποσοστά ευστοχίας.
50. Να γραφεί πρόγραμμα που διαβάζει σε πίνακα A[N], N ακέραιες τιμές και ελέγχει (ΝΑΙ/ΟΧΙ) εάν είναι **όλες διαφορετικές** μεταξύ τους. (*ola_diaforetika.psc, ola_diaforetika2.psc*). Π.χ. για N=5:

```
1
2
3
5
6
different values: 5
All Different!
```

```
1
2
3
1
2
different values: 3
```

51. Να γραφεί πρόγραμμα που εμφανίζει τους 256 (2^8) διαφορετικούς **δυαδικούς αριθμούς με 8 bits** (*dyadiko_systhma_8bits.psc*):

```
1 . 0 0 0 0 0 0 0 0
2 . 0 0 0 0 0 0 0 1
3 . 0 0 0 0 0 0 1 0
4 . 0 0 0 0 0 0 1 1
5 . 0 0 0 0 0 1 0 0
6 . 0 0 0 0 0 1 0 1
7 . 0 0 0 0 0 1 1 0
8 . 0 0 0 0 0 1 1 1
...
248 . 1 1 1 1 1 0 1 1 1
249 . 1 1 1 1 1 1 0 0 0
250 . 1 1 1 1 1 1 0 0 1
251 . 1 1 1 1 1 1 0 1 0
252 . 1 1 1 1 1 1 0 1 1
253 . 1 1 1 1 1 1 1 0 0
254 . 1 1 1 1 1 1 1 0 1
255 . 1 1 1 1 1 1 1 1 0
256 . 1 1 1 1 1 1 1 1 1
```

52. Να γραφεί πρόγραμμα (*diagwnismos_tragoydioy.psc*) που προσομοιώνει έναν **διαγωνισμό τραγουδιού** στον οποίο συμμετέχουν 5 χώρες. Το κύριο πρόγραμμα να καλεί τις παρακάτω διαδικασίες:
    - **Arxikopoihsh**(Xwres, Bathmologies) η οποία αρχικοποιεί τον πίνακα Xwres με τα ονόματα 5 χωρών και τον παράλληλο πίνακα Bathmologies με το μηδέν(0)
    - **Bathmologhsh**(Xwres, Bathmologies) η οποία ζητάει από κάθε χώρα τον αριθμό των χωρών (1-5) στις οποίες δίνει 3 βαθμούς, το 10 το 11 και το 12. Η εισαγωγή να γίνεται με έλεγχο εγκυρότητας, ώστε:
        - ο αριθμός της προς βαθμολόγηση χώρας να είναι 1-5
        - να μην επιτρέπεται η αυτοβαθμολόγηση (βαθμολόγηση της χώρας που βαθμολογεί)
        - να μην επιτρέπεται η επαναβαθμολόγηση (πολλαπλή βαθμολόγηση της ίδιας χώρας)

```
Bathmologies ths xwras  Greece
1 . Greece : 0
2 . England : 0
3 . France : 0
4 . Italy : 0
5 . Spain : 0
Poy dineis th bathmologia  10 (1- 5 )?
```

      Να ενημερώνει τον πίνακα Bathmologies με τις συνολικές βαθμολογίες των χωρών.
    - **Taksinomhsh**(Xwres, Bathmologies) η οποία ταξινομεί κατά φθίνουσα σειρά τον πίνακα Bathmologies και παράλληλα τον πίνακα Xwres
    - **Emfanish**(Xwres, Bathmologies) η οποία εμφανίζει την τελική κατάταξη των χωρών

```
Telikh katataksh
1 . France : 46
2 . England : 43
3 . Greece : 41
4 . Italy : 24
5 . Spain : 11
```

53. Να δημιουργήσετε πρόγραμμα το οποίο να εμφανίζει, να μετράει και να αθροίζει όλους τους τριψήφιους θετικούς ακέραιους αριθμούς των οποίων **τουλάχιστον δύο από τα ψηφία τους είναι ο αριθμός 8** (AtLeast2Eights.psc) `cnt =  29 sum =  22667`
54. Να γραφεί πρόγραμμα το οποίο να εμφανίζει τους 18 θετικούς ακέραιους (1-1000) των οποίων **το γινόμενο των ψηφίων τους είναι το διπλάσιο του αθροίσματός τους** (*ginomeno_pshfiwn_2plasio_athroismatos.psc*): 36, 44, 63, 138, 145, 154, 183, 224, 242, 318, 381, 415, 422, 451, 514, 541, 813, 831
55. **Πολλαπλασιαστική επιμονή** ενός ακέραιου είναι ο αριθμός των βημάτων που απαιτούνται ώστε το γινόμενο των ψηφίων του να είναι μονοψήφιος αριθμός. Π.χ. η πολλαπλασιαστική επιμονή του αριθμού 679 είναι 5, διότι: 679 → 6*7*9=378 → 3*7*8=168 → 1*6*8=48 → 4*8=32 → 3*2=6. 54. Να γραφεί πρόγραμμα (*pollaplasiastikh_epimonh.psc*) το οποίο να διαβάζει έναν θετικό ακέραιο και να εμφανίζει την πολλαπλασιαστική επιμονή του καθώς και τα βήματα υπολογισμού της:

```
679
--------------------
1 . 378
2 . 168
3 . 48
4 . 32
5 . 6
--------------------
pollaplasiastikh_epimonh =  5
```

Προέκταση: βρείτε τον θετικό ακέραιο (≤ 100) με τη μεγαλύτερη πολλαπλασιαστική επιμονή. (*pollaplasiastikh_epimonh2.psc*)

```
max pollaplasiastikh_epimonh =  4 twn arithmwn:
77
```

56. Να γραφεί πρόγραμμα το οποίο να εμφανίζει τους 56 θετικούς 3ψήφιους ακέραιους με **όλα τα ψηφία τους να αποτελούν διαιρέτες του αριθμού** (*ola_ta_pshfia_diairoyn_ton_arithmo.psc*)

```
1 . 111      45 . 672
2 . 112      46 . 728
3 . 115      47 . 735
4 . 122      48 . 777
5 . 124      49 . 784
6 . 126      50 . 816
7 . 128      51 . 824
8 . 132      52 . 848
9 . 135      53 . 864
10 . 144     54 . 888
11 . 155     55 . 936
12 . 162 ... 56 . 999
```

57. Στα μαθηματικά η σειρά **Fibonacci** είναι η εξής: 0, 1, 1, 2, 3, 5, 8, 13, 21… Εξ ορισμού, οι δύο πρώτοι αριθμοί της σειράς είναι το 0 και το 1, ενώ καθένας από τους επόμενους αριθμούς είναι το άθροισμα των δύο προηγούμενων. Να δημιουργήσετε πρόγραμμα (*fibonacci.psc*) το οποίο να διαβάζει έναν ακέραιο αριθμό Ν και να εμφανίζει τους πρώτους Ν αριθμούς της σειράς Fibonacci στην οθόνη.
Παράδειγμα εισόδου: 10, Παράδειγμα εξόδου: 0 1 1 2 3 5 8 13 21 34
58. Αριθμός **ariadne** ορίζεται ο αριθμός που το άθροισμα του πιο σημαντικού (ψηφίο στα αριστερά) και του λιγότερου σημαντικού ψηφίου (ψηφίο στα δεξιά) ισούται με το άθροισμα των υπόλοιπων ψηφίων (π.χ. 121, 1335). Να γραφεί πρόγραμμα (*ariadne.psc*) το οποίο να εμφανίζει
    - τους 45 θετικούς 3ψήφιους ακέραιους που είναι αριθμοί ariadne:

```
1 . 110      34 . 583
2 . 121      35 . 594
3 . 132      36 . 660
4 . 143      37 . 671
5 . 154      38 . 682
6 . 165      39 . 693
7 . 176      40 . 770
8 . 187      41 . 781
9 . 198      42 . 792
10 . 220     43 . 880
11 . 231     44 . 891
12 . 242 ... 45 . 990
```

    - τους 63 θετικούς 4ψήφιους ακέραιους (από 1000 - 2000) που είναι αριθμοί ariadne:

```
1 . 1010
2 . 1021
3 . 1032
4 . 1043
5 . 1054
6 . 1065
7 . 1076
8 . 1087
9 . 1098
10 . 1100
11 . 1111
12 . 1122
...
52 . 1627
53 . 1638
54 . 1649
55 . 1706
56 . 1717
57 . 1728
58 . 1739
59 . 1807
60 . 1818
61 . 1829
62 . 1908
63 . 1919
```

59. Να γραφεί πρόγραμμα το οποίο για όλες τις πιθανές **τιμές ρίψης 3 ζαριών**, να εμφανίζει: α) τη μεγαλύτερη συχνότητα εμφάνισης αθροίσματος, β) τα πιο συχνά αθροίσματα και γ) τις τιμές των ζαριών με τα αθροίσματα αυτά (*tria_zaria.psc*).

```
max syxnothta athroismatos 3 zariwn =  27 twn athroismatwn:
10
11
Times 3 zariwn me ta pio syxna athroismata:
-----------------------------------------
1 + 3 + 6 + = 10
1 + 4 + 5 + = 10
1 + 4 + 6 + = 11
1 + 5 + 4 + = 10
1 + 5 + 5 + = 11
...
5 + 4 + 1 + = 10
5 + 4 + 2 + = 11
5 + 5 + 1 + = 11
6 + 1 + 3 + = 10
6 + 1 + 4 + = 11
6 + 2 + 2 + = 10
6 + 2 + 3 + = 11
6 + 3 + 1 + = 10
6 + 3 + 2 + = 11
6 + 4 + 1 + = 11
```

60. Να συμπληρώσετε το **κενό** ώστε
    - ο παρακάτω κώδικας να γεμίζει τον Α[100] με τις τιμές (*grifosA100.psc*):

A	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10	...	1	2	3	4	5	6	7	8	9	10
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20		91	92	93	94	95	96	97	98	99	100

```
ΓΙΑ i ΑΠΟ 1 ΜΕΧΡΙ 100
  A[i] <-- ...
ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ
```

    - ο παρακάτω κώδικας να γεμίζει τον Α[5, 5] με τις τιμές (*grifosA5x5.psc*):

A

1	2	3	4	5
2	3	4	5	6
3	4	5	6	7
4	5	6	7	8
5	6	7	8	9

```
ΓΙΑ i ΑΠΟ 1 ΜΕΧΡΙ  5
  ΓΙΑ j ΑΠΟ 1 ΜΕΧΡΙ 5
    A[i,j] <-- ...
  ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ
ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ
```

61. Να γραφεί πρόγραμμα το οποίο εμφανίζει τους 10 πρώτους όρους των παρακάτω **ακολουθιών** (*series.psc*):
    - 1, 3, 6, 10 ,15, 21, 28, 36, 45, 55
    - 1, 2, 6, 24, 120, 720, 5040, 40320, 362880, 3628800
    - 5, 10, 13, 26, 29, 58, 61, 122, 125, 250
    - 1, 2, 4, 8, 16, 32, 64, 128, 256, 512
    - 8, 24, 12, 36, 18, 54, 27, 81, 40, 120
62. Να γραφεί πρόγραμμα το οποίο γενικεύει τον αλγόριθμο του **πολλαπλασιασμού αλά ρωσικά** ώστε να λειτουργεί και με είσοδο δύο πραγματικές τιμές. Επιτρέπονται επιπλέον οι πράξεις * 10, / με δύναμη του 10 και ^ με βάση το 10 (*pollaplasiasmos_ala_rwsika_gia_pragmatikoys.psc*):

```
Δώσε 2 πραγματικές τιμές:
1.23
-4.567
Γινόμενο =  -5.6174100000
```

63. Να γραφεί πρόγραμμα το οποίο υλοποιεί τη **μέθοδο πολλαπλασιασμού "Πλέγμα"** (Grid Method https://www.mathstudies.eu/single-post/multiplication01): στη μέθοδο αυτή σπάμε τους αριθμούς σε μονάδες, δεκάδες, εκατοντάδες, κ.λ.π. Στη συνέχεια, τοποθετούμε τους αριθμούς που προκύπτουν σε ένα πλέγμα και συμπληρώνουμε τα τετράγωνα κάνοντας τους αντίστοιχους πολλαπλασιασμούς. Το αποτέλεσμα προκύπτει αν αθροίσουμε τους αριθμούς όλων των τετραγώνων (*multiplication_grid_method.psc*). π.χ. για τον πολλαπλασιασμό 285x46:

Βήμα 1. Φτιάχνουμε το πλέγμα:

x	200	80	5
40			
6			

Βήμα 2. Βρίσκουμε τα γινόμενα:

x	200	80	5
40	8000	3200	200
6	1200	480	30

Βήμα 3. Προσθέτουμε όλα τα επιμέρους γινόμενα: 8000+3200+200+1200+480+30 = 13110

```
Δώσε 2 θετικές ακέραιες τιμές
285
46
Πρόσθεση γινομένου  200 x 40 = 8000
Πρόσθεση γινομένου  200 x 6 = 1200
Πρόσθεση γινομένου  80 x 40 = 3200
Πρόσθεση γινομένου  80 x 6 = 480
Πρόσθεση γινομένου  5 x 40 = 200
Πρόσθεση γινομένου  5 x 6 = 30
------------------------------
grid method result =  13110
classic result (*) =  13110
```

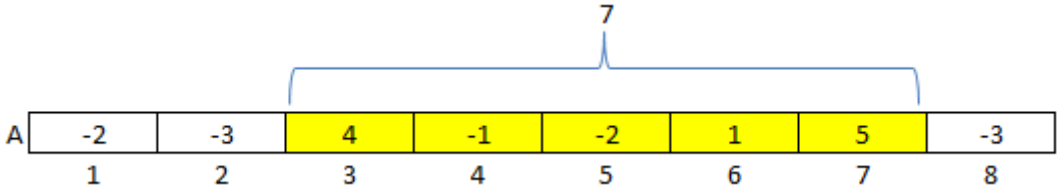
Για να πολλαπλασιάσουμε δεκαδικούς αριθμούς, σπάμε τα δεκαδικά μέρη σε δέκατα, εκατοστά κ.λ.π. και ακολουθούμε παρόμοια διαδικασία. Παρακάτω παρουσιάζουμε τον πολλαπλασιασμό 3.15x5.8:

x	3	1/10	5/100
5	15	0.5	2.5
8/10	2.4	0.08	0.04

```
Δώσε 2 τιμές
3.15
5.8
Πρόσθεση γινομένου  3 x 5 = 15
Πρόσθεση γινομένου  3 x 0.80 = 2.40
Πρόσθεση γινομένου  0.10 x 5 = 0.50
Πρόσθεση γινομένου  0.10 x 0.80 = 0.08
Πρόσθεση γινομένου  0.05 x 5 = 0.25
Πρόσθεση γινομένου  0.05 x 0.80 = 0.04
------------------------------
grid method result =  18.27
classic result (*) =  18.27
```

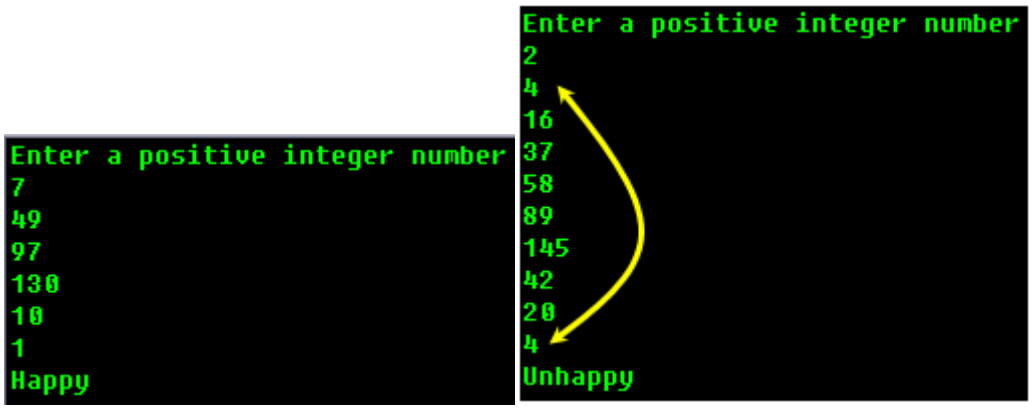
Το αποτέλεσμα δίνεται από το άθροισμα 15+0.5+0.25+2.4+0.08+0.04 = 18.27

64. Να γραφεί πρόγραμμα το οποίο διαβάζει έναν ακέραιο Α[8] και εμφανίζει τον συνεχόμενο **υποπίνακά** του με το μέγιστο άθροισμα (έστω μόνο ένας) (*maxSubArray.psc*). Π.χ.




```
maxSum =  7 form index 3 to index 7 of the elements:
4
-1
-2
1
5
```

65. «**Ευτυχής**» αριθμός (**happy number**) είναι ένας θετικός ακέραιος που εάν αθροίσουμε τα τετράγωνα των ψηφίων του και επαναλάβουμε τη διαδικασία με το αποτέλεσμα, θα καταλήξει σε άθροισμα ίσο με τη μονάδα (1). Αν αντιθέτως προκύψει αποτέλεσμα που έχει ξαναυπολογισθεί σε προηγούμενη επανάληψη, τότε ο αριθμός είναι «**Δυστυχής**». Να γραφεί πρόγραμμα το οποίο διαβάζει έναν ακέραιο και τον χαρακτηρίζει ως «Ευτυχή» ή «Δυστυχή» (*HappyNumber.psc*). Π.χ.



Οι 143 ευτυχείς αριθμοί από το 1 - 1.000 είναι:
1, 7, 10, 13, 19, 23, 28, 31, 32, 44, 49, 68, 70, 79, 82, 86, 91, 94, 97, 100, 103, 109, 129, 130, 133, 139, 167, 176, 188, 190, 192, 193, 203, 208, 219, 226, 230, 236, 239, 262, 263, 280, 291, 293, 301, 302, 310, 313, 319, 320, 326, 329, 331, 338, 356, 362, 365, 367, 368, 376, 379, 383, 386, 391, 392, 397, 404, 409, 440, 446, 464, 469, 478, 487, 490, 496, 536, 556, 563, 565, 566, 608, 617, 622, 623, 632, 635, 637, 638, 644, 649, 653, 655, 656, 665, 671, 673, 680, 683, 694, 700, 709, 716, 736, 739, 748, 761, 763, 784, 790, 793, 802, 806, 818, 820, 833, 836, 847, 860, 863, 874, 881, 888, 899, 901, 904, 907, 910, 912, 913, 921, 923, 931, 932, 937, 940, 946, 964, 970, 973, 989, 998, 1000

66. Στη θεωρία των αριθμών, ένας **ναρκισσιστικός αριθμός** (narcissistic number) είναι ένας ακέραιος αριθμός που ισούται με το άθροισμα των ψηφίων του υψωμένα στον αριθμό των ψηφίων του. π.χ. 153 = $1^3+5^3+3^3$, 1634 = $1^4+6^4+3^4+4^4$, 54748 = $5^5+4^5+7^5+4^5+8^5$. Οι ναρκισσιστικοί αριθμοί είναι: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 153, 370, 371, 407, 1634, 8208, 9474, 54748, 92727, 93084, 548834, ... Να γράψετε πρόγραμμα που εμφανίζει όλους τους ναρκισσιστικούς αριθμούς στο διάστημα 1-N, όπου N στοιχείο εισόδου (*Narcissistic_number.psc*).

$$153 = 1^3 + 5^3 + 3^3$$
$$370 = 3^3 + 7^3 + 0^3$$
$$371 = 3^3 + 7^3 + 1^3$$
$$407 = 4^3 + 0^3 + 7^3$$

```
1 Narcissistic
2 Narcissistic
3 Narcissistic
4 Narcissistic
5 Narcissistic
6 Narcissistic
7 Narcissistic
8 Narcissistic
9 Narcissistic
153 Narcissistic
370 Narcissistic
371 Narcissistic
407 Narcissistic
```

67. **Τετραγωνικοί** είναι οι ακέραιοι που ισούνται με το τετράγωνο κάποιου ακέραιου. Π.χ. 49 = 7^2. **Τριγωνικοί** είναι οι ακέραιοι που ισούνται με το άθροισμα 1+2+…+n για κάποιον ακέραιο n. Π.χ. 10 = 1+2+3+4. Να γραφεί πρόγραμμα το οποίο εμφανίζει τους N πρώτους (N: στοιχείο εισόδου) ακέραιους που είναι ταυτόχρονα και τετραγωνικοί και τριγωνικοί (*triangular_square_numbers.psc*). Π.χ. 36 = 6^2 = 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8

```
1 . 1 = 1 ^2 = Sum(1- 1 )
2 . 36 = 6 ^2 = Sum(1- 8 )
3 . 1225 = 35 ^2 = Sum(1- 49 )
4 . 41616 = 204 ^2 = Sum(1- 288 )
5 . 1413721 = 1189 ^2 = Sum(1- 1681 )
```

68. Σύμφωνα με το **Θεώρημα των Πυθαγόρειων Τριπλετών** (Pythagorean Triples Theorem) μπορούμε να πάρουμε κάθε ακέραια Πυθαγόρεια τριάδα (a, b, c: $\mathbf{a^2+b^2=c^2}$) χρησιμοποιώντας τους τύπους:

$$a = st, \qquad b = \frac{s^2 - t^2}{2}, \qquad c = \frac{s^2 + t^2}{2},$$

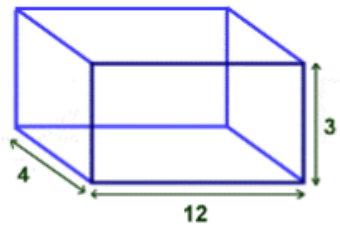
όπου s > t ≥ 1 οποιοιδήποτε περιττοί ακέραιοι αριθμοί χωρίς κοινούς παράγοντες. Π.χ. για μέγιστο s=9:

s	t	$a = st$	$b = \frac{s^2 - t^2}{2}$	$c = \frac{s^2 + t^2}{2}$
3	1	3	4	5
5	1	5	12	13
7	1	7	24	25
9	1	9	40	41
5	3	15	8	17
7	3	21	20	29
7	5	35	12	37
9	5	45	28	53
9	7	63	16	65

Να γραφεί πρόγραμμα το οποίο εμφανίζει τις Πυθαγόρειες τριπλέτες για δεδομένο μέγιστο s ως στοιχείο εισόδου (*PythagoreanTriplesTheorem.psc*)

```
s= 3 t= 1 a= 3 b= 4 c= 5
s= 5 t= 1 a= 5 b= 12 c= 13
s= 7 t= 1 a= 7 b= 24 c= 25
s= 9 t= 1 a= 9 b= 40 c= 41
s= 5 t= 3 a= 15 b= 8 c= 17
s= 7 t= 3 a= 21 b= 20 c= 29
s= 7 t= 5 a= 35 b= 12 c= 37
s= 9 t= 5 a= 45 b= 28 c= 53
s= 9 t= 7 a= 63 b= 16 c= 65
```

69. Ο όγκος ενός **ορθογώνιου παραλληλεπιπέδου** ισούται με το γινόμενο των τριών διαστάσεών του (μήκος x πλάτος x ύψος). Να γραφεί πρόγραμμα το οποίο διαβάζει τον όγκο ενός ορθογώνιου παραλληλεπιπέδου και εμφανίζει όλες τις πιθανές ακέραιες διαστάσεις του. (*ogkos_orthogwnioy_parallhlepipedoy.psc*)



Λίγη γεωμετρία:

Όγκος κουτιού: μήκος x πλάτος x ύψος = **12 x 4 x 3 = 144cm³**

```
10
1 . 1 1 10
2 . 1 2 5
3 . 1 5 2
4 . 1 10 1
5 . 2 1 5
6 . 2 5 1
7 . 5 1 2
8 . 5 2 1
9 . 10 1 1
```

```
12
1 . 1 1 12
2 . 1 2 6
3 . 1 3 4
4 . 1 4 3
5 . 1 6 2
6 . 1 12 1
7 . 2 1 6
8 . 2 2 3
9 . 2 3 2
10 . 2 6 1
11 . 3 1 4
12 . 3 2 2
13 . 3 4 1
14 . 4 1 3
15 . 4 3 1
16 . 6 1 2
17 . 6 2 1
18 . 12 1 1
```

70. Να γραφεί πρόγραμμα το οποίο διαβάζει έναν ακέραιο αριθμό (>99) και ελέγχει εάν είναι **αριθμός ariks**. Αριθμός ariks ορίζεται ο αριθμός που το άθροισμα του πιο σημαντικού (αριστερότερου) και του λιγότερου σημαντικού ψηφίου (δεξιότερου) ισούται με το άθροισμα των υπόλοιπων ψηφίων (π.χ. 121, 1335). (*ariks_number.psc*)

```
112349
ΑΛΗΘΗΣ
```

71. Ένας πίνακας A[N] λέγεται μονοκόρυφος εάν για κάποιο στοιχείο p (2=p<=N-1) έχουμε A[1]<A[2]<...<A[p] και A[p]>A[p+1]>...>A[N]. Να γραφεί πρόγραμμα που διαβάζει έναν ακέραιο A[10] και ελέγχει εάν είναι μονοκόρυφος ή όχι. Σε περίπτωση που είναι να εμφανίζει το στοιχείο p (monokoryfos1D.psc) π.χ.

```
A = [ 1   2   3   4   5   5   3   2   1   0   ]
oxi monokoryfos
```

```
A = [ 1   2   3   4   5   4   3   2   1   0   ]
monokoryfos sto stoixeio A[ 5 ]= 5
```

72. Να γραφεί πρόγραμμα που διαβάζει έναν θετικό ακέραιο και υπολογίζει τον ακέραιο που προκύπτει εάν το **1ο ψηφίο γίνει τελευταίο** (*prwtoPshfioTeleytaio.psc*).

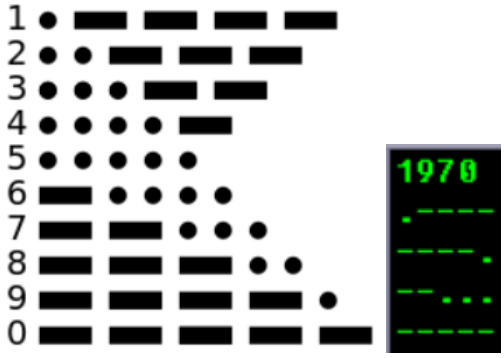
```
1234567890
2345678901
```

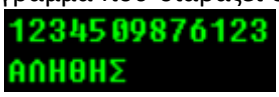
. Προέκταση: να κάνει αντιμετάθεση του 1ου με το τελευταίο ψηφίο:

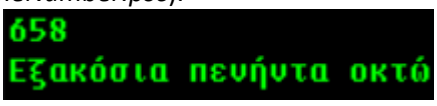
```
12345
52341
```

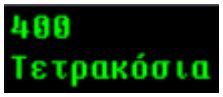
(*swapPrwtoPshfioMeTeleytaio.psc*)

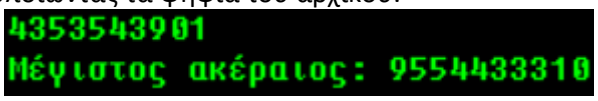
73. Να γραφεί πρόγραμμα που αρχικά γεμίζει έναν πίνακα χαρακτήρων MC[10] με τον **κώδικα Μορς** των ψηφίων 0-9 και στη συνέχεια διαβάζει έναν θετικό ακέραιο και εμφανίζει τα ψηφία του κωδικοποιημένα στον κώδικα Μορς (*morse_code.psc*).



74. Να γραφεί πρόγραμμα που διαβάζει έναν πραγματικό Α[10] και τον **ταξινομεί** κατά αύξουσα σειρά **με βάση το δεκαδικό τους μέρος**. Όσοι αριθμοί έχουν το ίδιο δεκαδικό μέρος να ταξινομούνται με βάση το ακέραιο μέρος τους. Π.χ. 2.3, 5, 1, 9.8, 5.9, 3.6, 2.7, 4, 8, 12.4 ➔ 1, 4, 5, 8, 2.30, 12.40, 3.60, 2.70, 9.80, 5.90 (*sortByDecimalPlace.psc*).
75. Να γραφεί πρόγραμμα που διαβάζει έναν θετικό ακέραιο και υπολογίζει τον ακέραιο που προκύπτει εάν **κάθε ψηφίο υψωθεί στο τετράγωνο**. Π.χ. 9119 ➔ 811181, 2483 ➔ 416649, 3212 ➔ 9414 (*squareDigits.psc*)
76. Να γραφεί πρόγραμμα που διαβάζει έναν θετικό ακέραιο και υπολογίζει τον ακέραιο που προκύπτει εάν αφαιρεθούν από τον αρχικό τα **επαναλαμβανόμενα συνεχόμενα ψηφία**. Π.χ. 1122222333 ➔ 123 (*removeSameConsequtiveDigits.psc*)
77. Ως **πανψηφιακός αριθμός** ονομάζεται ο φυσικός αριθμός στον οποίο περιέχονται τουλάχιστον μια φορά το κάθε ένα ψηφίο (0-9). 76. Να γραφεί πρόγραμμα που διαβάζει έναν θετικό ακέραιο και ελέγχει εάν είναι πανψηφιακός (*pandigitalNumber.psc*). π.χ. 

78. Να γραφεί πρόγραμμα που διαβάζει έναν ακέραιο Α[10] και τον **ταξινομεί με βάση τον αριθμό των ψηφίων** κάθε τιμής και κατά φθίνουσα σειρά, ενώ οι τιμές με τον ίδιο αριθμό ψηφίων τα ταξινομούνται με αύξουσα σειρά. Π.χ. 3242, 435, 4534, 2, 8, 9, 567, 23456, 98765, 23 ➔ 23456, 98765, 3242, 4534, 435, 567, 23, 2, 8, 9 (*sortByNumOfDigits.psc*)
79. **Ετερομήκιος** (heteromecic) είναι ένας ακέραιος που ισούται με το γινόμενο 2 συνεχόμενων ακεραίων. Είναι δηλ. της μορφής n*(n+1). Να γραφεί πρόγραμμα που διαβάζει έναν θετικό ακέραιο και ελέγχει εάν είναι ετερομήκιος εμφανίζοντας τους 2 αντίστοιχους συνεχόμενους ακέραιους (*heteromecicNumber.psc πιο αποδοτικά: heteromecicNumber2.psc*). π.χ. 9702 ==> Yes: 9702 = 98 * 99
80. Να γραφεί πρόγραμμα που διαβάζει έναν θετικό ακέραιο και τον εμφανίζει "**περιφραστικά**" (*lookAndSayNumber.psc*). π.χ. 122333444455555 => 1 1 s, 2 2 s, 3 3 s, 4 4 s, 5 5 s
81. Να γραφεί πρόγραμμα που διαβάζει έναν ακέραιο Α[N] και εντοπίζει τα στοιχεία του που είναι "**σημεία ισορροπίας**". Δηλ. το άθροισμα των στοιχείων στα αριστερά ισούται με το άθροισμα των στοιχείων στα δεξιά. π.χ. για τα στοιχεία 2,3,4,1,4,5 σημείο ισορροπίας είναι το 4ο (1) διότι τα στοιχεία στα αριστερά (2,3,4) και τα στοιχεία στα δεξιά (4,5) έχουν το ίδιο άθροισμα (9). (*balanceElementOfArray.psc*)
82. Να γραφεί πρόγραμμα που διαβάζει έναν θετικό ακέραιο τριψήφιο και εμφανίζει τη **λεκτική περιγραφή** του (*sayTheNumber.psc*).
π.χ. 



83. Να γραφεί πρόγραμμα που διαβάζει έναν θετικό ακέραιο και εμφανίζει τον μεγαλύτερο ακέραιο που μπορεί να φτιαχτεί χρησιμοποιώντας τα ψηφία του αρχικού.
π.χ. 

Υπόδειξη (λύση 1η): συλλέξτε τα ψηφία του ακέραιου σε πίνακα Α[100], ταξινομήστε τα κατά φθίνουσα σειρά και σχηματίστε τον ζητούμενο ακέραιο (*makeMaxNumber.psc*).

Υπόδειξη (λύση 2η): φτιάξτε τον πίνακα συχνοτήτων Α[10] με τις συχνότητες των ψηφίων του ακέραιου (Α[1]: οι άσσοι, …, Α[9]: τα εννιάρια, Α[10]: τα μηδενικά) και σχηματίστε τον ζητούμενο ακέραιο από τα εννιάρια μέχρι τους άσσους και τελευταία τα μηδενικά (*makeMaxNumber2.psc*)

84. Να γραφεί πρόγραμμα που γεμίζει έναν ακέραιο Α[Ν,Μ] (Ν και Μ: σταθερές) με τις τιμές 1,2,...,Ν*Μ κατά γραμμές και κατασκευάζει τον **επίκεντρο χάρτη** Χ[Ν,Μ] έτσι ώστε το κάθε στοιχείο Χ[i,j] να ισούται με το άθροισμα του στοιχείου Α[i,j] με όλα τα γειτονικά στοιχεία του (*epikentrosXarths.psc*). π.χ.

Α

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20

Χ

16	27	33	39	28
39	63	72	81	57
69	108	117	126	87
56	87	93	99	68

85. Να γραφεί πρόγραμμα που διαβάζει έναν ακέραιο Α[Ν] (Ν: σταθερά) και τον **ανακατεύει στο μέγιστο βαθμό**. Τοποθετεί δηλ. στο Α[1] το 1ο μεγαλύτερο, στο Α[2] το 1ο μικρότερο, στο Α[3] το 2ο μεγαλύτερο, στο Α[4] το 2ο μικρότερο κ.ο.κ. π.χ. εάν ο Α έχει τις τιμές: 1,2,3,4,5,6,7,8,9,10 καταλήγει με τις τιμές: 10,1,9,2,8,3,7,4,6,5 (*anakatemaPinaka.psc*)
86. Να γραφεί πρόγραμμα που διαβάζει έναν θετικό ακέραιο Ν και εμφανίζει όλους του **"δίδυμους" πρώτους** αριθμούς έως και το Ν. Δηλ. τους αριθμούς που είναι πρώτοι (διαιρούνται μόνο με το 1 και τον εαυτό τους) και διαφέρουν κατά 2. π.χ. οι "δίδυμοι" πρώτοι μέχρι το 200 είναι οι: 5 7, 11 13, 17 19, 29 31, 41 43, 59 61, 71 73, 101 103, 107 109, 137 139, 149 151, 179 181, 191 193, 197 199 (*prwtoi_didymoi_arithmoi.psc*)
87. Να γραφεί πρόγραμμα που διαβάζει έναν ακέραιο και ελέγχει αν στον αριθμό **κάθε ψηφίο εμφανίζεται μία και μοναδική φορά**. Για παράδειγμα ο αριθμός 38213 περιέχει 2 φορές το ψηφίο 3 ενώ στον αριθμό 953 κάθε ψηφίο εμφανίζεται μία και μοναδική φορά (*pshfia_apo_mia_fora.psc*).
88. **Ποια μέρα είναι Χριστούγεννα;** Έστω ΕΤ η χρονολογία που μας ενδιαφέρει. Χωρίζουμε το ΕΤ σε δύο μέρη: ονομάζουμε Α τις εκατονταετίες της χρονολογίας, δηλαδή τα 2 αριστερά ψηφία του ΕΤ. Ονομάζουμε Β τα υπόλοιπα χρόνια της χρονολογίας, δηλαδή τα 2 δεξιά ψηφία του ΕΤ. Διαιρούμε το Α με τον αριθμό 4 και κρατάμε μόνο το ακέραιο μέρος. Αυτό το ονομάζουμε Κ. Διαιρούμε το Β με τον αριθμό 4 και κρατάμε μόνο το ακέραιο μέρος. Αυτό το ονομάζουμε Λ. Για να βρούμε την ημέρα της εβδομάδας εφαρμόζουμε τον τύπο: Χ = 50 + Β + Κ + Λ - 2Α. Διαιρούμε το Χ με το 7 και το υπόλοιπο το ονομάζουμε Μ. Αν Μ = 0 σημαίνει ότι τα Χριστούγεννα θα πέσουν Κυριακή, αν Μ = 1 θα πέσουν Δευτέρα, ... κ.ο.κ. Να γραφεί πρόγραμμα που εμφανίζει την ημέρα των Χριστουγέννων από το 2000 μέχρι το 2025 (*christmas_day.psc*).

```
2020 Παρασκευή
2021 Σάββατο
2022 Κυριακή
2023 Δευτέρα
2024 Τετάρτη
2025 Πέμπτη
```

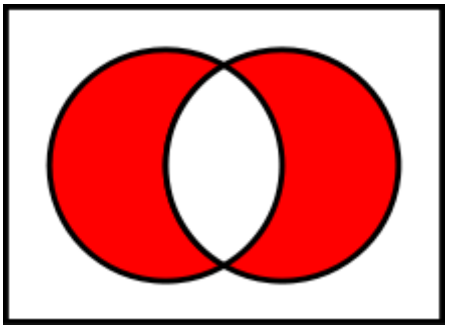
89. Ένας αριθμός έχει ένα **σημείο διακοπής (breakpoint)** εάν μπορεί να διαχωρισθεί με τέτοιο τρόπο ώστε τα ψηφία στην αριστερή πλευρά και τα ψηφία στη δεξιά πλευρά να έχουν το ίδιο άθροισμα. Για παράδειγμα, ο αριθμός 35291 έχει ένα breakpoint διότι μπορεί να διαχωρισθεί μεταξύ των ψηφίων 352 και 91, και 3 + 5 + 2 = 10 και 9 + 1 = 10. Ο αριθμός 1234 δεν έχει breakpoint. Να γραφεί πρόγραμμα το οποίο διαβάζει έναν θετικό ακέραιο και εμφανίζει όλα τα σημεία διακοπής που έχει ή κατάλληλο μήνυμα εάν δεν έχει κανένα. (*breakpoint.psc*)

```
35291
Checking:  3 5291
Checking:  35 291
Checking:  352 91
Breakpoint with sum of digits:  10
Checking:  3529 1
```

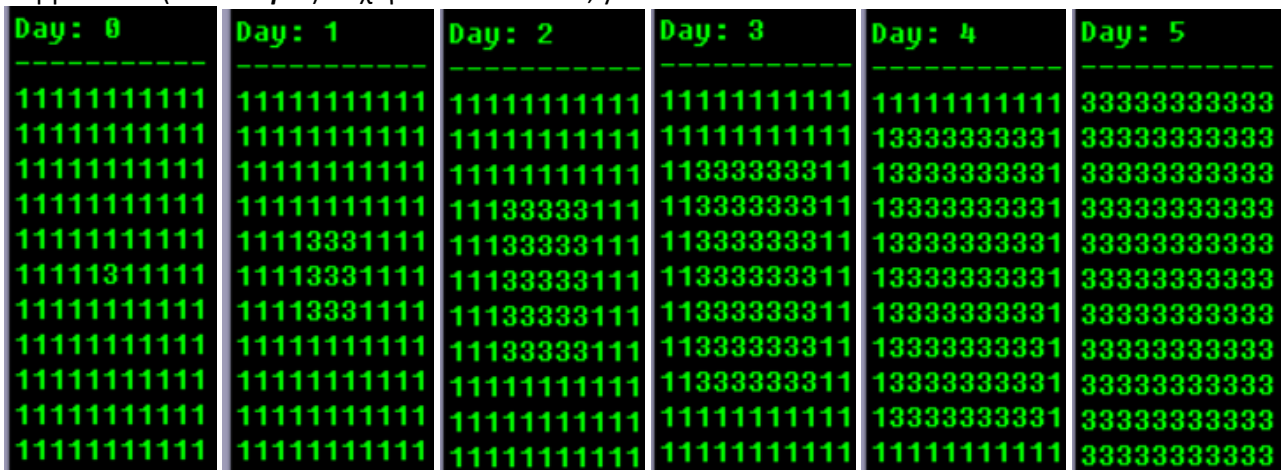
```
1234
Checking:  1 234
Checking:  12 34
Checking:  123 4
No breakpoint found
```

```
1203
Checking:  1 203
Checking:  12 3
Breakpoint with sum of digits:  3
Checking:  120 3
Breakpoint with sum of digits:  3
```

90. Υπάρχουν 6 θετικοί τριψήφιοι ακέραιοι με **άθροισμα ψηφίων ίσο με το γινόμενό τους**. Να γραφεί πρόγραμμα που τους εμφανίζει. (*athroisma_ginomeno_pshfiwn_isa.psc*)
91. Η **συμμετρική διαφορά** δύο συνόλων Α, Β ορίζεται ως τα στοιχεία του Α που δεν ανήκουν στο Β και τα στοιχεία του Β που δεν ανήκουν στο Α. π.χ. Α = {1,2,3,4,5} Β = {4,5,6,7,8} => συμμετρική διαφορά = {1,2,3,6,7,8}. Να γραφεί πρόγραμμα που διαβάζει τους πίνακες Α[5] και Β[5] και εμφανίζει τη συμμετρική διαφορά τους καθώς και το πλήθος των στοιχείων της. Υπόδειξη: φτιάξτε κατάλληλη διαδικασία που δέχεται ως παραμέτρους 2 πίνακες και εμφανίζει τα στοιχεία του 1ου που δεν ανήκουν στον 2ο και επιστρέφει το πλήθος τους. Καλέστε την 2 φορές. (*symmetrikh_diafora.psc*)

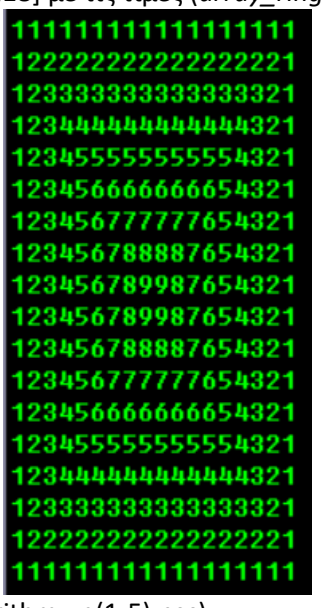


92. Έστω ότι στον πίνακα Α[Ν,Ν] καταγράφονται με 1 τα υγιή άτομα και με 3 τα άτομα που νοσούν από έναν **μεταδοτικό ιό**, σε έναν πληθυσμό ΝxΝ ατόμων. Κάθε ημέρα ο ιός μεταδίδεται από κάθε ασθενές άτομο προς όλα τα γειτονικά με αυτό άτομα. Να γραφεί πρόγραμμα που αρχικοποιεί έναν Α[Ν, Ν] με την τιμή 1 (όλοι αρχικά υγιείς) και διαβάζει ένα στοιχείο (x,y) που θα αποτελέσει το 1ο ασθενές άτομο του πληθυσμού θέτοντας στο Α[x,y] την τιμή 3. Να προσομοιώνει τη διαδικασία μετάδοσης του ιού εμφανίζοντας κάθε ημέρα πόσοι νοσούν, μέχρι να νοσήσει όλος ο πληθυσμός. Εμφανίζει πόσες ημέρες χρειάσθηκαν για να συμβεί αυτό (***covid19.psc***). Π.χ. για Ν=11 και x=6, y=6:



93. Να γραφεί πρόγραμμα το οποίο γεμίζει πίνακα Α[18,18] με τις τιμές (*array_rings.psc*):

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1
1	2	3	3	3	3	3	3	3	3	3	3	3	3	3	3	2	1
1	2	3	4	4	4	4	4	4	4	4	4	4	4	4	3	2	1
1	2	3	4	5	5	5	5	5	5	5	5	5	5	4	3	2	1
1	2	3	4	5	6	6	6	6	6	6	6	6	5	4	3	2	1
1	2	3	4	5	6	7	7	7	7	7	7	6	5	4	3	2	1
1	2	3	4	5	6	7	8	8	8	8	7	6	5	4	3	2	1
1	2	3	4	5	6	7	8	9	9	8	7	6	5	4	3	2	1
1	2	3	4	5	6	7	8	9	9	8	7	6	5	4	3	2	1
1	2	3	4	5	6	7	8	8	8	8	7	6	5	4	3	2	1
1	2	3	4	5	6	7	7	7	7	7	7	6	5	4	3	2	1
1	2	3	4	5	6	6	6	6	6	6	6	6	5	4	3	2	1
1	2	3	4	5	5	5	5	5	5	5	5	5	5	4	3	2	1
1	2	3	4	4	4	4	4	4	4	4	4	4	4	4	3	2	1
1	2	3	3	3	3	3	3	3	3	3	3	3	3	3	3	2	1
1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1



94. Πρόγραμμα που εμφανίζει το παρακάτω (trigwno_arithmwn(1-5).psc):

α)
```
1
12
123
1234
12345
123456
1234567
12345678
123456789
12345678
1234567
123456
12345
1234
123
12
1
```
β)
```
9
98
987
9876
98765
987654
9876543
98765432
987654321
98765432
9876543
987654
98765
9876
987
98
9
```
γ)
```
1
21
321
4321
54321
654321
7654321
87654321
987654321
87654321
7654321
654321
54321
4321
321
21
1
```
δ)
```
1
11
111
1111
11111
111111
1111111
11111111
111111111
11111111
1111111
111111
11111
1111
111
11
1
```
ε)
```
Δώσε ψηφίο (1-9)
8
------------
8
88
888
8888
88888
888888
8888888
88888888
888888888
88888888
8888888
888888
88888
8888
888
88
8
```

95. Ο αριθμός e (**αριθμός Όιλερ**) είναι σημαντική μαθηματική σταθερά, η οποία αποτελεί τη βάση του φυσικού λογαρίθμου (wikipedia). Μπορεί επίσης να υπολογιστεί ως το άθροισμα της άπειρης σειράς

$$e = \sum_{n=0}^{\infty} \frac{1}{n!} = 1 + \frac{1}{1} + \frac{1}{1 \cdot 2} + \frac{1}{1 \cdot 2 \cdot 3} + \cdots$$

Η αριθμητική αξία του e μέχρι τα 50 δεκαδικά ψηφία είναι 2,71828182845904523536028747135266249775724709369995. Να γραφεί πρόγραμμα που υπολογίζει την τιμή του e για τους 20 πρώτους όρους της παραπάνω σειράς (e_arithmos_Oiler.psc).
```
2.718281828459040
```
96. Ένας 13ψήφιος ραβδωτός κώδικας (**barcode**) έχει 12 ψηφία και ένα 13ο το ψηφίο ελέγχου που υπολογίζεται βάση των 12 πρώτων, ως εξής: προσθέτουμε τα 12 ψηφία με τα ψηφία στις μονές θέσεις ως έχουν και τα ψηφία στις ζυγές θέσεις πολλαπλασιασμένα επί 3. Από το αποτέλεσμα κρατάμε το τελευταίο ψηφίο. Αν αυτό είναι μηδέν τότε το ψηφίο ελέγχου είναι το μηδέν. Διαφορετικά, το ψηφίο ελέγχου είναι το αποτέλεσμα που προκύψει εάν αφαιρέσουμε από το 10 το τελευταίο ψηφίο. Να γραφεί πρόγραμμα που διαβάζει τα 12 πρώτα ψηφία ενός ραβδωτού κώδικα και υπολογίζει το 13ο ψηφίο ελέγχου. (barcode.psc)

97. Να γραφεί πρόγραμμα που γεμίζει έναν ακέραιο Α[Ν, Ν] (Ν: περιττός) με τις τιμές 1-N^2 και τον εμφανίζει «**κατά δακτυλίους**» δεξιόστροφα από το κέντρο και βγαίνοντας προς τα έξω ξεκινώντας από το επάνω αριστερά στοιχείο του κάθε δακτυλίου (sarwsh_kata_daktylioys.psc). Π.χ.

```
Gemisma pinaka. Please wait...
Pinakas A( 3 x 3 ):
-----------------------
1  2  3
4  5  6
7  8  9
-----------------------
sarwsh_kata_daktylioys:
daktylios 1: 5
daktylios 2 : 1 2 3 6 9 8 7 4
```

```
Gemisma pinaka. Please wait...
Pinakas A( 5 x 5 ):
-----------------------
1  2  3  4  5
6  7  8  9  10
11 12 13 14 15
16 17 18 19 20
21 22 23 24 25
-----------------------
sarwsh_kata_daktylioys:
daktylios 1: 13
daktylios 2 : 7 8 9 14 19 18 17 12
daktylios 3 : 1 2 3 4 5 10 15 20 25 24 23 22 21 16 11 6
```

```
Gemisma pinaka. Please wait...
Pinakas A( 7 x 7 ):
-----------------------
1  2  3  4  5  6  7
8  9  10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31 32 33 34 35
36 37 38 39 40 41 42
43 44 45 46 47 48 49
-----------------------
sarwsh_kata_daktylioys:
daktylios 1: 25
daktylios 2 : 17 18 19 26 33 32 31 24
daktylios 3 : 9 10 11 12 13 20 27 34 41 40 39 38 37 30 23 16
daktylios 4 : 1 2 3 4 5 6 7 14 21 28 35 42 49 48 47 46 45 44 43 36 29 22 15 8
```

```
Gemisma pinaka. Please wait...
Pinakas A( 9 x 9 ):
-----------------------
1  2  3  4  5  6  7  8  9
10 11 12 13 14 15 16 17 18
19 20 21 22 23 24 25 26 27
28 29 30 31 32 33 34 35 36
37 38 39 40 41 42 43 44 45
46 47 48 49 50 51 52 53 54
55 56 57 58 59 60 61 62 63
64 65 66 67 68 69 70 71 72
73 74 75 76 77 78 79 80 81
-----------------------
sarwsh_kata_daktylioys:
daktylios 1: 41
daktylios 2 : 31 32 33 42 51 50 49 40
daktylios 3 : 21 22 23 24 25 34 43 52 61 60 59 58 57 48 39 30
daktylios 4 : 11 12 13 14 15 16 17 26 35 44 53 62 71 70 69 68 67 66 65 56 47 38 29 20
daktylios 5 : 1 2 3 4 5 6 7 8 9 18 27 36 45 54 63 72 81 80 79 78 77 76 75 74 73 64 55 46 37 28 19 10
```

98. Να γραφεί πρόγραμμα που σχεδιάζει τα παρακάτω με κλήση ξεχωριστών διαδικασιών για το κάθε σχήμα. Χρησιμοποιείστε την εντολή "ΓΡΑΨΕ_ ..." η οποία κάνει εμφάνιση χωρίς να αλλάξει γραμμή και την εντολή "ΓΡΑΨΕ" η οποία απλά αλλάζει γραμμή (draw_stars.psc)

99. Να γραφεί πρόγραμμα που διαβάζει έναν θετικό ακέραιο και θα ελέγχει αν είναι ή δεν είναι της **μορφής qq**. Π.χ. οι 123123, 56215621 είναι της μορφής qq, ενώ οι 123321, 12312 δεν είναι (arithmos_qq.psc)
100. Ένας **φοιτητής** δίνει 4 μαθήματα σε ένα εξάμηνο. Περνάει το εξάμηνο εάν: (a) έχει σε όλα τουλάχιστον τη βάση (50) (b) έχει σε ένα κάτω από τη βάση και μέσο όρο άνω του 60 (c) έχει σε δύο κάτω από τη βάση αλλά πάνω από 40 και μέσο όρο άνω του 70. Να γραφεί πρόγραμμα που διαβάζει τους βαθμούς του φοιτητή (0-100) και ελέγχει εάν περνάει το εξάμηνο (foiththts.psc). Σενάρια ελέγχου: a. 50, 50, 50, 50 (ok) b. 49, 50, 50, 100 (ok) c. 49, 49, 90, 100 (ok) d. 49, 49, 80, 100 / 40, 49, 100, 100 (failed)
101. "Δυνατός" είναι ένας θετικός ακέραιος με άθροισμα των παραγοντικών των ψηφίων του ίσο με τον αριθμό. π.χ. ο 145 = 1! + 4! + 5!. Να γραφεί πρόγραμμα που διαβάζει έναν θετικό ακέραιο και ελέγχει αν είναι "δυνατός" (dynatos_arithmos.psc). Υπόδειξη: να γράψετε συνάρτηση που δέχεται έναν θετικό ακέραιο x και υπολογίζει το παραγοντικό του (x! = 1*2*...*x)
102. Να γραφεί πρόγραμμα το οποίο διαβάζει τον πραγματικό Α[9] και βρίσκει τον 1ο **υποπίνακα συνεχόμενων στοιχείων του με το μέγιστο άθροισμα**. π.χ. Α={-8, 3, 8, -5, 4, 5, -4, 3, -5} ➔ μέγιστο άθροισμα = 15 για τον υποπίνακα Α[2]-Α[6] (largest_sum_of_contiguous_subarray.psc) / (maxSubArray.psc)
103. Να γραφεί πρόγραμμα που διαβάζει έναν θετικό ακέραιο και εμφανίζει το πλήθος των **καταληκτικών μηδενικών** του (trailing_zeros.psc). π.χ. 120300(2), 1203(0)
104. Στα μαθηματικά, ένας **αυτομορφικός** αριθμός είναι ένας θετικός ακέραιος του οποίου το τετράγωνο "τελειώνει" με τα ίδια ψηφία με τον ίδιο τον αριθμό. π.χ. ο 376: 376^2 = 141376. Να γραφεί πρόγραμμα που διαβάζει έναν θετικό ακέραιο και ελέγχει αν είναι αυτομορφικός (aytomorfikos_arithmos.psc).
105. Στα μαθηματικά, ένας **ετερομεκτικός** (heteromecic) αριθμός είναι ένας θετικός ακέραιος που ισούται με το γινόμενο 2 συνεχόμενων ακεραίων. Είναι δηλ. της μορφής n*(n+1). π.χ. ο 342 = 18*19. Να γραφεί πρόγραμμα που διαβάζει έναν θετικό ακέραιο και ελέγχει αν είναι ετερομεκτικός (heteromecic_arithmos.psc).
106. Να γραφεί πρόγραμμα που διαβάζει 3 θετικούς ακέραιους και εμφανίζει τον μεγαλύτερο **χωρίς χρήση των δομών Αν και Επίλεξε** (max3WithoutIf.psc)
107. Να γραφεί πρόγραμμα που διαβάζει τον ακέραιο Α[10] τοποθετώντας κάθε στοιχείο με **παρεμβολή** στη σωστή θέση ώστε ο πίνακας να είναι ταξινομημένος κατά αύξουσα σειρά. π.χ. Αν ο Α περιέχει τα στοιχεία 5,7,8 και εισαχθεί η τιμή 6, αυτή να παρεμβληθεί μεταξύ του 5 και του 7. (parembolh_stoixeiwn.psc)
108. Να γραφεί πρόγραμμα το οποίο υπολογίζει τις **συχνότητες των αθροισμάτων** (2-12) **και των γινομένων** (1-36) της ρίψης **2 ζαριών**. Ποιο το συχνότερο άθροισμα και γινόμενο; (syxnothtes_2_zariwn.psc)

```
syxnotero athroisma: 7  syxnothta: 6
syxnotero ginomeno: 6  syxnothta: 4
syxnotero ginomeno: 12  syxnothta: 4
```

109. Σε μία φάση κυπέλλου ποδοσφαίρου συμμετέχουν **16 ομάδες με σκοπό να προκριθούν στους 8**. Να γραφεί πρόγραμμα που διαβάζει στον ΟΜ1[16] τα ονόματα των ομάδων. Έστω ότι τα ζευγάρια των αναμετρήσεων είναι ανά δύο (1-2, 3-4, ..., 15-16). Για κάθε μία από τις 8 αναμετρήσεις διαβάζει τον αριθμό του νικητή με έλεγχο εγκυρότητας. Γεμίζει τον ΟΜ2[8] με τις ομάδες που προκρίθηκαν και τον εμφανίζει. (epomenh_fash_kypelloy.psc)
110. Έστω **ανελκυστήρας** προγραμματισμένος να εξυπηρετεί τα **αιτήματα** με την εξής σειρά: αν κατευθύνεται προς τα πάνω και βρίσκεται στον όροφο x, εξυπηρετεί πρώτα τα αιτήματα από τους ορόφους που είναι

μεγαλύτεροι του x και κατά αύξουσα σειρά και έπειτα τα αιτήματα από τους ορόφους που είναι μικρότεροι του x και κατά φθίνουσα σειρά. π.χ. αν ο ανελκυστήρας βρίσκεται στον 9ο όροφο και έχουν καταγραφεί τα αιτήματα από τους ορόφους: 11, 3, 20, 19, 1, 4, 7, 8, 12, 5 τότε αυτά θα εξυπηρετηθούν με την εξής σειρά: 11,12,19,20,8,7,5,4,3,1. Να γραφεί πρόγραμμα το οποίο διαβάζει τον όροφο x που βρίσκεται ένας ανελκυστήρας σε 20όροφο κτίριο καθώς και τα 10 αιτήματα που έστω ότι έχουν καταγραφεί (1-20 και διαφορετικά του x). Να εμφανίζει τη σειρά εξυπηρέτησης των αιτημάτων (anelkysthras.psc)

111. Να γραφεί πρόγραμμα που διαβάζει έναν ακέραιο και εμφανίζει το **πρώτο και το τελευταίο ψηφίο** του (prwto_pshfio.psc)
112. Να γραφεί πρόγραμμα που διαβάζει έναν ακέραιο A[N] (έστω με διαφορετικές τιμές) και μία τιμή αναζήτησης x. Να **αναζητά** το x στον A **και από τις δύο κατευθύνσεις ταυτόχρονα** (από την αρχή και από το τέλος του) και να εμφανίζει το αποτέλεσμα της αναζήτησης καθώς και τον αριθμό των επαναλήψεων που χρειάστηκε (search_both_directions.psc)
113. Να γραφεί πρόγραμμα που διαβάζει δύο ώρες της ίδιας ημέρας (ώρα:λεπτά:δευτερόλεπτα) και εμφανίζει τη χρονική τους απόσταση σε ώρες, λεπτά και δευτερόλεπτα (xronikh_apostash.psc). π.χ.

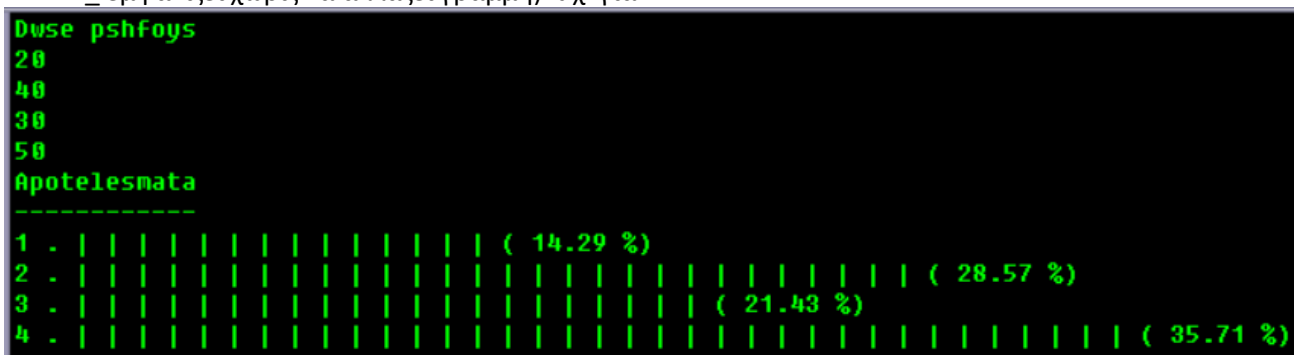
```
Dwse 1h wra
8
15
5
Dwse 1h wra
10
0
20
Apexoyn kata 1 wres 45 lepta kai 15 sec
```

114. Να γραφεί πρόγραμμα που προσομοιώνει τη **μετάδοση ενός ιού** ως εξής: αρχικοποιεί έναν πίνακα χαρακτήρων A[N,M] με την τιμή 'Y' (υγιές). Διαβάζει τις συντεταγμένες (x,y) ενός μολυσμένου στοιχείου και του εκχωρεί την τιμή 'M' (μολυσμένο). Εάν καθημερινά μολύνεται κάθε κελί που γειτονεύει με κάποιο ήδη μολυσμένο, να ενημερώνει τον πίνακα μέχρι να εξαπλωθεί ο ιός σε όλα τα στοιχεία του. Κάθε ημέρα να εμφανίζει το πλήθος των μολυσμένων στοιχείων. Τελικά να εμφανίζει το πλήθος των ημερών που απαιτήθηκαν για να συμβεί το γεγονός αυτό. (virus_spread.psc). π.χ. για A[3,5] και αρχικό μολυσμένο στοιχείο το A[1,1] ➔ 4 ημέρες, για A[3,5] και αρχικό μολυσμένο στοιχείο το A[2,3] ➔ 2 ημέρες
115. Πρόγραμμα το οποίο διαβάζει έναν θετικό ακέραιο και x και ελέγχει αν **αποτελεί δύναμη του 2** και αν ναι, ποια είναι αυτή (dynamh_toy_2.psc). Π.χ. x = 256 ➔ ναι 2^8
116. Πρόγραμμα που εμφανίζει όλους τους 3ψήφιους θετικούς ακέραιους **που ισούνται με το γινόμενο του αθροίσματος επί του γινομένου των ψηφίων τους**
(tripshfioi_isoi_me_athroisma_epi_ginomeno_pshfiwn.psc). π.χ. 135 = (1+3+5)x(1x3x5)
117. Να γραφεί πρόγραμμα που διαβάζει έναν θετικό 3ψήφιο ακέραιο και εμφανίζει την **κρυπτογραφημένη τιμή του αυξάνοντας το κάθε ψηφίο του κατά 1 κυκλικά** (0 => 1, 1 => 2, ... 8 => 9, 9 => 0) (kryptografhsh_tripshfioy.psc) π.χ. 327 => 438, 106 => 217, 394 => 405
118. Να γραφεί πρόγραμμα το οποίο διαβάζει έναν ακέραιο A[N] και υπολογίζει το **% ποσοστό ανακατέματός** του το οποίο προκύπτει από τον αριθμό των αντιμεταθέσεων που απαιτήθηκαν ώστε να ταξινομηθεί κατά αύξουσα σειρά δια του μέγιστου πλήθους των απαιτούμενων αντιμεταθέσεων για την ταξινόμησή του, επί 100. Δηλ. pososto_anakatematos = antimetatheseis_taksinomhshs / maxAntimetatheseis * 100, maxAntimetatheseis = N*(N-1)/2 (pososto_anakatematos.psc)
119. Να γραφεί πρόγραμμα το οποίο από τους N πρώτους ακέραιους εμφανίζει αυτούς με το **μεγαλύτερο πλήθος διαιρετών** (max_diairetes.psc). π.χ. για N=100 ➔

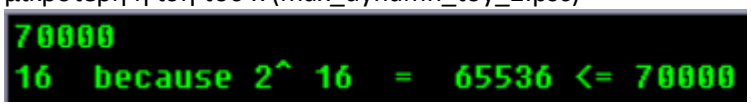
```
O 60 exei 12 diairetes
O 72 exei 12 diairetes
O 84 exei 12 diairetes
O 90 exei 12 diairetes
O 96 exei 12 diairetes
```

120. Να γραφεί πρόγραμμα που διαβάζει τις ψήφους N υποψηφίων σε μία εκλογική διαδικασία και εμφανίζει τα % ποσοστά των ψήφων τους σε μορφή **ραβδογράμματος** (rabdogramma.psc). (Σημείωση: η εντολή

ΓΡΑΨΕ_ εμφανίζει χωρίς να αλλάζει γραμμή) π.χ. για N=4 ➔



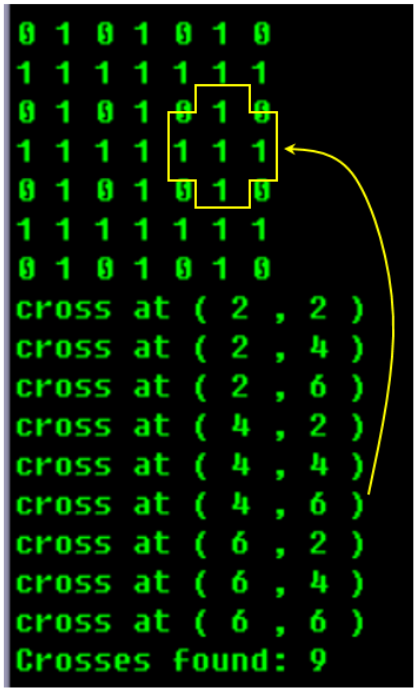
121. Πρόγραμμα που διαβάζει έναν ακέραιο x (>=1) και υπολογίζει τη μεγαλύτερη δύναμη του 2 (n) που είναι μικρότερη ή ίση του x (max_dynamh_toy_2.psc)



122. Να γραφεί πρόγραμμα το οποίο διαβάζει έναν ακέραιο πίνακα Α[100, 100] τιμές 0/1. Να εμφανίζει τους 3x3 υποπίνακες του Α που έχουν **σχήμα σταυρού** από άσσους:

0	1	0
1	1	1
0	1	0

καθώς και πόσοι είναι (find_crosses.psc) π.χ.



123. Πρόγραμμα που διαβάζει έναν ακέραιο και ελέγχει αν όλα τα ψηφία του είναι ίσα. π.χ. 999, 11111, 33 (ola_ta_pshfia_isa.psc)
124. Να γραφεί συνάρτηση που υπολογίζει το άθροισμα των ψηφίων ενός ακέραιου. Να γραφεί πρόγραμμα που διαβάζει δύο ακέραιους και **τους συγκρίνει με βάση το άθροισμα των ψηφίων τους** (compareBySumOfDigits.psc)
125. Η διαδικασία της **σταθεράς του Kaprekar** (σταματά όταν προκύψει ο αριθμός **6174**):

1. Παίρνουμε έναν ακέραιο Κ
2. Αναδιατάσσουμε τα ψηφία του Κ ώστε να προκύψει ο μέγιστος (και ο ελάχιστος) δυνατός αριθμός. Π.χ. αν ο Κ είναι ο 2309 τότε ο μέγιστος είναι ο 9320 και ο ελάχιστος ο 239 (θεωρείστε leading zero άρα 239 = 0239)
3. Αφαιρούμε τον ελάχιστο από τον μέγιστο και με τη διαφορά τους εκτελούμε το βήμα 1

Παράδειγμα 1: Έστω ο αριθμός 2600. Ο μέγιστος ακέραιος που μπορεί να προκύψει από αναδιάταξη των ψηφίων του 2600 είναι ο 6200 και ο ελάχιστος είναι ο 26 (0026). Αφαιρώντας τους δύο ακέραιους προκύπτει η σταθερά Kaprekar μετά από 1 επανάληψη (6200 – 0026 = 6174).

Παράδειγμα 2: Έστω ο αριθμός 1542. Ο μέγιστος ακέραιος που μπορεί να προκύψει από αναδιάταξη των ψηφίων του 1542 είναι ο 5421 και ο ελάχιστος είναι ο 1245. Αφαιρώντας τους δύο ακέραιους προκύπτει ο

αριθμός 4176. Ο μέγιστος ακέραιος που μπορεί να προκύψει από αναδιάταξη των ψηφίων του 4176 είναι ο 7641 και ο ελάχιστος είναι ο 1467. Αφαιρώντας τους δύο ακέραιους προκύπτει η σταθερά Kaprekar μετά από 2 επαναλήψεις (7641 – 1467 = 6174). (Kaprekar_Constant.psc)

Είσοδος	Έξοδος
0	Wrong Input
3333	Wrong Input
2600	1
1542	2
3215	7
9899	5

```
Dwse K
9899
999
8991
8082
8532
6174
steps =  5
```

126. Να γίνει πρόγραμμα το οποίο θα διαβάζει έναν μη αρνητικό ακέραιο αριθμό N από το χρήστη (1 ≤ N ≤ 100) και στη συνέχεια να διαβάζει N ακέραιους αριθμούς που θα τους αποθηκεύει σε ένα πίνακα Α. Το πρόγραμμα μέσω κατάλληλου επαναληπτικού μενού θα διαβάζει χαρακτήρες από το χρήστη μέχρι να δεχτεί το 'q'. Ανάλογα με το χαρακτήρα που διάβασε θα κάνει με κατάλληλα υποπρογράμματα, τα ακόλουθα:

• Αν διαβάσει το 'a' (Append) τότε θα διαβάζει έναν ακέραιο k και θα τον προσθέτει στο τέλος του πίνακα Α.

• Αν διαβάσει το 'd' (Delete) τότε θα διαβάζει έναν ακέραιο p και θα αφαιρεί από τον πίνακα τον ακέραιο που είχε στη θέση p. Προφανώς κατά την αφαίρεση στοιχείου ο πίνακας θα θεωρούμε ότι πλέον έχει ένα στοιχείο λιγότερο. Πχ. Αν από τον πίνακα {5,10,15,20} αφαιρεθεί το στοιχείο στη θέση 1, ο πίνακας θα είναι ο {10,15,20}.

• Αν διαβάσει το 'i' (Insert) τότε θα διαβάζει δύο ακέραιους k και p και θα εισαγάγει τον αριθμό k στη θέση p του πίνακα. Πχ. Αν στον πίνακα {5,10,15,20} εισαγάγω τον αριθμό 7 στη θέση 2, τότε ο νέος πίνακας θα είναι ο {5,7,10,15,20}.

• Αν διαβάσει το 'c' (Clear) τότε θα "αδειάζει" τον πίνακα

• Όταν το πρόγραμμα δεχτεί το q θα τερματίζει.

Μετά την εκτέλεση κάθε εντολής θα εμφανίζει τα στοιχεία του πίνακα. (Append_Insert_Delete_Array.psc)

127. **Διανομέας καφέ** έχει να παραδώσει καφέδες σε 4 σημεία της πόλης. Χρειάζεται μία εφαρμογή η οποία να του συστήνει τη συντομότερη διαδρομή που ξεκινάει από το μαγαζί που εργάζεται, περνάει από τα 4 σημεία διανομής και επιστρέφει στη βάση του. Να γραφεί πρόγραμμα το οποίο: α) καταχωρεί στον D[5, 5] τις αποστάσεις μεταξύ των 5 σημείων (γραμμές/στήλες 1-4: τα 4 σημεία διανομής και γραμμή/στήλη 5: το σημείο εκκίνησης και τερματισμού). Ενδεικτικά: D[i, j] = i*j για i<>j και μηδέν(0) στη διαγώνιο β) υπολογίζει το μήκος όλων των διαφορετικών διαδρομών (4! = 1x2x3x4 = 24) γ) εμφανίζει το μικρότερο μήκος διαδρομής και τη(τις) διαδρομές που το έχουν.

Travelling cafe delivery problem

Distances

D	1	2	3	4	5
1	0	2	3	4	5
2	2	0	6	8	10
3	3	6	0	12	15
4	4	8	12	0	20
5	5	10	15	20	0

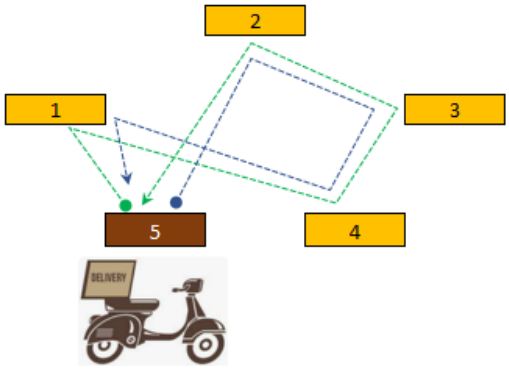
D[i, j] = i x j, i ≠ j

Paths (starting from and ending to 5)

1. 1 2 3 4 S= 45
2. 1 2 4 3 S= 42
3. 1 3 2 4 S= 42
4. 1 3 4 2 S= 38
5. 1 4 2 3 S= 38
6. 1 4 3 2 S= 37 *(5+4+12+6+10)*
7. 2 1 3 4 S= 47
8. 2 1 4 3 S= 43
9. 2 3 1 4 S= 43
10. 2 3 4 1 S= 37 *(10+6+12+4+5)*
11. 2 4 1 3 S= 40
12. 2 4 3 1 S= 38
13. 3 1 2 4 S= 48
14. 3 1 4 2 S= 40
15. 3 2 1 4 S= 47
16. 3 2 4 1 S= 38
17. 3 4 1 2 S= 43
18. 3 4 2 1 S= 42
19. 4 1 2 3 S= 47
20. 4 1 3 2 S= 43
21. 4 2 1 3 S= 48
22. 4 2 3 1 S= 42
23. 4 3 1 2 S= 47

(24=4!) 24. 4 3 2 1 S= 45

min= 37



traveling_salesman4.psc

128. **Ευγενείς αριθμοί** (Polite): ορίζονται οι θετικοί ακέραιοι αριθμοί που μπορούν να γραφτούν ως το άθροισμα δύο ή περισσότερων διαδοχικών θετικών ακεραίων. π.χ.6=1+2+3, 14=2+3+4+5, το 15 γράφεται με 2 τρόπους 15=1+2+3+4+5=7+8, ενώ το 16 δεν είναι ευγενής αριθμός. Να γράψετε πρόγραμμα το οποίο: να διαβάζει έναν θετικό ακέραιο αριθμό α. Να ελέγχει αν ο α αποτελεί Ευγενή αριθμό. Αν αυτό δεν συμβαίνει να εμφανίζει σχετικό μήνυμα διαφορετικά να εμφανίζει τους διαδοχικούς θετικούς ακέραιους των οποίων το άθροισμα ισούται με τον αριθμό α. Για αριθμούς που γράφονται ως άθροισμα διαδοχικών ακεραίων με περισσότερους από έναν τρόπους, να εμφανίζει τη σειρά με το μεγαλύτερο πλήθος διαδοχικών όρων. (π.χ. για το 15 τη σειρά 1,2,3,4,5 και όχι 7,8). Υπόδειξη με 2 παραδείγματα: α)Έστω ότι θέλουμε να ελέγξουμε τον αριθμό 7. Ελέγχουμε διαδοχικά τα αθροίσματα: 1+2, 1+2+3, 1+2+3+4, εδώ σταματάμε γιατί έχουμε άθροισμα=10>7. Ξεκινάμε πάλι ελέγχοντας 2+3, 2+3+4, εδώ σταματάμε γιατί έχουμε άθροισμα=9>7. Ξεκινάμε πάλι 3+4 =7, τέλος, άρα το 7 είναι ευγενής αριθμός. β)Έστω ότι θέλουμε να ελέγξουμε τον αριθμό 16. Ελέγχουμε διαδοχικά τα αθροίσματα: 1+2, 1+2+3, 1+2+3+4, 1+2+3+4+5, 1+2+3+4+5+6, εδώ σταματάμε γιατί έχουμε άθροισμα=21>16. Ξεκινάμε πάλι ελέγχοντας 2+3, 2+3+4, 2+3+4+5, 2+3+4+5+6, εδώ σταματάμε γιατί έχουμε άθροισμα=20>16. Ξεκινάμε πάλι 3+4, 3+4+5, 3+4+5+6,=18>16 άρα σταματάμε. Ξεκινάμε πάλι ελέγχοντας 4+5, 4+5+6, 4+5+6+7, =22>16 άρα σταματάμε. Ξεκινάμε πάλι ελέγχοντας 5+6, 5+6+7, =18>16. Ξανά 6+7, 6+7+8, =21>16 άρα σταματάμε. Ξανά 7+8, 7+8+9, =24>16 stop. Ξανά 8+9, =17>16. Εδώ τελειώνουμε διότι το επόμενο άθροισμα που πρέπει να ελέγξουμε είναι το 9+10 =19>16 και όλες οι επόμενες διαδοχές 10+11, 11+10, ... θα μας δίνουν όλο και μεγαλύτερα αθροίσματα. (Σκεφτείτε γενικότερα πότε πρέπει να τελειώνουμε τον έλεγχο για να κάνουμε πιο αποδοτικό τον αλγόριθμό μας). Άρα το 16 δεν είναι ευγενής αριθμός. (polite_numbers.psc)

```
Give a number
15
Polite number. Sequence found:
1 + 2 + 3 + 4 + 5
Polite number. Sequence found:
4 + 5 + 6
Polite number. Sequence found:
7 + 8
-------------------------
max sequence:
1 + 2 + 3 + 4 + 5
```

129. **Αριθμοί Cunningham**: ορίζονται οι θετικοί ακέραιοι αριθμοί C οι οποίοι μπορούν να γραφτούν στη μορφή C=b2-1, με b θετικό ακέραιο και b>1. Ονομάστηκαν έτσι προς τιμή του Allan Joseph Champneys Cunningham (1842–1928) Βρετανο-Ινδού μαθηματικού. π.χ. 15=4^2-1, 24=5^2-1, ενώ δεν υπάρχει θετικός ακέραιος b για να γραφτεί το 20 σ' αυτή τη μορφή. Να γράψετε πρόγραμμα το οποίο: να διαβάζει έναν θετικό ακέραιο αριθμό α. Να ελέγχει αν ο α αποτελεί αριθμό Cunningham και αν αυτό συμβαίνει να τον εμφανίζει στη μορφή α=b^2-1 (π.χ. για α=8 να εμφανίζει 8=3^2-1), διαφορετικά να εμφανίζει σχετικό μήνυμα. (cunningham_numbers.psc)

130. **Ταξινόμηση με Κάδους - BucketSort**: Έστω ότι ο πίνακας A n στοιχείων περιέχει στοιχεία που ανήκουν στο διάστημα [1..m].Ο αλγόριθμος BucketSort βασίζεται πάνω στα ακόλουθα βήματα: 1. Δημιουργούμε ένα πίνακα buckets μήκους m και θέτουμε buckets[i]=0, για όλα τα i (Αυτά τα είναι τα buckets - κάδοι). 2. Διαβάζουμε τον πίνακα A ξεκινώντας από το πρώτο στοιχείο. Αν διαβάσουμε το στοιχείο α, τότε αυξάνουμε την τιμή του buckets[α] κατά ένα. Επαναλαμβάνουμε το βήμα μέχρι το τελευταίο στοιχείο. 3. Τέλος, διαβάζουμε γραμμικά τον πίνακα buckets, ο οποίος περιέχει αναπαράσταση του ταξινομημένου πίνακα, και θέτουμε τα στοιχεία του πίνακα A με την ταξινομημένη ακολουθία. Να γραφεί πρόγραμμα που διαβάζει έναν ακέραιο A[20] με τιμές στο [1, 6] και τον ταξινομεί με τον παραπάνω αλγόριθμο (BucketSort.psc)

PS[1]	4
PS[2]	4
PS[3]	3
PS[4]	3
PS[5]	3
PS[6]	3

π.χ.
```
A: 1 2 3 4 5 6 1 2 3 4 5 6 1 2 3 4 5 6 1 2
A sorted:  1 1 1 1 2 2 2 2 3 3 3 4 4 4 5 5 5 6 6 6
```

131. Να γραφεί τμήμα προγράμματος ισοδύναμο με το παρακάτω χωρίς τη χρήση των δομών Αν/Επίλεξε (An_Oso.psc):

```
ΔΙΑΒΑΣΕ x, y
ΑΝ x < y ΤΟΤΕ
  ΓΡΑΨΕ -1
ΑΛΛΙΩΣ_ΑΝ  x > y  ΤΟΤΕ
  ΓΡΑΨΕ 1
ΑΛΛΙΩΣ
  ΓΡΑΨΕ 0
ΤΕΛΟΣ_ΑΝ
```

132. Να γραφεί πρόγραμμα το οποίο να εμφανίζει τους ακέραιους 1-100 **κατά φθίνουσα σειρά πλήθους διαιρετών**. Όσοι έχουν τον ίδιο αριθμό διαιρετών να εμφανίζονται κατά αύξουσα σειρά. (sortByDividers.psc)

```
60 : 12
72 : 12
84 : 12
90 : 12
96 : 12
48 : 10
80 : 10
36 : 9
100 : 9
24 : 8
...
43 : 2
47 : 2
53 : 2
59 : 2
61 : 2
67 : 2
71 : 2
73 : 2
79 : 2
83 : 2
```

133. Να γραφεί πρόγραμμα το οποίο διαβάζει έναν θετικό ακέραιο και ελέγχει εάν έχει τα **ψηφία του σε αύξουσα σειρά**. π.χ. 12489. (pshfiaSeAyksoysaSeira.psc)

134. Να γραφεί πρόγραμμα το οποίο διαβάζει έναν θετικό ακέραιο Ν και στη συνέχεια διαβάζει **Ν θετικούς ακέραιους και τους συνενώνει σε έναν**. (concatNumbers.psc) π.χ. για Ν=3 και τιμές εισόδου: 32, 4, 128 => αποτέλεσμα = 324128. Υπόδειξη: να φτιάξετε συνάρτηση Digits(x): Ακέραια η οποία επιστρέφει το πλήθος των ψηφίων ενός ακέραιου x.

135. Να γραφεί πρόγραμμα το οποίο διαβάζει τον ακέραιο Α[10] με θετικές τιμές και τον **ταξινομεί** κατά φθίνουσα σειρά **με βάση το άθροισμα των ψηφίων τους**. Οι τιμές με ίσο άθροισμα ψηφίων να ταξινομούνται

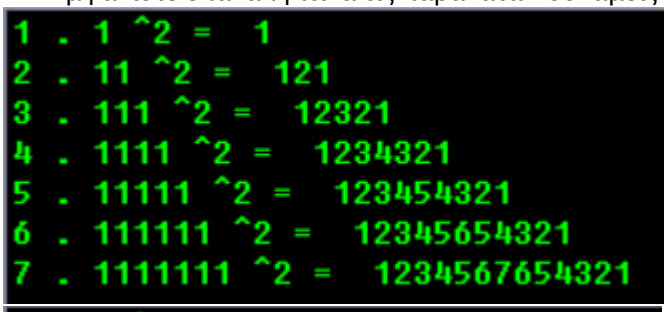
κατά φθίνουσα σειρά (sort_by_sum_of_digits.psc). Π.χ. για τιμές εισόδου: 12, 23, 34, 45, 56, 67, 78, 89, 100, 121:

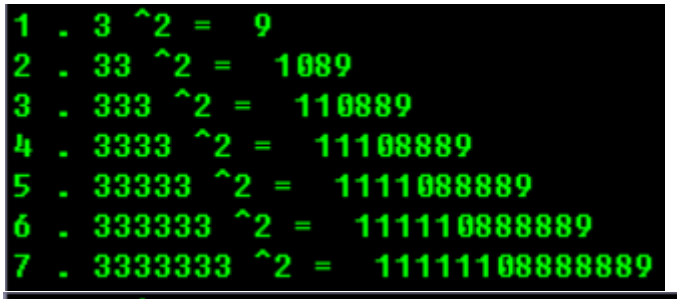
Arithmos 89 athroisma pshfiwn 17
Arithmos 78 athroisma pshfiwn 15
Arithmos 67 athroisma pshfiwn 13
Arithmos 56 athroisma pshfiwn 11
Arithmos 45 athroisma pshfiwn 9
Arithmos 34 athroisma pshfiwn 7
Arithmos 23 athroisma pshfiwn 5
Arithmos 121 athroisma pshfiwn 4
Arithmos 12 athroisma pshfiwn 3
Arithmos 100 athroisma pshfiwn 1

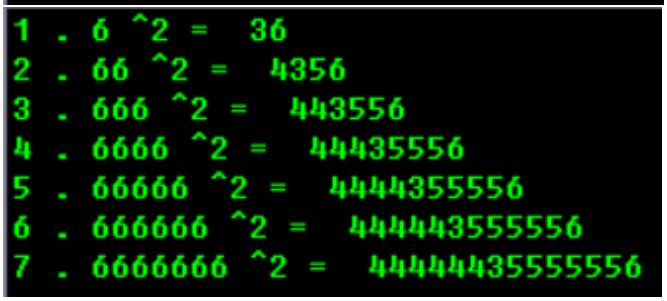
136. Να γραφεί πρόγραμμα το οποίο διαβάζει έναν ακέραιο (1-39) και τον εμφανίζει στη **ρωμαϊκή γραφή** (ArabicToRoman.psc):

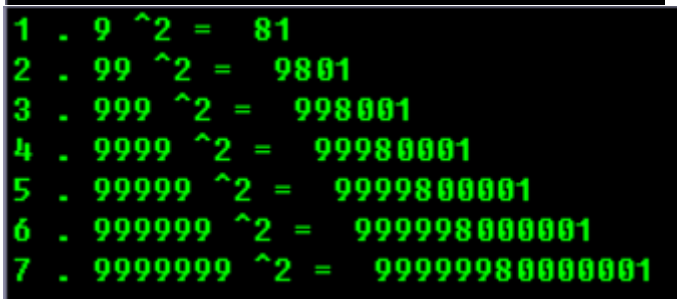
Arabic	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39
Roman	I	II	III	IV	V	VI	VII	VIII	IX	X	XI	XII	XIII	XIV	XV	XVI	XVII	XVIII	XIX	XX	XXI	XXII	XXIII	XXIV	XXV	XXVI	XXVII	XXVIII	XXIX	XXX	XXXI	XXXII	XXXIII	XXXIV	XXXV	XXXVI	XXXVI	XXXVI	XXXIX

137. Εμφανίστε επαναληπτικά τις παρακάτων δυνάμεις του 2 (DynameisToyDyo.psc):



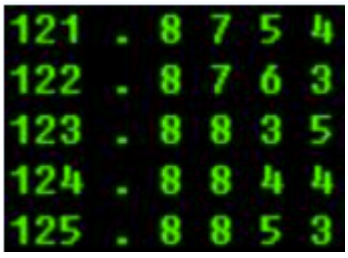




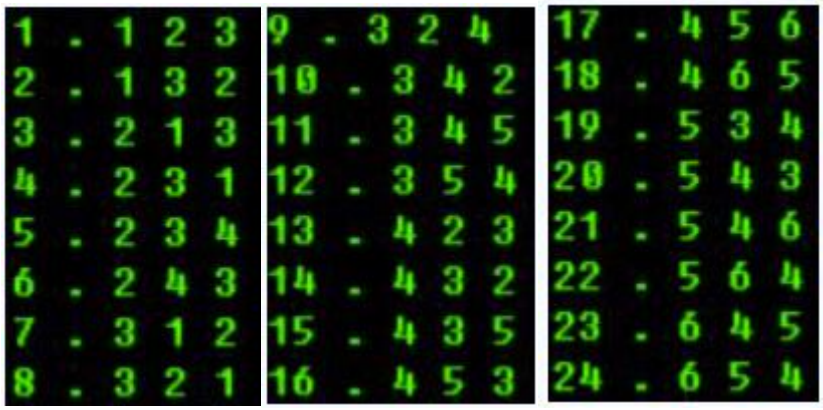


138. Να γραφεί πρόγραμμα που διαβάζει **2 ημερομηνίες** (d/m/y) και **εμφανίζει την απόστασή τους σε ημέρες**. Να υλοποιηθούν οι παρακάτω συναρτήσεις: DaysOfYear(y): Ακέραια η οποία υπολογίζει τις ημέρες του έτους y (365/366 για δίσεκτο) και DaysOfMonth(m, y) : Ακέραια η οποία υπολογίζει τις ημέρες του μήνα m και του έτους y (30 για τους μήνες 4/6/9/11, 28 για το μήνα 2 ή 29 για το μήνα 2 και δίσεκτο έτος, 31 για τους υπόλοιπους μήνες). π.χ. Είσοδος: 8,9,1970 και 18,6,2025 Έξοδος: 20007. Είσοδος: 1,6,2025 και 18,6,2025 Έξοδος: 17. Είσοδος: 1,6,2025 και 5,5,2025 Έξοδος: 66. (DatesDistance.psc)

139. Θέλουμε να μοιράσουμε 24 **καραμέλες** σε 4 παιδιά ώστε κάθε παιδί να πάρει τουλάχιστον 3 και το πολύ 8 καραμέλες. Να βρείτε τους 125 διαφορετικούς τρόπους να γίνει αυτό. (karameles.psc)

...


140. Να γραφεί πρόγραμμα που εμφανίζει τους 24 διαφορετικούς συνδυασμούς ρίψης τριών(3) ζαριών ώστε οι τιμές τους να είναι διαδοχικές (zaria3_diadoxika.psc)



141. Να βρείτε τους 2ψηφιους **πρώτους** αριθμούς που όταν **αντιστρέψετε τα ψηφία** τους, να προκύπτουν πάλι πρώτοι. Π.χ. ο 13, διότι ο 31 είναι πρώτος. 11 11, 13 31 , 17 71 , 31 13 , 37 73 , 71 17 , 73 37 , 79 97 , 97 79 (symmetrikosPrwtos.psc)

142. Να γραφεί η **Διαδικασία** deyterobathmia(a, b, c, r1, r2, r, eidos) η οποία δέχεται τους συντελεστές a, b, c της **δευτεροβάθμιας εξίσωσης** a*x^2+b*x+c=0 και την επιλύει επιστρέφοντας το eidos: 1. για δύο ρίζες (r1, r2), 2. για μία διπλή ρίζα (r), 3. μία ρίζα (r), 4. για αδύνατη, 5. για αόριστη. Να γραφεί κύριο πρόγραμμα που επιλύει όλες τις δευτεροβάθμιες εξισώσεις a*x^2+b*x+c=0 για κάθε a, b, c στο ακέραιο διάστημα [-5, 5]. (deyterobathmia_for_diadikasia.psc)

143. **Δίδυμοι πρώτοι αριθμοί** είναι το 11 και το 13, καθώς είναι και οι δύο πρώτοι και η διαφορά τους είναι 2. Να γραφεί πρόγραμμα το οποίο εμφανίζει όλους τους θετικούς διψήφιους δίδυμους πρώτους. (didymoiPrwtoi.psc)

twin prime pairs | less than 1000

Values:

(3, 5) | (5, 7) | (11, 13) | (17, 19) | (29, 31) | (41, 43) | (59, 61) | (71, 73) | (101, 103) | (107, 109) | (137, 139) | (149, 151) | (179, 181) | (191, 193) | (197, 199) | (227, 229) | (239, 241) | (269, 271) | (281, 283) | (311, 313) | (347, 349) | (419, 421) | (431, 433) | (461, 463) | (521, 523) | (569, 571) | (599, 601) | (617, 619) | (641, 643) | (659, 661) | (809, 811) | (821, 823) | (827, 829) | (857, 859) | (881, 883) (35 prime pairs)

144. Να βρείτε τους 2ψηφιους πρώτους αριθμούς που όταν αντιστρέψετε τα ψηφία τους, να προκύπτουν πάλι πρώτοι. Π.χ. ο 13, διότι ο 31 είναι πρώτος. 11 11, 13 31 , 17 71 , 31 13 , 37 73 , 71 17 , 73 37 , 79 97 , 97 79 (symmetrikosPrwtos.psc)

145. Μια γραμμική γεννήτρια συμβατών αριθμών (linear congruential generator - LCG) είναι ένας αλγόριθμος που αποδίδει μια ακολουθία **ψευδοτυχαίων** αριθμών που υπολογίζονται με μια εξίσωση. Η μέθοδος αντιπροσωπεύει έναν από τους παλαιότερους και πιο γνωστούς αλγόριθμους γεννήτριας ψευδοτυχαίων αριθμών. Η γεννήτρια ορίζεται από τη σχέση αναδρομής: X_next = (a * X_previous + b) mod m όπου X_next ο επόμενος τυχαίος αριθμός, X_previous ο προηγούμενος τυχαίος αριθμός, m το "modulus" (m > 0), a ο "multiplier" (0 < a < m), b το "increment" (0 < b < m) και αρχικός αριθμός τροφοδότησης (seed) της γεννήτριας ο xo (0 < xo < m). Να γραφεί πρόγραμμα που διαβάζει τα: m, a, b, xo με τους παραπάνω περιορισμούς και εμφανίζει 100 τυχαίους αριθμούς (randomGenerator.psc). π.χ. για m=101, a=3, b=7, xo=11:

1 . 40
2 . 26
3 . 85
4 . 60
5 . 86
.....
95 . 88

96 . 69
97 . 12
98 . 43
99 . 35
100 . 11

146. Να γραφεί πρόγραμμα που διαβάζει 10 πραγματικούς και εμφανίζει τον μικρότερο και τον μεγαλύτερο **χωρίς τη χρήση συγκριτικού τελεστή** (*MinMaxWithoutIf.psc*)

147. Να γραφεί πρόγραμμα που υπολογίζει τους 16 όρους της παράστασης:

$$\sqrt{12+\sqrt{12+\sqrt{12+\sqrt{12+\cdots}}}}$$

(*riza12.psc*)

148. Δίνεται ένας θετικός ακέραιος διψήφιος αριθμός ΧΥ με δύο ψηφία Χ και Υ. Αντιστρέψτε τα ψηφία και δημιουργήστε τον αριθμό ΥΧ. Στη συνέχεια βρείτε τη θετική διαφορά μεταξύ ΧΥ και ΥΧ. Επαναλάβατε τη διαδικασία με το θετικό διψήφιο αριθμό που προέκυψε ως διαφορά και μέχρις ότου ως νέα διαφορά προκύψει μονοψήφιος αριθμός. Η λίστα των αριθμών που δημιουργούνται καλείται "**λίστα προς το 9**" γιατί καταλήγει πάντα στον αριθμό 9. Γράψτε πρόγραμμα που για κάθε διψήφιο θετικό ακέραιο αριθμό με διαφορετικά ψηφία να δημιουργεί τη λίστα του προς το 9. Να εμφανίζει τους αριθμούς της λίστας καθώς και το μήκος της. π.χ. λίστα του αριθμού 19 προς το 9: 19, 72, 45, 9. Μήκος λίστας: 4 (lista_pros_to_9_ALL.psc)