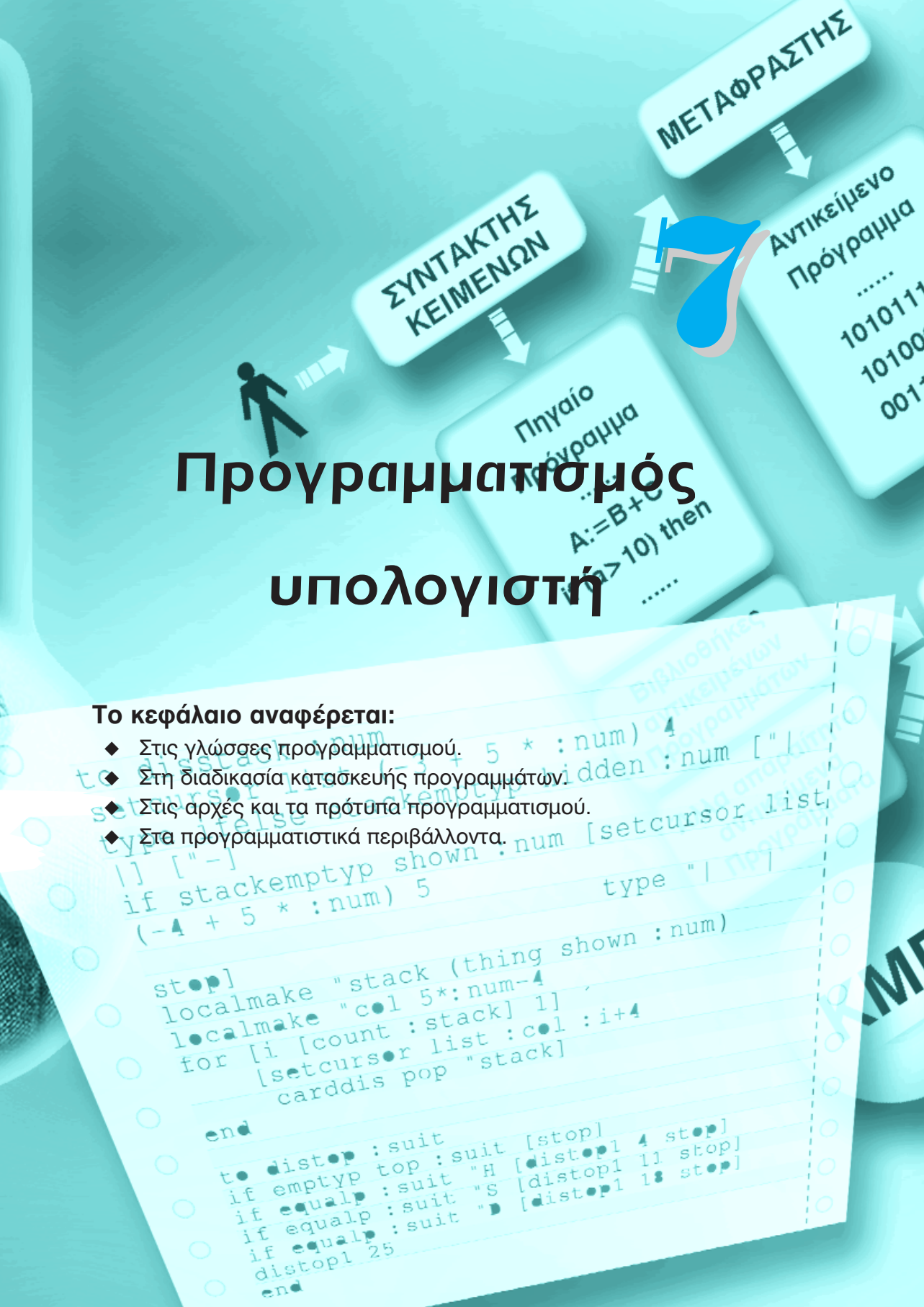


7 Προγραμματισμός υπολογιστή

Το κεφάλαιο αναφέρεται:

- Στις γλώσσες προγραμματισμού.
- Στη διαδικασία κατασκευής προγραμμάτων.
- Στις αρχές και τα πρότυπα προγραμματισμού.
- Στα προγραμματιστικά περιβάλλοντα.

7.1 Χρονοπρογραμματισμός

Συνολικός προτεινόμενος χρόνος 10 ώρες (4 θεωρία, 6 ασκήσεις).

Δίωρο 1ο: **Κατανομή ύλης**

Η προγραμματιζόμενη μηχανή
Οι χρήστες
Πρόγραμμα - Γλώσσες προγραμματισμού
- Το Πρόγραμμα
- Γλώσσα μηχανής
- Συμβολικές γλώσσες
- Γλώσσες υψηλού επιπέδου
- Ένα συγκριτικό παράδειγμα
- Ιστορία γλωσσών υψηλού επιπέδου
- Εξάρτηση των γλωσσών από το σκοπό
- Μεταφραστές

Αρχές κατασκευής λογισμικού

Δίωρο 2ο: **Κατανομή ύλης**

Πρότυπα προγραμματισμού
- Διαδικαστικός προγραμματισμός
- Αντικειμενοστρεφής προγραμματισμός
- Λογικός προγραμματισμός
- Συναρτησιακός προγραμματισμός

Προγραμματίζοντας
- Καθορισμός προβλήματος
- Ανάλυση του προβλήματος
- Σχεδιασμός
- Υλοποίηση
- Έλεγχος
- Συντήρηση
- Τεκμηρίωση
- Τρόποι παράστασης αλγόριθμων

Προγραμματιστικά περιβάλλοντα
Ένα παράδειγμα ανάπτυξης προγράμματος
- Προσδιορισμός των απαιτήσεων του προβλήματος
- Ανάλυση του προβλήματος
- Σχεδιασμός αλγόριθμου για την επίλυση του προβλήματος
- Υλοποίηση του αλγόριθμου

Έλεγχος του προγράμματος

Δίωρο 3ο: **Κατανομή ύλης**

Ασκήσεις

Δίωρο 4ο: **Κατανομή ύλης**

Ασκήσεις

Δίωρο 5ο: **Κατανομή ύλης**

Ασκήσεις

7.2 Γενικές παρατηρήσεις

Σε αυτό το κεφάλαιο οι μαθητές έρχονται σε επαφή με τα θέματα του προγραμματισμού του υπολογιστή. Πρέπει να έχουμε κατά νου ότι στόχος του μαθήματος δεν είναι να κάνουμε τους μαθητές μας προγραμματιστές, οπότε πρέπει να αποφύγουμε τον πειρασμό.
Ο στόχος είναι να μπορέσουν να καταλάβουν ότι οι υπολογιστές λειτουργούν με προγράμματα τα οποία κάποιοι κατασκευάζουν και που στη βάση τους αποτελούνται από απλές εντολές που καθοδηγούν τον υπολογιστή, με σκοπό τελικά να απομυθοποιηθεί στα μάτια τους ο υπολογιστής.
Στο τέλος του κεφαλαίου θα πρέπει:

- Να έχουν ξεκαθαρίσει τις έννοιες του προγραμματισμού, των γλωσσών και των προγραμματιστικών περιβαλλόντων.
- Να έχουν αντιληφθεί το πώς ένα πρόβλημα μπορεί να λυθεί με τον κατάλληλο προγραμματισμό ενός υπολογιστή.
- Να αντιλαμβάνονται την ανάγκη χρήσης συγκεκριμένης μεθοδολογίας στην ανάπτυξη προγραμμάτων.

Οι εργαστηριακές ασκήσεις θα τους δείξουν πώς μπορούν να κατασκευάσουν απλά προγράμματα για να διερευνήσουν, εκτός από τον προγραμματισμό του υπολογιστή, θέματα και προβλήματα από άλλα γνωστικά αντικείμενα.

Γενικά οι ασκήσεις δεν απαιτούν εξεζητημένο προγραμματισμό και ο αλγόριθμος κάθε προγράμματος μπορεί εύκολα να εξηγηθεί. Οι αλγόριθμοι που προτείνονται στη συνέχεια έχουν γίνει με γνώμονα την ευκολία κατανόησής τους από τους μαθητές και έχουν αποφευχθεί προγραμματιστικές βελτιστοποιήσεις. Οι αλγόριθμοι που δίνονται είναι γραμμένοι σε ψευδογλώσσα που είναι πολύ κοντά στο συντακτικό της Pascal.
Οι ασκήσεις 35 και 36 δεν υπάρχουν στο τετράδιο εργασίας μαθητή και μένουν στην κρίση σας να διδαχθούν στην περίπτωση που υπάρχει από την τάξη η άνεση για αναφορά σε αρχεία.

Στους αλγορίθμους των λύσεων των ασκήσεων χρησιμοποιούνται τα:

- m mod n: Το ακέραιο υπόλοιπο της διαίρεσης m : n
- m div n: Το ακέραιο πηλίκο της διαίρεσης m : n.

Προτείνεται, στην περίπτωση των ασκήσεων που στη λύση τους απαιτείται χρήση πινάκων, να συζητείται το αντίστοιχο θέμα (Πίνακες) της παραγράφου των ειδικών θεμάτων του τετραδίου ασκήσεων μαθητή. Το ίδιο να γίνει και

στην περίπτωση των υποπρογραμμάτων. Σημειωτέον ότι στα ειδικά θέματα, στο τμήμα της Pascal, γίνεται αναφορά και σε έννοιες προγραμματισμού συμπληρωματικά από τη θεωρία.

Οι συνθετικές εργασίες που συνδέονται με το κεφάλαιο αυτό είναι οι:

- Βάση Παράλληλων Ιστορικών Γεγονότων «ΘΟΥΚΥΔΙΔΗΣ»
- Δανειστική Βιβλιοθήκη «ΑΛΕΞΑΝΔΡΕΙΑ»

7.3 Σχέδιο μαθημάτων

7.3.1 1ο Δίωρο

Διδακτέα ύλη

Η προγραμματιζόμενη μηχανή
Οι χρήστες
Πρόγραμμα - Γλώσσες προγραμματισμού
- Το Πρόγραμμα
- Γλώσσα μηχανής
- Συμβολικές γλώσσες
- Γλώσσες υψηλού επιπέδου
- Ένα συγκριτικό παράδειγμα
- Ιστορία γλωσσών υψηλού επιπέδου
- Εξάρτηση των γλωσσών από το σκοπό
- Μεταφραστές

Αρχές κατασκευής λογισμικού

Ερωτήσεις: 1-11

Ασκήσεις:

Υποδείξεις - Παρατηρήσεις Σε αυτό το δίωρο θα παρουσιαστεί η ιστορική εξέλιξη των γλωσσών προγραμματισμού. Να γίνει κατανοητή στους μαθητές η ανάγκη για γλώσσες υψηλού επιπέδου καθώς και οι έννοιες του compiler & interpreter

7.3.2 2ο, 3ο Δίωρο

Διδακτέα ύλη

Πρότυπα προγραμματισμού

Διαδικαστικός προγραμματισμός
Αντικειμενοστρεφής προγραμματισμός
Λογικός προγραμματισμός
Συναρτησιακός προγραμματισμός
Προγραμματίζοντας
Καθορισμός προβλήματος
Ανάλυση του προβλήματος
Σχεδιασμός
Υλοποίηση
Έλεγχος
Συντήρηση
Τεκμηρίωση
Τρόποι παράστασης αλγόριθμων
Προγραμματιστικά περιβάλλοντα
Ένα παράδειγμα ανάπτυξης προγράμματος
Προσδιορισμός των απαιτήσεων του προβλήματος
Ανάλυση του προβλήματος
Σχεδιασμός αλγόριθμου για την επίλυση του προβλήματος
Υλοποίηση του αλγόριθμου
Έλεγχος του προγράμματος

Ερωτήσεις: 12-17

Ασκήσεις: 1-12

Υποδείξεις - Παρατηρήσεις Βαρύτητα να δοθεί στη μεθοδολογία επίλυσης προβλημάτων.

7.2.3 4ο, 5ο Δίωρο

Διδακτέα ύλη

Ερωτήσεις:

Ασκήσεις: 12-34 και (35, 36)

7.3 Λύσεις - υποδείξεις ερωτήσεων

1. Τι είναι αυτό που κάνει τον ίδιο υπολογιστή ικανό να εκτελεί εργασίες διαφορετικές μεταξύ τους (π.χ. να κάνει στατιστική ανάλυση, να σχεδιάζει σπίτια, να επεξεργάζεται βίντεο);

☐ Το υλικό

☑ Το λογισμικό

☐ Άλλο.

2. «Για να χρησιμοποιηθεί ο υπολογιστής ως εργαλείο σε ένα τομέα είναι αρκετό να προμηθευτούμε τα κατάλληλα προγράμματα. Το υλικό δεν έχει καμία σημασία». Συμφωνείτε;

☐ Ναι.

☑ Όχι.

3. Τι ονομάζεται πρόγραμμα υπολογιστή;
[§ Το πρόγραμμα]

4. Τι εννοούμε λέγοντας ότι ο υπολογιστής προγραμματίζεται;
[§ Η προγραμματιζόμενη μηχανή]

5. Τι είναι η γλώσσα μηχανής;
[§ Γλώσσα μηχανής]

6. Να αναφέρετε τρεις κατηγορίες εντολών γλώσσας μηχανής.
α) . . . . . .
β) . . . . . .
γ) . . . . . .
[§ Γλώσσα μηχανής]

7. Οι προγραμματισμός σε συμβολική γλώσσα είναι πιο **δύσκολος** σε σχέση με τον προγραμματισμό σε γλώσσες υψηλού επιπέδου.
(εύκολος / δύσκολος)

8. Να αναφέρετε τρεις γλώσσες προγραμματισμού ειδικού σκοπού και τρεις γενικού:
α) . . . . . .
β) . . . . . .
γ) . . . . . .
δ) . . . . . .
ε) . . . . . .

στ) . . . . . .
[§ Ιστορία γλωσσών υψηλού επιπέδου]

9. Η LISP και η SMALLTALK είναι:

☑ Γλώσσες προγραμματισμού υψηλού επιπέδου

☐ Γλώσσες μηχανής του μικροεπεξεργαστή Z80

☐ Άλλο.

10. Να περιγράψετε τα βήματα για την εκτέλεση ενός προγράμματος με τη χρήση μεταφραστή.
[§ Μεταφραστές]

11. Ποιες από τις παρακάτω διαπιστώσεις είναι σωστές;

☑ Ο τμηματικός προγραμματισμός αποτελεί στρατηγική κατασκευής προγραμμάτων.

☑ Ο δομημένος προγραμματισμός βασίζεται στην αρχή της λειτουργικής αποσύνθεσης.

☐ Η υπορουτίνα είναι ένα μια δομή επιλογής.

12. Ποιος ο σκοπός των εντολών συνθήκης στο διαδικαστικό προγραμματισμό;
[§ Διαδικαστικός προγραμματισμός]

13. Τι είναι οι μέθοδοι και τι οι ιδιότητες στον αντικειμενοστρεφή προγραμματισμό;
[§ Αντικειμενοστρεφής προγραμματισμός]

14. Τα δεδομένα στον αντικειμενοστρεφή προγραμματισμό αποθηκεύονται στις μεθόδους. Συμφωνείτε;

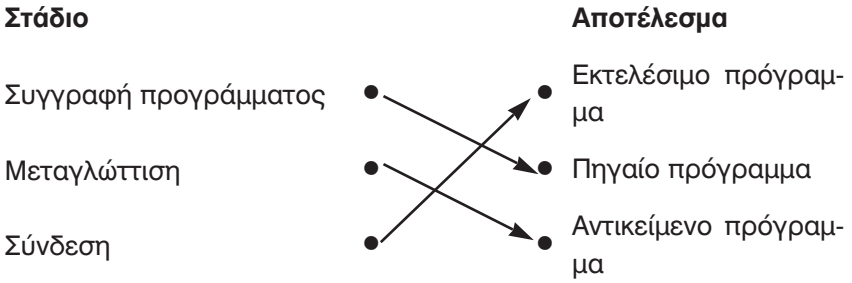
☐ Ναι.

☑ Όχι.

15. Τι είναι ένα προγραμματιστικό περιβάλλον και από τι αποτελείται;
[§ Προγραμματιστικά περιβάλλοντα]

16. Ποια βήματα ακολουθούνται κατά την ανάπτυξη λογισμικού;
[§ Προγραμματίζοντας]

17. Να συνδέσετε τα στοιχεία της πρώτης στήλης με αυτά της δεύτερης.



7.4 Λύσεις και σχόλια ασκήσεων και δραστηριοτήτων

1. Διερευνήστε το προγραμματιστικό περιβάλλον που είναι διαθέσιμο στο εργαστήριό σας.

- Πώς ονομάζεται;
- Ποια ή ποιες γλώσσες υποστηρίζει;
- Να γράψετε ένα πρόγραμμα που να τυπώνει «Καλημέρα».
- Να καταγράψετε τα βήματα και τις εντολές που απαιτήθηκαν στο προγραμματιστικό περιβάλλον του εργαστηρίου σας για τον πιο πάνω σκοπό.

ΣΧΟΛΙΟ

Σκοπός αυτής της εργασίας είναι οι μαθητές να κάνουν μια πρώτη γνωριμία με το προγραμματιστικό περιβάλλον του εργαστηρίου τους και τα βασικά βήματα που απαιτούνται για να υλοποιήσουν ένα πρόγραμμα (γραφή, μεταγλώττιση, εκτέλεση).

2. Να γραφεί ένα πρόγραμμα που να διαβάζει τους βαθμούς θερμοκρασίας σε Fahrenheit και να τυπώνει τους αντίστοιχους σε Celsius.

ΣΧΟΛΙΟ

```
Algorithm Celsius
Variables
   far, cel: real
Begin
   read(far)
   cel <- 5/9*(far - 32)
   write(cel)
end algorithm
```

3. Να γραφεί ένα πρόγραμμα που να διαβάζει τρεις αριθμούς και να απαντά αν μπορούν να αποτελέσουν γωνίες τριγώνου (μονάδα μέτρησης των γωνιών η μοίρα).

ΣΧΟΛΙΟ

```
Algorithm  Gonies_trigonou
Variables
    a, b, c, sum: real
Begin
    read(a, b, c)
    sum <- a + b + c
    if sum = 180 then
          write('Είναι γωνίες τριγώνου')
    else
          write('Δεν είναι γωνίες τριγώνου')
    end if
end algorithm
```

4. Να γραφεί πρόγραμμα που να υπολογίζει το εμβαδό ενός τριγώνου από τις πλευρές του.

ΣΧΟΛΙΟ

```
Algorithm  Emvado
Variables
    a, b, c, t, emv: real
Begin
    read(a, b, c)
    t <- (a + b + c) / 2
    emv <- √t(t-a)(t-b)(t-c)
    write('Εμβαδό=', emv)
end algorithm
```

5. Να γραφεί πρόγραμμα που να διαβάζει δύο ακέραιους αριθμούς και να τυπώνει το ποιος είναι ο μεγαλύτερος και ποιος ο μικρότερος.

ΣΧΟΛΙΟ

```
Algorithm  min_max
Variables
    a, b, min, max : integer
Begin
    read(a, b)
    if a > b then
```

```
            max <- a
            min <- b
      else
            max <- b
            min <- a
      end if
      write('Max=', max, 'Min=', min)
end algorithm
```

6. Να γραφεί πρόγραμμα που να διαβάζει έναν ακέραιο αριθμό και να βρίσκει αν είναι άρτιος ή περιττός.

ΣΧΟΛΙΟ

```
Algorithm  Artios_i_peritos
Variables
    a: integer
Begin
    read(a)
    if (a mod 2 = 0) then
          write('Είναι άρτιος')
    else
          write('Είναι περιττός')
    end if
end algorithm
```

7. Να γραφεί πρόγραμμα που να διαβάζει έναν ακέραιο αριθμό Ν, και να υπολογίζει το άθροισμα των περιττών μεταξύ 1 και Ν.

ΣΧΟΛΙΟ

```
Algorithm  Athrisma_peritwn
Variables
    n, i, sum: integer
Begin
    read(n)
    sum <- 0
    for i <- 1 to n step 2
          sum <- sum + i
    end for
    write('Sum=', sum)
end algorithm
```

8. Να γραφεί πρόγραμμα που να διαβάζει 20 αριθμούς και να τυπώσει πόσοι από αυτούς είναι θετικοί και πόσοι αρνητικοί. Θεωρήστε το 0 θετικό.

ΣΧΟΛΙΟ

```
Algorithm  Thetikoi_arnitikoi
Variables
   i, pcount, ncount: integer
   a: real
Begin
   pcount <- 0
   ncount <- 0
   for i <- 1 to 20 step 1
         read(a)
         if a >= 0 then
               pcount <- pcount + 1
         else
               ncount <- ncount + 1
         end if
   end for
   write('Θετικοί: ', pcount, 'Αρνητικοί:', ncount)
end algorithm
```

9. Να γραφεί πρόγραμμα που να διαβάζει έναν ακέραιο αριθμό Ν και να υπολογίζει το άθροισμα 1+2+3+...+Ν.

ΣΧΟΛΙΟ

```
Algorithm Athrisma
Variables
   n, i, sum: integer
Begin
   read(n)
   sum <- 0
   for i <- 1 to n step 1
         sum <- sum + i
   end for
   write('1+2+..+',n,'=', sum)
end algorithm
```

10. Να γραφεί πρόγραμμα που να διαβάζει έναν ακέραιο αριθμό Ν και να υπολογίζει το γινόμενο 1*2*3...*N.

ΣΧΟΛΙΟ

```
Algorithm  Paragodiko
Variables
   n, i, par: integer
Begin
   read(n)
   par <- 1
   for i <- 1 to n step 1
         par <- par * i
   end for
   write('1*2*..*',n,'=', par)
end algorithm
```

11. Να γραφεί πρόγραμμα το οποίο να εναλλάσσει τα περιεχόμενα δύο μεταβλητών.

ΣΧΟΛΙΟ

```
Algorithm  swap
Variables
   a, b, temp: real
Begin
....
   temp <- a
   a <- b
   b <- temp
....
end algorithm
```

12. Να γραφεί ένα πρόγραμμα που να διαβάζει το ποσό ενός κεφαλαίου και το ετήσιο επιτόκιο και να υπολογίζει το τελικό κεφάλαιο μετά από 10 χρόνια με ετήσιο ανατοκισμό.

ΣΧΟΛΙΟ

```
Algorithm  Anatokismos
Variables
   kef, tel_kef, epit:real
   n: integer
Begin
```

```
    n<-10
    read(kef, epit)
    tel_kef <- kef * (1 + epit / 100)^n
    write('Τελικό κεφάλαιο=', tel_kef)
end algorithm
```

13. Σε μια εταιρεία πωλήσεων, στο τέλος κάθε μήνα ο κάθε πωλητής εκτός από το μισθό του παίρνει και ένα bonus που είναι ανάλογο των πωλήσεων που έκανε. Να γράψετε ένα πρόγραμμα που να δέχεται στην είσοδο το όνομα και το ύψος των πωλήσεων ενός πωλητή και να τυπώνει στην έξοδο το όνομα και το αντίστοιχο bonus. Ο τρόπος υπολογισμού του bonus είναι:

Ύψος πωλήσεων	Bonus
0 - 2.000.000	0
2.000.001 - 5.000.000	2%
5.000.001 - 10.000.000	4%
10.000.001 -	5%

Το ποσοστό του bonus υπολογίζεται χωριστά για κάθε τμήμα της κλίμακας.

ΣΧΟΛΙΑ

```
Algorithm  bonus
Variables
    bonus, poliseis : real
    onoma           : string
Begin
    bonus <- 0
    read(onoma, poliseis)
    if poliseis > 10000000 then
          temp <- poliseis - 10000000
          bonus <- bonus + temp * .05
          poliseis <- poliseis - 10000000
    end if
    if poliseis > 5000000 then
            temp <- poliseis - 5000000
          bonus <- bonus + temp * .04
          poliseis <- poliseis - temp
```

```
    end if
    if poliseis > 2000000 then
            temp <- poliseis - 2000000
          bonus <- bonus + temp * .02
    end if
write('Ο πωλητής: ',name,'δικαιούται bonus: ',
       bonus)
end algorithm
```

14. Να γράψετε ένα πρόγραμμα που να διαβάζει έναν αριθμό δευτερολέπτων και να τυπώνει τον αντίστοιχο χρόνο σε ώρες, λεπτά και δευτερόλεπτα στη μορφή ΩΩ:ΛΛ:ΔΔ. Π.χ. 3:15:43.

ΣΧΟΛΙΑ

```
Algorithm  time
Variables
    hour,min, sec, num_sec: integer
Begin
    read(num_sec)
    hour <- num_sec div 3600
       num_sec <- num_sec - hour * 3600
    min <- num_sec div 60
    num_sec <- num_sec - min * 60
    sec <- num_sec
    write(hour, ':', min, ':', sec)
end algorithm
```

15. Να γράψετε ένα πρόγραμμα που να βρίσκει σε έναν κύκλο την περίμετρο και το εμβαδόν του, αν είναι γνωστή η ακτίνα του.

ΣΧΟΛΙΑ

```
Algorithm  circle
Variables
    aktina, emvado, perim: real
Begin
    pi <- 3.1415
    read(aktina)
    emvado <- pi * aktina2
    perim <- 2 * pi * aktina
    write(`Εμβαδό: ',emvado, 'Περίμετρος: ', perim)
end algorithm
```

16. Να γράψετε ένα πρόγραμμα που να βρίσκει το ΜΚΔ δύο ακεραίων θετικών αριθμών.

ΣΧΟΛΙΑ

Βασιζόμενοι στον αλγόριθμο του Ευκλείδη μπορούμε να ακολουθήσουμε τα παρακάτω βήματα:

1. *Αν οι αριθμοί είναι ίσοι, τότε ο ΜΚΔ είναι ένας από αυτούς και ο αλγόριθμος τερματίζεται.*

2. Βρίσκουμε τον μεγαλύτερο και τον αντικαθιστούμε με τη διαφορά του από τον μικρότερο.

3. Μεταφερόμαστε στο βήμα 1.

```
Algorithm mkd
Variables
    a, b: integer
Begin
    read(a,b)
while a<>b
          if a>b then
                a <- a - b
          else
                b <- b - a
          end if
end while
write('ΜΚΔ: ', a)
end algorithm
```

17. Να γράψετε ένα πρόγραμμα που να βρίσκει το ΕΚΠ δύο ακεραίων θετικών αριθμών.

ΣΧΟΛΙΑ

Παίρνουμε διαδοχικά τα πολλαπλάσια του μεγαλύτερου αριθμού, έως ότου βρούμε κάποιο που το διαιρεί ακριβώς ο μικρότερος.

```
Algorithm  EKP
Variables
   a, b, max, min, ekp: integer
Begin
   read(a,b)
   if a>b then
         max <- a
         min <- b
   else
         max <- b
         min <- a
   end if
   ekp <- max
   while ekp mod min <> 0
         ekp <- ekp + max
   end while
   write('ΕΚΠ: ', ekp)
end algorithm
```

18. Να γράψετε ένα πρόγραμμα που να βρίσκει το ακέραιο πηλίκο και το υπόλοιπο δύο ακεραίων αριθμών, μόνο με τη χρήση πρόσθεσης και αφαίρεσης.

ΣΧΟΛΙΑ

```
Algorithm  piliko_ipolipo_A
Variables
   a, b, piliko, ipolipo, temp :integer
Begin
   read(a, b)
   piliko <- 0
   temp <- b
   while a >= temp
         temp <- temp + b
         piliko <- piliko + 1
```

```
    end while
    ipolipo <- b - (temp - a)
    write('Πηλίκο: ',piliko, ' Υπόλοιπο: ', ipolipo)
end algorithm
```

19. Μπορείτε να λύσετε το προηγούμενο πρόβλημα μόνο με τη χρήση πρόσθεσης;

ΣΧΟΛΙΑ

```
Algorithm  piliko_ipolipo_B
Variables
    a, b, piliko, ipolipo, temp :integer
Begin
    read(a, b)
    piliko <- 0
    temp <- 0
    while a >= temp + b
          temp <- temp + b
          piliko <- piliko + 1
    end while
    ipolipo <- 0
    while a > temp + ipolipo
          ipolipo <- ipolipo + 1
    end while
    write('Πηλίκο: ',piliko, ' Υπόλοιπο: ', ipolipo)
end algorithm
```

20. Σε ένα πείραμα Χημείας μετράμε τις τιμές ενός μεγέθους (π.χ. θερμοκρασίας). Έχουμε επαναλάβει το πείραμα είκοσι φορές και έχουμε καταγράψει τα αποτελέσματα της μέτρησης. Θέλουμε να κατασκευάσουμε ένα πρόγραμμα που να διαβάζει τις μετρήσεις και να τυπώνει το μέσο όρο και την τυπική απόκλιση των μετρήσεων.

ΣΧΟΛΙΑ

Προτείνεται να εξηγηθούν στους μαθητές οι στατιστικές έννοιες του μέσου όρου και της τυπικής απόκλισης.
Επίσης να συζητηθεί η έννοια του πίνακα.

```
Algorithm  Chemistry
Variables
```

```
    m[20] , sum, sum_dx, mesos, apokl: real
    i: integer

Begin
{Διάβασμα μετρήσεων}
    for i <- 1 to 20 step 1
          read(m[i])
    end for
{Υπολογισμός μέσου όρου}
    sum <- 0
    for i <- 1 to 20 step 1
          sum <- sum + m[i]
    end for
    mesos <- sum / 20
{Υπολογισμός τυπικής απόκλισης}
    sum_dx <- 0
    for i <- 1 to 20 step 1
          sum_dx <- sum_dx + (m[i]-mesos)^2
    end for

    apokl <- √(sum_dx/20)
{Εκτύπωση}
    write('Μέσος όρος:', mesos)
    write('Τυπική απόκλιση: ', apokl)
end algorithm
```

21. Να γραφεί ένα πρόγραμμα που να διαβάζει ένα ακέραιο από το 1 έως το 7 και να τυπώνει σε ποια ημέρα της εβδομάδας αυτός αντιστοιχεί. Το 1 αντιστοιχεί στη Δευτέρα.

ΣΧΟΛΙΑ

Η άσκηση αυτή μπορεί να επεκταθεί, έτσι ώστε να γίνεται επαναληπτικά εισαγωγή του αριθμού ημέρας μέχρι να δοθεί μια τιμή περάτωσης της διαδικασίας.

```
Algorithm  days_of_week
Variables
    days[7]: string
    day_num: integer
Begin
```

```
    days[1] <- 'Δευτέρα'
    days[2] <- 'Τρίτη'
    days[3] <- 'Τετάρτη'
    days[4] <- 'Πέμπτη'
    days[5] <- 'Παρασκευή'
    days[6] <- 'Σάββατο'
    days[7] <- 'Κυριακή'

    read(day_num)
    write(days[day_num])
end algorithm
```

22. Κατασκευάστε πρόγραμμα που να εξετάζει αν ένας αριθμός είναι πρώτος.

ΣΧΟΛΙΑ

Μια πιο ορθή προγραμματιστικά λύση θα έπρεπε να τερματίζει την επανάληψη την πρώτη φορά που ο αριθμός διαιρείται ακριβώς με το i. Αυτό θα μπορούσε να υλοποιηθεί με τη χρήση της while.

```
Algorithm  Protos_A
Variables
    m, n, i: integer
    protos: boolean
Begin
    read(m)

    n <- ακέραιο μέρος του √m
    protos <- true
    for i <- 2 to n step 1
          if (m mod i) = 0 then
                protos <- false
          end if
    end for
    if protos then
          write('Είναι πρώτος')
    else
          write('Είναι σύνθετος')
    end if
end algorithm
```

23. Να μετατρέψετε το προηγούμενο πρόγραμμα σε κατάλληλο υποπρόγραμμα και χρησιμοποιήστε το για να τυπώσετε τους πρώτους αριθμούς από 2 έως Ν. Το Ν δίνεται από το χρήστη.

ΣΧΟΛΙΑ

Το μετατρέπουμε σε υποπρόγραμμα τύπου συνάρτησης με παράμετρο τον υπό εξέταση αριθμό, η οποία επιστέφει true αν ο αριθμός είναι πρώτος, αλλιώς false. Προτείνεται με αυτή την ευκαιρία να τους μιλήσετε για τα υποπρογράμματα (από τη θεωρία και τα ειδικά θέματα του Τετράδιου Εργασίας Μαθητή) και να τονιστεί η αξία τους τόσο στην επαναχρησιμοποίηση του κώδικα, όσο και στην τμηματική ανάπτυξη ενός προγράμματος.

Ο αλγόριθμος για το υποπρόγραμμα είναι ο ακόλουθος:

```
function einai_protos(m : integer): boolean
Variables
   n, i: integer
   protos: boolean
begin

   n <- ακέραιο μέρος του √m
   protos <- true
   for i <- 2 to n step 1
         if (m mod i) = 0 then
               protos <- false
         end if
   end for
   return protos
end function
```

Ο αλγόριθμος του προγράμματος γίνεται:

```
Algorithm  Protos_B
Variables
   i, N: integer
Begin
   read(N)
   for i <- 2 to N step 1
         if einai_protos(i) then
               write(i)
```

```
    end if
end for
end algorithm
```

24. Η ημερομηνία του Πάσχα κάποιου έτους μπορεί να υπολογιστεί χρησιμοποιώντας τον τύπο του Gauss. Ο τύπος του Gauss είναι:
Η ημερομηνία του Πάσχα είναι 3 Απριλίου + Π, όπου
$\Pi = D + F$, $D = [19A + 16]_{30}$, $F = [2B + 4C + 6D]_7$, $A = [E]_{19}$, $B = [E]_4$, $C = [E]_7$
Ε: το έτος και $[\alpha]_\beta$ είναι το ακέραιο υπόλοιπο της διαίρεσης του α με το β.
Χρησιμοποιώντας τον τύπο του Gauss, κατασκευάστε ένα πρόγραμμα που να τυπώνει την ημερομηνία του Πάσχα για τα έτη από 2000 έως 2100.

ΣΧΟΛΙΑ

```
Algorithm  pasha
Variables
    a, b, c, d, e, f, p: integer
Begin
    for e <- 2000 to 2100 step 1
          a <- e mod 19
          b <- e mod 4
          c <- e mod 7
          d <- (19 * a + 16) mod 30
          f <- (2 * b + 4 * c + 6 * d) mod 7
          p <- d + f + 3
          if p <= 30 then
                write(p, ' Απριλίου')
                else
                write(p-30, ' Μαϊου')
          end if
    end for
end algorithm
```

25. Να κατασκευάσετε πρόγραμμα που να δέχεται στην είσοδο τις μέσες θερμοκρασίες ενός αγνώστου αριθμού ημερών και να τυπώνει τη μέγιστη και την ελάχιστη καθώς και όλες τις ημερομηνίες που αυτές μετρήθηκαν. Το τέλος των δεδομένων δηλώνεται με το σύμβολο @ στη θέση της ημερομηνίας. Οι ημερομηνίες δίνονται στη μορφή «ΗΗ/ΜΜ/ΕΕΕΕ». Οι θερμοκρασίες έχουν μετρηθεί μέσα στο ίδιο έτος.

ΣΧΟΛΙΑ

```
Algorithm temperature
Variables
    days[366]:string
    tempr[366]: real
    max, min: real
    i, N: integer
Begin
    i <- 1
    read(days[i], tempr[i])
    while days[i] <> '@'
          i <- i + 1
          read(days[i], tempr[i])
    end while
    if i > 1 then
          max <- days[1]
          min <- days[1]
          N <- i - 1
          for i <- 1 to N step 1
                if tempr[i] > max then
                      max <- tempr[i]
                end if
                if tempr[i] < min then
                      min <- tempr[i]
                end if
          end for
          write('Η μέγιστη θερμοκρασία είναι: ', max)
          write('Σημειώθηκε στις:')
          for i <- 1 to N step 1
                if tempr[i] = max then
                      write(days[i])
                end if
          end for
          write('Η ελάχιστη θερμοκρασία είναι: ', min)
          write('Σημειώθηκε στις:')
          for i <- 1 to N step 1
                if tempr[i] = min then
                      write(days[i])
```

```
                end if
            end for
        end if
    end algorithm
```

26. Κατασκευάστε πρόγραμμα που να μετατρέπει έναν ακέραιο θετικό αριθμό από το δεκαδικό σύστημα στα συστήματα δυαδικό, οκταδικό και δεκαεξαδικό.

ΣΧΟΛΙΑ

```
Algorithm  10_to_2_8_16
Variables
    base[3]:integer
    result[100]: char
    N: integer
    i, j, k, m, x: integer
    temp: string

Begin
    base[1] <- 2
    base[2] <- 8
    base[3] <- 16

    read(N)
    for i <- 1 to 3 step 1
          m <- N
          j <- 0
          while m >0
                j <- j + 1
                x <- m mod base[i]
                if x<10 then
                result[j] <- χαρακτήρας με ascii αυτόν του
                             ('0'+x)
          else
                result[j] <- χαρακτήρας με ascii αυτόν του
                             ('A'+x-10)
          end if
                m <- m div base[i]
          end while
```

```
            temp <- κενό string
            for k <- j to 1 step -1
                  temp <- temp συνένωση result[k]
            end for
            write(N,' βάση 10 -> ',temp,'βάση ',base[i])
      end for
end algorithm
```

27. Ένας ταμίας θέλει να πληρώνει τους μισθούς σε μετρητά χρησιμοποιώντας τον ελάχιστο δυνατό αριθμό νομισμάτων. Κατασκευάστε ένα πρόγραμμα για να τον βοηθήσετε. Θεωρήστε οτι οι μισθοί είναι στρογγυλοποιημένοι στις 100 δραχμές.

ΣΧΟΛΙΑ

Ο ακόλουθος αλγόριθμος έχει τη μορφή υποπρογράμματος, ώστε να αξιοποιηθεί στην επόμενη άσκηση.

```
subroutine  anal_posou(poso: integer,
      analisi[6]: integer,
      nomism[6]:  integer)
Variables
      i:  integer
Begin
      for i <- 1 to 6 step 1
            analisi[i] <- poso div nomism[i]
            poso <- poso mod nomism[i]
      end for
end subroutine
```

28. Τροποποιήστε το προηγούμενο πρόγραμμα ώστε να μπορεί, με βάση τους μισθούς ενός μήνα, να υπολογίζει τον ακριβή αριθμό νομισμάτων που θα πρέπει να προμηθεύεται ο ταμίας από την τράπεζα για να πληρώσει όλους τους μισθούς χρησιμοποιώντας τον ελάχιστο δυνατό αριθμό νομισμάτων.

ΣΧΟΛΙΑ

Το τέλος των δεδομένων ορίζεται με την τιμή 0 στο ποσό.

```
Algorithm  Nomismata_B
Variables
```

```
    anal[6]: integer
    total[6]: integer
    poso: integer
    nomism[6]: integer
    i: integer
Begin
    nomism[1] <- 10000
    nomism[2] <- 5000
    nomism[3] <- 1000
    nomism[4] <- 500
    nomism[5] <- 200
    nomism[6] <- 100
    read(poso)

    for i <- 1 to 6 step 1
          total[i] <- 0
    end for

    while poso <> 0
          anal_posou(poso, anal, nomism)
          for i <- 1 to 6 step 1
                total[i] <- total[i] + anal[i]
          end for
          read (poso) {επομένως εργαζόμενος}
    end while
    for i <- 1 to 6 step 1
          write(nomism[i], ' x ', total[i])
    end for
end algorithm
```

29. Απλό παιχνίδι: Ο υπολογιστής «σκέπτεται» έναν αριθμό από το 1 έως το 1000. Στη συνέχεια προσπαθούμε να μαντέψουμε τον αριθμό με διαδοχικές δοκιμές. Σε κάθε μας προσπάθεια, ο υπολογιστής απαντά με ένα από τα:

- Μπράβο το βρήκες!
- Ο αριθμός που έδωσες είναι πιο μικρός.
- Ο αριθμός που έδωσες είναι πιο μεγάλος.

α) Γράψτε το κατάλληλο πρόγραμμα.

β) Μπορεί κάποιος τυχαία να βρει τον αριθμό στη πρώτη προσπάθεια, όμως είναι πιθανό να χρειαστεί και 1000 προσπάθειες. Μπορείτε να αναπτύξετε μια στρατηγική που να εγγυάται, στη χειρότερη περίπτωση, τον μικρότερο αριθμό προσπαθειών από οποιαδήποτε άλλη; Ποιος είναι αυτός ο αριθμός;

ΣΧΟΛΙΑ

Η άσκηση αυτή αποτελεί μια καλή ευκαιρία να τους μιλήσουμε για τη δυαδική αναζήτηση.

Επίσης προτείνεται να τους μιλήσετε για τη δημιουργία ψευδοτυχαίων αριθμών.

```
Algorithm  game
Variables
    num: integer
    i: integer
    a: integer
    found: boolean
Begin
    num <- ένας τυχαίος αριθμός από το 1-1000
    i <- 1
    found <- false
    repeat
          write('Προσπάθεια: ', i)
          read(a)
          if a = num then
                write(' Μπράβο το βρήκες!')
                found <- true
          else
    if a > num then
    write(' Ο αριθμός που έδωσες είναι πιο μεγάλος.')
          else
    write(' Ο αριθμός που έδωσες είναι πιο μικρός.')
          end if
    end if
    until found
end algorithm
```

30. Κατασκευάστε ένα πρόγραμμα που να δέχεται στην είσοδο τον αριθμητή και τον παρανομαστή ενός κλάσματος και να τυπώνει το κλάσμα απλοποιημένο.

ΣΧΟΛΙΑ

Η εύρεση του ΜΚΔ γίνεται στην άσκηση 5 του κεφαλαίου και έτσι εδώ μπορεί να χρησιμοποιηθεί ως υποπρόγραμμα. Αυτό σημαίνει ότι πρέπει να έχει προηγηθεί η άσκηση αυτή.

```
Algorithm  klasma
Variables
    arith, par: integer
    mkd: integer
Begin
    read(arith, par)
    mkd <- ΜΚΔ (arith, par)
    write(arith/mkd, '/', par/mkd)
end algorithm
```

31. Να κατασκευάσετε ένα πρόγραμμα που να δέχεται τους βαθμούς ενός μαθητή, προφορικούς και γραπτούς, καθώς και τον αριθμό των απουσιών του και να τυπώνει πληροφορίες σχετικές με το αν προάγεται ή όχι.

```
Algorithm  vathmoi
Variables
    grapta, prof, mesos: real
    apousies: integer
    orio: integer
Begin
    orio <- 30
    read(prof, grapta, apousies)
    mesos <- (grapta + prof)/2
    if (mesos >= 10) and (apousies <= orio) then
          if mesos > 18 then
                write('Προάγεται με ;arista: ', mesos)
          else
                write('Προάγεται με: ', mesos)
          end if
    else
```

```
            if (apousies > orio) and (mesos >= 10) then
                  write('Απορίπτεται λόγω απουσιών')
            else
                  if mesos < 10 then
                        write('Απορίπτεται: ', mesos)
            end if
         end if
      end if
end algorithm
```

32. Μετατρέψτε το πρόγραμμα της προηγούμενης άσκησης σε κατάλληλο υποπρόγραμμα και χρησιμοποιήστε το για να δώσετε τα αποτελέσματα μιας τάξης. Στο τέλος δώστε χρήσιμα στατιστικά στοιχεία. Π.χ. μέσο όρο, ποσοστό αυτών που αρίστευσαν, ποσοστό αυτών που προήχθησαν, κλπ.

ΣΧΟΛΙΑ

```
Algorithm  vathmoi_taxis
Variables
   N: integer {αριθμός μαθητών}
   name: string
   prof, grapt, mo: real
   apous: integer
   ok: boolen
   num_arist, num_ok: integer
Begin
   read(N)
   num_arist <- 0
   num_ok <- 0
   for i <- 1 to N step 1
         read(name, prof, grapt, apous)
         {Χρησιμοποιούμε την προηγούμενη άσκηση ως υπο-
         πρόγραμμα compute}
         compute(prof, grapt, apous, ok, mo)
         if ok then
               num_ok <- num_ok + 1
               if mo >= 18 then
                     num_arist <- num_arist + 1
               end if
         end if
```

```
    end for

    write('Σε σύνολο: ', N, 'Μαθητών')
    write('Προήχθησαν: ',num_ok,' ',num_ok/N*100,'%')
    write(`Απορίφθηκαν:',N-num_ok,' ', (N-
num_ok)/N*100,'%')

    write('Αρίστευσαν:',num_arist,' ',num_arist/N*100,'%')
end algorithm
```

33. Θέλουμε να κατασκευάσουμε ένα πρόγραμμα που να εξομοιώνει την οριζόντια βολή από κάποιο ύψος. Η γραφική παράσταση της βολής δεν είναι απαραίτητη. Στην είσοδο θα δέχεται τις διάφορες παραμέτρους οι οποίες καθορίζουν τη βολή (π.χ. το ύψος από το οποίο γίνεται η βολή) και θα τυπώνει την απόστασή της. Πειραματιστείτε δίνοντας διάφορες τιμές εισόδου και γράψτε τις παρατηρήσεις σας. Τι θα συμβεί αν η βολή γίνεται στη σελήνη;

ΣΧΟΛΙΑ

Οι μαθητές γνωρίζουν ότι η εξίσωση τροχιάς στην οριζόντια βολή είναι η $y = \frac{g}{2U_0^2} \cdot x^2$ *οπότε αν θέσουμε στο y το ύψος από το οποίο γίνεται η βολή, το x θα μας δώσει την απόσταση.*

```
Algorithm Voli
Variables
    x, y, u0: real
Begin
    read(y, u0)
    x <- u0 * (√2y/g)
    write(x)
end algorithm
```

34. Αναζητήστε και καταγράψτε από την ύλη των άλλων μαθημάτων θέματα στα οποία ένα πρόγραμμα θα μπορούσε να βοηθήσει στην κατανόησή τους. Επιλέξτε ένα από αυτά και κατασκευάστε το αντίστοιχο πρόγραμμα.

ΣΧΟΛΙΑ

Αυτή η άσκηση μπορεί να εξελιχθεί σε μια μικρή ή και μεγαλύτερη δραστηριότητα.

Ο στόχος μας είναι να αντιληφθούν οι μαθητές μας ότι ο υπολογιστής είναι ένα εργαλείο με το οποίο μπορούν να διερευνούν έννοιες και άλλων μαθημάτων.
Να παροτρύνετε τους μαθητές να συζητήσουν με συναδέλφους των άλλων ειδικοτήτων απλές περιπτώσεις που ένα πρόγραμμα θα τους βοηθούσε να κατανοήσουν καλύτερα κάποιες έννοιες.

35. Να κατασκευάσετε ένα απλό πρόγραμμα κρυπτογράφησης αρχείων κειμένου. Χρησιμοποιήστε την εξής μέθοδο: σε κάθε χαρακτήρα που μπορεί να υπάρχει στο αρχείο αντιστοιχούμε ένα μοναδικό τριψήφιο ακέραιο. Αυτή η αντιστοίχιση είναι το κρυφό κλειδί. Για παράδειγμα, αν στους χαρακτήρες g, !, 1, Δ αντιστοιχούν οι αριθμοί 003, 212, 800, 010, τότε, όπου στο αρχικό κείμενο εμφανίζονται αυτοί οι χαρακτήρες, στο κρυπτογραφημένο θα εμφανίζονται οι αντίστοιχοι αριθμοί.

ΣΧΟΛΙΑ

Η άσκηση αυτή σε μικρό τμήμα της χρησιμοποιεί τη έννοια του αρχείου.
Εφόσον γίνει η άσκηση αυτή, θα πρέπει να τους μιλήσετε για αρχεία κειμένου.

```
Algorithm  cryptografisi
Variables
    key[256]:string
    f1, f2:file
    c:char
    i: integer
    cryp: string
Begin
{Δημιουργία κλειδιού. Το κλειδί του κάθε χαρακτήρα τοποθετείται στη θέση που αντιστοιχεί ο ascii κωδικός του χαρακτήρα}
    key[65] <- '006'
    key[66] <- '123'
    ....
    ανοιγμα αρχείου f1 για διάβασμα
    ανοιγμα αρχείου f2 για γράψιμο
    read(f1, c)
    while not end_of_file f1
          cryp <- key[ascii code του c]
          write(f2, cryp)
          read(f1, c)
    end while
    κλείσιμο αρχείου f1
    κλείσιμο αρχείου f2
```

```
end algorithm
```

36. Κατασκευάστε ένα πρόγραμμα που να κάνει αποκρυπτογράφηση για την προηγούμενη μέθοδο.

ΣΧΟΛΙΑ

Εφόσον γίνει η άσκηση αυτή θα πρέπει να μιλήσετε στους μαθητές για αρχεία κειμένου.

```
Algorithm  decrypt
Variables
    key[256]: string
    f1,  f2: file
    st: string[3]
    c:  char
    i:  integer
    cryp:  string
Begin
{Δημιουργία κλειδιού. Το κλειδί του κάθε χαρακτήρα τοποθετείται στη θέση που αντιστοιχεί ο ascii κωδικός του χαρακτήρα}
    key[65] <- '006'
    key[66] <- '123'
    ....
    ανοιγμα αρχείου f1 για διάβασμα
    ανοιγμα αρχείου f2 για γράψιμο
    read(f1, st)
    while not end_of_file f1
          i <- 1
          {ευρεση χαρακτήρα από κλειδί}
          while key[i] <> st
                i <- i + 1
          end while
          c <- χαρακτήρας με ascii code i
          write(f2, c)
          read(f1, st)
    end while
    κλείσιμο αρχείου f1
    κλείσιμο αρχείου f2
end algorith
```