ΚΕΦΑΛΑΙΟ 1

Εισαγωγή στην Έννοια του Αλγορίθμου και στον Προγραμματισμό

Εισαγωγή

Στις προηγούμενες τάξεις αναφέρθηκε ότι ο υπολογιστής μπορεί να μας υποστηρίζει σε διάφορες δραστηριότητές μας, επιτελώντας απλές λειτουργίες (π.χ. αριθμητικές πράξεις) με μεγάλη ταχύτητα. Μπορούμε, όμως, να χρησιμοποιήσουμε τον υπολογιστή και στην επίλυση πιο σύνθετων προβλημάτων. Στην Ενότητα αυτή θα θέσουμε τον υπολογιστή στην υπηρεσία μας, δημιουργώντας τα δικά μας προγράμματα. Ήρθε η ώρα να δημιουργήσουμε ακόμα και τα δικά μας παιχνίδια.

✔ Τι είναι πρόβλημα;
✔ Πώς μπορούμε να περιγράψουμε με σαφήνεια τη λύση ενός προβλήματος;
✔ Σε ποια γλώσσα «καταλαβαίνει» ο υπολογιστής τις εντολές που του δίνουμε;

Στο Κεφάλαιο που ακολουθεί θα προσπαθήσουμε να προσδιορίσουμε τι είναι πρόβλημα και θα μάθουμε να περιγράφουμε με σαφήνεια τη λύση του.

Λέξεις Κλειδιά

Αλγόριθμος, Γλώσσες Προγραμματισμού, Δεδομένα Προβλήματος, Εντολή, Ζητούμενα, Κατανόηση Προβλήματος, Πρόβλημα, Πρόγραμμα, Προγραμματισμός, Προγραμματιστής

1.1 Η έννοια του προβλήματος

Τη λέξη πρόβλημα την έχετε συναντήσει πολλές φορές από τις πρώτες τάξεις του σχολείου. Έχετε λύσει, για παράδειγμα, προβλήματα στα Μαθηματικά και τη Φυσική. Προβλήματα, όμως, αντιμετωπίζουμε και καθημερινά, όπως: ποιος είναι ο πιο σύντομος δρόμος, για να πάμε στο σχολείο μας, πώς να οργανώσουμε μία εκδρομή, πώς να τακτοποιήσουμε τα βιβλία στη βιβλιοθήκη, ώστε να τα βρίσκουμε ευκολότερα. Τα προβλήματα που μόλις αναφέραμε είναι σχετικά απλά και σύντομα βρίσκουμε τη λύση τους. Πολλά προβλήματα, όμως, είναι πιο πολύπλοκα και η επίλυσή τους μας δυσκολεύει ιδιαίτερα. Για παράδειγμα, η ρύπανση της ατμόσφαιρας, η εξοικονόμηση ενέργειας, η θεραπεία ορισμένων ασθενειών, η εξερεύνηση του διαστήματος και η κατασκευή μιας γέφυρας μεγάλου μήκους, είναι ιδιαίτερα σύνθετα προβλήματα. Υπάρχουν επίσης και άλλες κατηγορίες προβλημάτων που:

- είτε δεν μπορούμε να τα επιλύσουμε με τις μέχρι τώρα γνώσεις μας, όπως η ακριβής πρόβλεψη των σεισμών, η γήρανση του ανθρώπου, η ανακάλυψη εξωγήινων πολιτισμών και η επικοινωνία μαζί τους,
- είτε έχει αποδειχθεί ότι δεν μπορούμε να τα επιλύσουμε, όπως: ο τετραγωνισμός του κύκλου με κανόνα και διαβήτη ή το ταξίδι στο παρελθόν.

Τα προβλήματα που καλούμαστε να επιλύσουμε στο σχολείο είναι συνήθως υπολογιστικά και απαιτούν μια σειρά από λογικές σκέψεις και μαθηματικές πράξεις. Για παράδειγμα: «ποιο είναι το εμβαδόν ενός τετραγώνου με πλευρά μήκους 10 εκατοστών;», «σε πόσο χρόνο θα πέσει ένα αντικείμενο που εκτελεί ελεύθερη πτώση από ύψος 10 μέτρων;» Παρόμοια υπολογιστικά προβλήματα συχνά καλούμαστε να επιλύσουμε και στην καθημερινή μας ζωή, όπως: «ποιος είναι ο μέσος όρος της βαθμολογίας μου;», «τι διαστάσεις πρέπει να έχει το γραφείο που θα αγοράσω, για να χωράει στο δωμάτιο μου;», «πόσα χρήματα χρειαζόμαστε, για να αγοράσουμε τον αγαπημένο μας δίσκο μουσικής, όταν η αρχική του τιμή είναι 15 € και έχει έκπτωση 20%;».

Γενικότερα, ως πρόβλημα θεωρούμε κάθε ζήτημα που τίθεται προς επίλυση, κάθε κατάσταση που μας απασχολεί και πρέπει να αντιμετωπιστεί. Η λύση ενός προβλήματος δεν μας είναι γνωστή, ούτε προφανής.

Η πρώτη μας ενέργεια για να λύσουμε πιο εύκολα ένα πρόβλημα, είναι η καταγραφή των δεδομένων. **Δεδομένα προβλήματος** είναι τα στοιχεία που μας είναι γνωστά και μπορούν να μας βοηθήσουν στη λύση του προβλήματος. Σε κάθε πρόβλημα ψάχνουμε να βρούμε την απάντηση σε μια ερώτηση. Αυτό που ψάχνουμε είναι το **ζητούμενο**. Για παράδειγμα, το ζητούμενο σε μια κατασκήνωση μπορεί να είναι το στήσιμο της σκηνής ή ο καταμερισμός των εργασιών. Σε μια παρτίδα σκάκι ζητούμενο είναι οι κατάλληλες κινήσεις που θα μας οδηγήσουν σε «ματ» του αντίπαλου βασιλιά. Σε ένα γεωμετρικό πρόβλημα ζητούμενο μπορεί να είναι το μήκος ενός ευθυγράμμου τμήματος. Η διαδικασία μέσω της οποίας βρίσκουμε το ζητούμενο και επιτυγχάνουμε τον επιθυμητό στόχο ονομάζεται **επίλυση προβλήματος**. Υπάρχουν προβλήματα, των οποίων τη λύση μπορούμε να περιγράψουμε με ακρίβεια (π.χ.: ο υπολογισμός της υποτείνουσας ορθογωνίου τριγώνου) και προβλήματα που δεν έχουν ακριβή λύση (π.χ.: η αξιοποίηση του ελεύθερου χρόνου μας). Ακόμα πολλές φορές πρέπει να ελέγχουμε, αν τα δεδομένα του προβλήματος που έχουμε είναι επαρκή, ώστε να μπορούμε να σχεδιάσουμε την επίλυσή του.

Πολλές φορές η λύση ενός προβλήματος χρειάζεται περισσότερη διερεύνηση. Για παράδειγμα στο επόμενο πρόβλημα:

Ένας εργάτης χτίζει 1 μέτρο τοίχο σε 2 ώρες. Σε πόσο χρόνο θα έχει ολοκληρώσει το χτίσιμο 11 μέτρων, αν δουλέψει μόνος του;

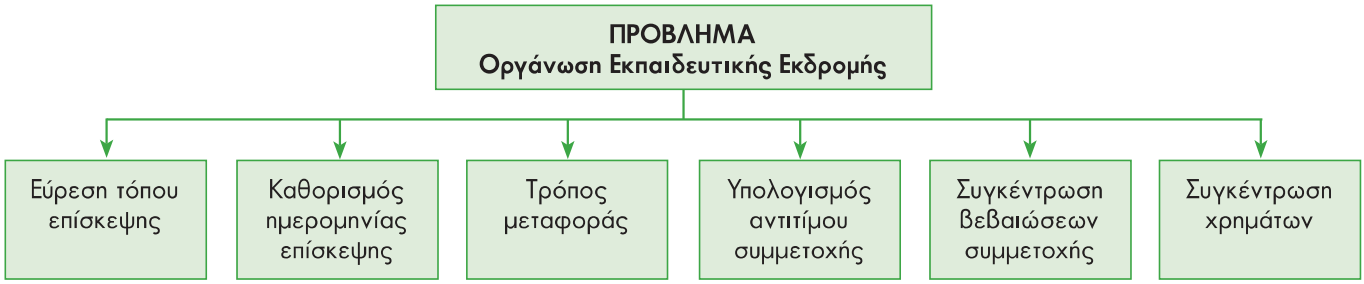
Η απάντηση: *σε 22 ώρες* φαίνεται λογική, αλλά ξεχνάμε ότι ένας εργάτης δεν μπορεί να δουλέψει 22 ώρες συνεχόμενες!

Έτσι, για να επιλύσουμε ένα πρόβλημα πρέπει αρχικά να το **κατανοήσουμε**. Πρέπει δηλαδή να καταλάβουμε καλά το περιεχόμενό του, να διακρίνουμε τα δεδομένα που έχουμε στη διάθεσή μας και τα ζητούμενά του. Είναι σημαντικό, όμως, να προσδιορίσουμε και το **«περιβάλλον»** ή το πλαίσιο μέσα στο οποίο εντάσσεται το πρόβλημα (χώρος του προβλήματος).

Για παράδειγμα, στο σύνολο των φυσικών αριθμών η αφαίρεση 3–9 είναι αδύνατη, ενώ στο σύνολο των ακεραίων αριθμών η ίδια αφαίρεση έχει αποτέλεσμα 3–9=–6. Στο παράδειγμα της οργάνωσης μιας εκδρομής το «περιβάλλον» του προβλήματος είναι το σχολικό περιβάλλον. Η οργάνωση μιας εκπαιδευτικής εκδρομής έχει αρκετά διαφορετικά στοιχεία από την οργάνωση μιας εκδρομής με φίλους. Μια εκπαιδευτική εκδρομή πρέπει να πραγματοποιηθεί μέσα στα πλαίσια των κανόνων που καθορίζονται από το σχολικό περιβάλλον, ενώ μια εκδρομή με φίλους ακολουθεί διαφορετικούς κανόνες.

Στην πραγματικότητα, τα περισσότερα προβλήματα είναι σύνθετα και δε μας έρχεται στο νου η λύση τους με την πρώτη ματιά. Χρειάζεται πολλές φορές να τα μελετήσουμε σε βάθος και να εξερευνήσουμε διαφορετικούς πιθανούς τρόπους επίλυσής τους. Όσο περισσότερο μελετάμε ένα πρόβλημα, τόσο πιο πιθανό είναι να το επιλύσουμε. Συχνά μάλιστα η λύση του μας έρχεται σαν αναλαμπή, σε άσχετη φαινομενικά στιγμή. Αρκεί να θυμηθούμε το πρόβλημα του Αρχιμήδη που βασάνιζε για καιρό το μυαλό του –πώς θα μπορέσει να αποδείξει ότι το στέμμα του βασιλιά αποτελείται μόνο από χρυσάφι ή από πρόσμιξη και άλλων μετάλλων ίδιου βάρους– και όταν ξαφνικά βρήκε τη λύση την ώρα που έκανε μπάνιο, πήδησε έξω ενθουσιασμένος φωνάζοντας «Εύρηκα!».

Για να μπορέσουμε να επιλύσουμε ένα σύνθετο πρόβλημα, είναι αναγκαίο να το αναλύσουμε σε απλούστερα προβλήματα. Για παράδειγμα, η οργάνωση μίας σχολικής εκδρομής (Σχήμα 1.1), αν και φαίνεται απλή, είναι ένα σύνθετο πρόβλημα. Για την καλύτερη επίλυσή του μπορούμε να το χωρίσουμε σε μια σειρά από απλούστερα



Σχήμα 1.1. Ανάλυση του προβλήματος «Οργάνωση Εκπαίδευσης Εκδρομής» σε απλούστερα προβλήματα.

Η καταγραφή της ανάλυσης ενός προβλήματος καθώς και των βημάτων για την επίλυσή του είναι πολύ χρήσιμη τις επόμενες φορές που θα χρειαστεί να λύσουμε παρόμοια προβλήματα.

Εισαγωγική Δραστηριότητα

Προσπαθήστε να δώσετε σε κάποιο συμμαθητή σας σαφείς και ακριβείς οδηγίες, για να παρασκευάσει ένα ποτήρι φρέσκο χυμό πορτοκαλιού.

προβλήματα. Αντιμετωπίζοντας καθένα από τα απλούστερα προβλήματα ξεχωριστά, στο τέλος θα καταφέρουμε να επιλύσουμε και το πιο πολύπλοκο πρόβλημα της «οργάνωσης σχολικής εκδρομής».

Η περιγραφή της λύσης ενός προβλήματος, όμως, περιέχει συχνά δυσκολίες. Όταν θέλουμε να δώσουμε οδηγίες σε κάποιον, για να κάνει μια σύνθετη εργασία, διαπιστώνουμε πόσο δύσκολη είναι η **διατύπωση σωστών οδηγιών**.

Οι σαφείς και απλές στη διατύπωσή τους οδηγίες είναι περισσότερο απαραίτητες, όταν στην προσπάθεια επίλυσης ενός προβλήματος συμμετέχουν περισσότεροι άνθρωποι, που πρέπει να συνεργαστούν μεταξύ τους (στην επίλυση του προβλήματος της εκδρομής του σχολείου συμμετέχουν ο Διευθυντής, οι καθηγητές και οι μαθητές που θα βοηθήσουν).

Αν τύχει και ταξιδέψετε με πλοίο προς ένα από τα όμορφα νησιά της πατρίδας μας, θα παρατηρήσετε ότι σε εμφανή σημεία του πλοίου υπάρχει αναρτημένος ένας κατάλογος με τέσσερις απλές οδηγίες για το πώς μπορούμε να βάλουμε ένα σωσίβιο θαλάσσης σε περιπτώσεις ανάγκης. Οι οδηγίες αυτές έχουν διατυπωθεί σε ξεχωριστά βήματα – ενέργειες, με λογική σειρά και με απλά λόγια, ώστε ο καθένας να μπορεί να τις καταλάβει και να είναι σε θέση να τις εκτελέσει.

1.2 Τι είναι Αλγόριθμος

Οι οδηγίες που δίνουμε με λογική σειρά, ώστε να εκτελέσουμε μια εργασία ή να επιλύσουμε ένα πρόβλημα, συνθέτουν έναν **Αλγόριθμο**. Για παράδειγμα, οι οδηγίες για την κατασκευή ενός χαρταετού μπορεί να αποτελέσουν έναν αλγόριθμο. ***Αλγόριθμο ονομάζουμε τη σαφή και ακριβή περιγραφή μιας σειράς ξεχωριστών οδηγιών-βημάτων, με σκοπό την επίλυση ενός προβλήματος.***

Αλγόριθμος μπορεί να είναι μια συνταγή μαγειρικής ή η βήμα προς βήμα περιγραφή της λύσης ενός μαθηματικού προβλήματος. Όταν σχεδιάζουμε έναν αλγόριθμο, πρέπει να είμαστε ιδιαίτερα προσεκτικοί, ώστε να βάζουμε με **λογική σειρά** τις **οδηγίες (instructions)** που θα μας οδηγήσουν στη λύση του προβλήματός μας. Αν, για παράδειγμα, δεν περιγράψουμε σωστά τα βήματα που πρέπει να ακολουθηθούν, ώστε να μαγειρέψει ένας άπειρος μάγειρας μια μακαρονάδα, τότε είναι πιθανό να μείνουμε νηστικοί.

1. Άνοιξε το μάτι της κουζίνας στο 2.
2. Βάλε 3 λίτρα νερό σε μία κατσαρόλα χωρητικότητας 4 λίτρων.
3. Τοποθέτησε την κατσαρόλα στο μάτι της κουζίνας, που έχεις ήδη ανάψει.
4. Πρόσθεσε στην κατσαρόλα μία κουταλιά του καφέ αλάτι.
5. Περίμενε μέχρι να βράσει το νερό.
6. Βγάλε τα μακαρόνια από το πακέτο.
7. Βάλε τα μακαρόνια στην κατσαρόλα.
8. Ανακάτευε τα μακαρόνια για 10 λεπτά.

9. Κλείσε το μάτι της κουζίνας που άνοιξες.
10. Βγάλε την κατσαρόλα από το μάτι της κουζίνας.
11. Άδειασε τα μακαρόνια από την κατσαρόλα σε ένα σουρωτήρι.
12. Ρίξε κρύο νερό από τη βρύση στα μακαρόνια για 20 δευτερόλεπτα.
13. Άφησε για 2 λεπτά τα μακαρόνια να στραγγίξουν.
14. Σερβίρισε τα μακαρόνια στο πιάτο.
15. Πρόσθεσε σε κάθε πιάτο 3 κουταλιές της σούπας τριμμένο τυρί.

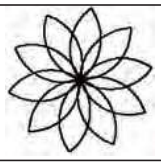
Πριν προχωρήσουμε παρακάτω προσπάθησε να απαντήσεις στις ακόλουθες ερωτήσεις:
1. Τι θα συμβεί, αν ξεχάσουμε την οδηγία 9 στον παραπάνω αλγόριθμο;
2. Μπορούμε να αντιμεταθέσουμε τις οδηγίες 7 και 8;
3. Τι θα συμβεί, αν αντικαταστήσουμε την οδηγία στο βήμα 4 με την οδηγία «πρόσθεσε αλάτι»;
4. Αν αντιμεταθέσουμε τις οδηγίες 1 και 2, θα υπάρξει κάποιο πρόβλημα στον αλγόριθμο;

1.3 Ιδιότητες ενός Αλγορίθμου

Τα βήματα που αποτελούν έναν αλγόριθμο ονομάζονται **οδηγίες** ή **εντολές**. Αν ακολουθηθούν οι οδηγίες ενός αλγορίθμου στο τέλος πρέπει να προκύπτει ένα αποτέλεσμα, ένα έργο. Για παράδειγμα, αν ακολουθήσουμε τις οδηγίες μιας συνταγής μαγειρικής θα παραγάγουμε το επιθυμητό φαγητό. Μια παρτιτούρα περιέχει οδηγίες· αν γνωρίζουμε μουσική και τις εφαρμόσουμε σε ένα μουσικό όργανο, παράγουμε μουσική.

Όπως περιγράψαμε στα προηγούμενα παραδείγματά μας, για να μπορέσουμε από έναν αλγόριθμο να πάρουμε αποτελέσματα χρειαζόμαστε κάποιον που θα υλοποιήσει τον αλγόριθμο, δηλαδή κάποιον που θα ακολουθήσει τις οδηγίες που περιλαμβάνει ο αλγόριθμος. Αυτός που υλοποιεί τον αλγόριθμο μπορεί να είναι ένας άνθρωπος ή ένας υπολογιστής. Για την υλοποίηση μιας συνταγής μαγειρικής υπεύθυνος είναι ο μάγειρας. Για τον υπολογισμό του εμβαδού ενός τετραγώνου αυτός που θα υλοποιήσει τον αλγόριθμο μπορεί να είναι ένας υπολογιστής.

Οι αλγόριθμοι που κατασκευάζουμε πρέπει να πληρούν κάποιες προϋποθέσεις. Πρώτα απ' όλα, πρέπει να είμαστε σίγουροι ότι, αν υλοποιήσουμε τον αλγόριθμο, **κάποτε θα τελειώσει** επιτυγχάνοντας τον αρχικό σκοπό. Φανταστείτε να δώσουμε μία εντολή σε ένα δρομέα, να αρχίσει να τρέχει και να μην του πούμε πότε θα σταματήσει. Όμοια, αν δώσουμε εντολή σε έναν υπολογιστή, ώστε να ζωγραφίσει τα δέκα πέταλα ενός λουλουδιού, πρέπει να αναφέρουμε τον αριθμό των πετάλων που θα έχει το λουλούδι (δέκα), ώστε να είμαστε βέβαιοι ότι ο υπολογιστής θα σταματήσει το σχεδιασμό μόλις σχηματιστεί το λουλούδι.

Αλγόριθμος δημιουργίας ενός λουλουδιού με 10 πέταλα	Το αποτέλεσμα υλοποίησης του Αλγορίθμου
`επανάλαβε 10 φορές [σχεδίασε_πέταλο]`	

Αντίθετα η οδηγία:

`επανάλαβε συνεχώς [σχεδίασε_πέταλο]`

δεν μπορεί να χαρακτηριστεί αλγόριθμος, γιατί ο υπολογιστής θα σχεδιάζει πέταλα **συνεχώς** χωρίς να σταματήσει ποτέ!

Οι εντολές ενός αλγορίθμου πρέπει να έχουν **ακρίβεια** και **σαφήνεια**, ώστε να μην μπερδευτεί αυτός που θα υλοποιήσει τον αλγόριθμο και τις εκτελέσει με λανθασμένο τρόπο. Σε μια συνταγή μαγειρικής, για παράδειγμα, πρέπει να περιγρά-

Ιστορικά Στοιχεία για τους Αλγόριθμους

Ο πέρσης μαθηματικός Mohammed ibn-Musa al-Khuwarizmi (780-850 μ.Χ.) εισήγαγε την έννοια του αλγορίθμου αναφερόμενος σε μια μαθηματική επεξεργασία αριθμών. Για την ονομασία αυτής της διαδικασίας χρησιμοποιήθηκε στην αρχή η λατινική λέξη algorismus, που δημιουργήθηκε από την παραφθορά του συνθετικού του ονόματος al-Khuwarizmi (ο άνθρωπος από την πόλη Khuwarizmi). Στα τέλη του 17ου αιώνα η ονομασία συνδυάστηκε με την ελληνική λέξη αριθμός και μετατράπηκε στη λέξη αλγόριθμος (algorithm).

ψουμε ακριβώς την ποσότητα αλατιού που θα ρίξει ο μάγειρας (μία κουταλιά του καφέ, ή 10 γρ.). Όταν δώσουμε μία εντολή στον υπολογιστή να εμφανίσει ένα μήνυμα, πρέπει να του περιγράψουμε πού θα το εμφανίσει (στην οθόνη ή στον εκτυπωτή), σε ποιο σημείο, με τι μέγεθος, σε ποια χρονική στιγμή κ.λπ.

Τέλος, οι εντολές ενός αλγορίθμου πρέπει να είναι **εκφρασμένες με απλά λόγια**, ώστε να είναι απόλυτα κατανοητές.

Δεν πρέπει να ξεχνάμε ότι ο αλγόριθμος είναι η περιγραφή της λύσης ενός προβλήματος με μια συγκεκριμένη διαδοχική σειρά βημάτων. Για να μπορέσουμε να περιγράψουμε σε κάποιον τα βήματα που οδηγούν στη λύση ενός προβλήματος, πρέπει πρώτα να έχουμε κατανοήσει το πρόβλημα, να βρούμε τη λύση του και στη συνέχεια να περιγράψουμε τη λύση αυτή με μορφή αλγορίθμου.

Ας δούμε δύο παραδείγματα, για να κατανοήσουμε καλύτερα τη διαδικασία σχεδίασης ενός αλγορίθμου:

1ο Παράδειγμα: *«Έχει κάποιος ένα πρόβατο, ένα λύκο και ένα καφάσι με χόρτα στη μία όχθη ενός ποταμού και θέλει να τα περάσει στην απέναντι όχθη χρησιμοποιώντας μία βάρκα. Η βάρκα όμως είναι μικρή και μπορεί να μεταφέρει, εκτός από τον ίδιο, άλλο ένα από τα ζώα ή το καφάσι. Ωστόσο δεν πρέπει να μείνουν μαζί ο λύκος με το πρόβατο και το πρόβατο με τα χόρτα. Μπορείτε να δώσετε οδηγίες στον βαρκάρη για το πώς πρέπει να κάνει τη μεταφορά τους;»*

Πριν δώσουμε οδηγίες, πρέπει να κατανοήσουμε το πρόβλημα, να σκεφτούμε τις πιθανές λύσεις, να επιλέξουμε την πιο κατάλληλη και στη συνέχεια να περιγράψουμε με ακρίβεια τη λύση στον βαρκάρη.

Δεδομένα:	1 πρόβατο, 1 λύκος, 1 καφάσι με χόρτα, μία θέση επιπλέον στη βάρκα, 2 όχθες ποταμού.
Πλαίσιο του προβλήματος:	Ο λύκος δεν πρέπει να μείνει μαζί με το πρόβατο. Το πρόβατο δεν πρέπει να μείνει μαζί με τα χόρτα.
Ζητούμενο:	Να περάσει ο λύκος, το πρόβατο και το καφάσι με τα χόρτα στην απέναντι όχθη.

Εικόνα 1.1. Σχηματική αναπαράσταση του προβλήματος

Για να κατανοήσουμε καλύτερα το περιβάλλον του προβλήματος, μπορούμε να κάνουμε μια σχηματική αναπαράστασή του στο χαρτί, όπως στην Εικόνα 1.1.

Τώρα είμαστε έτοιμοι να σκεφτούμε τις πιθανές λύσεις του προβλήματος. Μετά από διάφορες σκέψεις και πειραματισμούς διαπιστώνουμε ότι μπορούμε να αφήνουμε το λύκο με το καφάσι μαζί και ότι χρειάζεται μερικές φορές να μεταφέρουμε και από την απέναντι στην αρχική όχθη κάποιο ζώο ή το καφάσι.

Η τελική περιγραφή της λύσης έχει ως εξής:

Αρχή του αλγορίθμου

1. Βάλε το πρόβατο στη βάρκα.
2. Πήγαινε στην απέναντι όχθη.
3. Άφησε το πρόβατο στην όχθη.
4. Γύρνα πίσω στην αρχική όχθη.
5. Φόρτωσε το καφάσι με τα χόρτα.
6. Πήγαινε στην απέναντι όχθη.
7. Άφησε το καφάσι στην όχθη.
8. Βάλε το πρόβατο στη βάρκα.
9. Πήγαινε στην αρχική όχθη.
10. Άφησε το πρόβατο στην όχθη.
11. Βάλε το λύκο στη βάρκα.
12. Πήγαινε στην απέναντι όχθη.
13. Άφησε τον λύκο στην όχθη.
14. Γύρνα πίσω στην αρχική όχθη.
15. Βάλε το πρόβατο στη βάρκα
16. Πήγαινε στην απέναντι όχθη.
17. Άφησε το πρόβατο στην όχθη.

Τέλος του αλγορίθμου

Ο Βαρκάρης ακολουθώντας πιστά (υλοποιώντας) τις οδηγίες του αλγορίθμου μπορεί να μεταφέρει με επιτυχία τα ζώα του και το καφάσι με τα χόρτα στην απέναντι όχθη του ποταμού.

2ο Παράδειγμα

Θέλουμε να περιγράψουμε σε ένα μικρό παιδί πώς θα δημιουργήσει με τις πατούσες του ένα τετράγωνο στην άμμο. Αν το παιδί δε γνωρίζει τι σχήμα θέλουμε να αποτυπωθεί στην άμμο, ποιες είναι οι κατάλληλες οδηγίες που πρέπει να του δώσουμε;

Κατ' αρχάς πρέπει να αναλύσουμε την έννοια «τετράγωνο»:

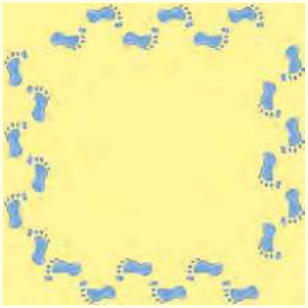
- Ένα τετράγωνο είναι ένα κλειστό γεωμετρικό σχήμα με **4 ίσες πλευρές**. Άρα, για να σχηματίσουμε τις πλευρές, πρέπει κάθε φορά να κάνουμε τον ίδιο αριθμό βημάτων.
- Ένα τετράγωνο έχει 4 ορθές γωνίες δηλαδή **4 γωνίες των 90°**. Άρα, μόλις σχηματίζουμε μία πλευρά πρέπει να γυρνάμε κατά 90° γύρω από τον εαυτό μας και πάντοτε με την ίδια φορά.

Αφού έχουμε κατανοήσει την έννοια «τετράγωνο», μπορούμε να πειραματιστούμε δίνοντας τις ακόλουθες οδηγίες στο παιδί:

Αρχή του αλγορίθμου

1. Περπάτησε 5 βήματα μπροστά.
2. Στρίψε δεξιά κατά ενενήντα μοίρες.
3. Περπάτησε 5 βήματα μπροστά.
4. Στρίψε δεξιά κατά ενενήντα μοίρες.
5. Περπάτησε 5 βήματα μπροστά.
6. Στρίψε δεξιά κατά ενενήντα μοίρες.
7. Περπάτησε 5 βήματα μπροστά.

Τέλος του αλγορίθμου



Η υλοποίηση του παραπάνω αλγορίθμου ήταν επιτυχής στο σχεδιασμό ενός τετραγώνου. Μερικές φορές, όμως, ένας αλγόριθμος μπορεί να μη μας δώσει τα προσδοκώμενα αποτελέσματα. Τότε είμαστε υποχρεωμένοι να γυρίσουμε πίσω στις εντολές που δώσαμε και να ελέγξουμε πού κάναμε λάθος. Στη συνέχεια αντικαθιστούμε τις λανθασμένες εντολές με τις σωστές και υλοποιούμε ξανά τον αλγόριθμο. Αυτή η ανατροφοδοτούμενη μορφή σχεδιασμού μας βοηθάει να καταλάβουμε καλύτερα το πρόβλημα και την επίλυσή του. Αν, για παράδειγμα, κάνουμε λάθος στο σχεδιασμό του αλγορίθμου του τετραγώνου, η διαδικασία εύρεσης του λάθους θα μας βοηθήσει να κατανοήσουμε καλύτερα την έννοια του τετραγώνου.

1.4 Υλοποίηση Αλγορίθμου με υπολογιστή – Προγραμματισμός

Τα πολλά διαφορετικά προγράμματα που μπορεί να εκτελεστούν σε έναν υπολογιστή, αποτελούν τον κύριο λόγο που χρησιμοποιούμε σήμερα τους υπολογιστές για διαφορετικές χρήσεις. Οι υπολογιστές χρησιμοποιούνται σε επιχειρήσεις-οργανισμούς, στη δημόσια διοίκηση, σε εκπαιδευτικά ιδρύματα, αλλά και σε σπίτια. Κάθε υπολογιστής γίνεται μια διαφορετική μηχανή ανάλογα με το πρόγραμμα που εκτελεί και αυτό είναι το μεγάλο του πλεονέκτημα. Τι είναι όμως ένα πρόγραμμα;

Ένα **πρόγραμμα** είναι η αναπαράσταση ενός αλγορίθμου γραμμένη σε γλώσσα κατανοητή για έναν υπολογιστή. Ένα πρόγραμμα, δηλαδή, αποτελείται από μία σειρά **εντολών** που δίνονται στον υπολογιστή με σκοπό να εκτελέσει κάποια συγκεκριμένη λειτουργία ή να υπολογίσει κάποιο επιθυμητό αποτέλεσμα. Η εργασία σύνταξης των προγραμμάτων ονομάζεται **προγραμματισμός**, ενώ τα άτομα που γράφουν και συντάσσουν ένα πρόγραμμα ονομάζονται **προγραμματιστές**.

Οι πύργοι του Ανόι

Ο Θρύλος: Σε κάποιους Ινδούς μοναχούς δόθηκε η δοκιμασία να μετακινήσουν 64 εύθραυστους δίσκους από μία τοποθεσία σε μια άλλη, έναν κάθε φορά, αποφεύγοντας την τοποθέτηση ενός μεγαλύτερου δίσκου πάνω σε έναν μικρότερο. Υπήρχε μόνο μια ακόμα ενδιάμεση τοποθεσία, πέρα από τις δύο, που ένας δίσκος μπορούσε να τοποθετηθεί.

Το παιχνίδι: Υπάρχει ένα παιχνίδι βασισμένο στο μύθο. Έχετε μια μικρή συλλογή από δίσκους και τρεις πασσάλους πάνω στους οποίους μπορείτε να τους τοποθετήσετε (ο κάθε δίσκος έχει στη μέση μία τρύπα ώστε να τοποθετείται στον πάσσαλο). Οι δίσκοι είναι όλοι τοποθετημένοι στον αριστερό πάσσαλο σε αύξουσα σειρά ανάλογα με το μέγεθός τους. Θα πρέπει να τους μετακινήσεις στο δεξιό πάσσαλο χωρίς ποτέ όμως να βάλεις έναν μεγαλύτερο δίσκο πάνω σε έναν μικρότερο. Καταγράψτε τον κατάλληλο αλγόριθμο που να περιγράφει πώς να μεταφέρετε τους δίσκους από τον αριστερό πάσσαλο στον δεξιό. (Ο ελάχιστος αριθμός βημάτων του αλγορίθμου είναι: 3 βήματα για 2 δίσκους, 7 βήματα για 3 δίσκους, 15 βήματα για 4 δίσκους και 31 βήματα για 5 δίσκους).

Ο μύθος λέει πως όταν οι μοναχοί καταφέρουν να μετακινήσουν τους 64 δίσκους στη νέα τοποθεσία, τότε ο ναός τους θα καταρρεύσει και θα μετατραπεί σε σκόνη και ακόμα ο κόσμος θα καταστραφεί.

Όλα τα προγράμματα του υπολογιστή αποτελούνται από ένα πλήθος κατάλληλων εντολών, που είναι γραμμένες σε λογική σειρά. Τα παιχνίδια, ο Επεξεργαστής Κειμένου, η Ζωγραφική, το Λειτουργικό Σύστημα αποτελούνται από ένα πλήθος εντολών κατανοητών από τον υπολογιστή (Εικόνα 1.2). Κάθε φορά που χρειαζόμαστε ένα πρόγραμμα, για να εκτελέσουμε μια λειτουργία ή να επιλύσουμε κάποιο πρόβλημα, ένα σύνολο εντολών αποθηκεύονται («φορτώνονται») στη μνήμη του υπολογιστή, για να εκτελεστούν στη συνέχεια πιστά από την Κεντρική Μονάδα Επεξεργασίας. (Δείτε επίσης την Εικόνα 5.2 της Α' Γυμνασίου).

Στο επόμενο κεφάλαιο θα μάθουμε και εμείς να προγραμματίζουμε τον υπολογιστή, ώστε να δημιουργούμε τα δικά μας προγράμματα. Τα προγράμματα που θα αναπτύξουμε μπορεί να είναι απλά στην αρχή, αλλά οι βασικές αρχές του προγραμματισμού είναι παρόμοιες και στα πιο σύνθετα προγράμματα. Με τον καιρό θα διαπιστώσετε ότι μπορείτε να δημιουργείτε όλο και πιο σύνθετα προγράμματα, παιχνίδια, εκπαιδευτικά προγράμματα, ή ιστοσελίδες και να επιλύετε διάφορα προβλήματα με τη βοήθεια του υπολογιστή.


```
void DisplayBlock(SBlock Block)
{
    if (Block.nY < 1) return;
    RECT rcBlock = g_rcBlock;
    rcBlock.left = Block.nColor * BLOCK_DIAMETER;
    rcBlock.right = Block.nColor * BLOCK_DIAMETER +      BLOCK_DIAMETER;

g_pDisplay->Blt( (DWORD)Block.nX * BLOCK_DIAMETER - 2 ,
                 (DWORD)Block.nY * BLOCK_DIAMETER ,
                 g_pSecondarySurface, &rcBlock );
    }
```

Εικόνα 1.2. Το γνωστό παιχνίδι ΤΕΤΡΙΣ είναι ένα πρόγραμμα το οποίο περιλαμβάνει μια σειρά εντολών (ένα μικρό υποσύνολο των εντολών του μπορείτε να δείτε στα δεξιά της εικόνας).

1.5 Γλώσσες Προγραμματισμού

Διαβάζοντας τα παραπάνω μπορεί κάποιος να αναρωτηθεί σε ποια γλώσσα μπορούμε να προγραμματίσουμε έναν υπολογιστή. Οι γλώσσες που «καταλαβαίνουν» οι υπολογιστές είναι τεχνητές γλώσσες που ονομάζονται **γλώσσες προγραμματισμού**. Οι γλώσσες προγραμματισμού χρησιμοποιούνται για την επικοινωνία του ανθρώπου με τη μηχανή, όπως οι φυσικές γλώσσες (ελληνική, αγγλική, γαλλική κ.λπ.) χρησιμοποιούνται για την επικοινωνία μεταξύ των ανθρώπων.

Οι γλώσσες προγραμματισμού έχουν κι αυτές το δικό τους λεξιλόγιο και το δικό τους συντακτικό. Αν θέλουμε να προγραμματίζουμε τον υπολογιστή, για να εκτελεί πιστά τις λειτουργίες που του ζητάμε, πρέπει να μάθουμε κάποια γλώσσα προγραμματισμού. Δυστυχώς οι υπολογιστές δεν έχουν σχεδιαστεί, ώστε να καταλαβαίνουν τη γλώσσα που μιλάμε, δηλαδή τη φυσική γλώσσα. Η πρόοδος, όμως, στον τομέα αυτό είναι σημαντική και πιθανόν στο μέλλον να δίνουμε οδηγίες στον υπολογιστή με την ομιλία.

Γλώσσα Μηχανής

Όπως έχει αναφερθεί στη Β' Γυμνασίου, η λειτουργία των υπολογιστών βασίζεται στην αναπαράσταση μόνο δύο ψηφίων, των «0» και «1». Στα πρώτα βήματα της ιστορίας των υπολογιστών οι άνθρωποι, για να επικοινωνήσουν με τον υπολογιστή,

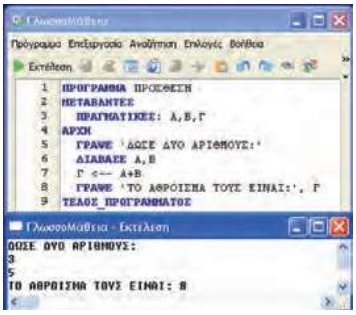
έπρεπε να χρησιμοποιούν μία γλώσσα που είχε ως αλφάβητο το «0» και το «1». Αν ήθελαν λοιπόν να δώσουν μία απλή εντολή στον υπολογιστή, π.χ. να προσθέσει το 3+5 και να εμφανίσει το αποτέλεσμα, έπρεπε να μετατρέψουν όλη την εντολή σε μία γραμμή από 0 και 1. Η γλώσσα αυτή ονομάστηκε **γλώσσα μηχανής**. Η γλώσσα μηχανής είναι αρκετά δύσκολη για να την μάθει κάποιος, γιατί είναι πολύ διαφορετική από τη φυσική μας γλώσσα (Εικόνα 1.3). Επίσης δεν είναι ενιαία σε όλους τους υπολογιστές, μια και κάθε τύπος υπολογιστή (με διαφορετικό επεξεργαστή) έχει τη δική του γλώσσα μηχανής.

```
00000000
00000001
00000010
00000110
00000000
00100000
```

Εικόνα 1.3. Τμήμα Προγράμματος σε γλώσσα μηχανής

Χαρακτηριστικά Γλωσσών Προγραμματισμού

Με την πάροδο των χρόνων οι γλώσσες προγραμματισμού εξελίχθηκαν, ώστε να μοιάζουν όλο και περισσότερο με τη φυσική μας γλώσσα. Στις μέρες μας υπάρχουν διάφορες γλώσσες προγραμματισμού, που χρησιμοποιούνται για την ανάπτυξη γενικών εφαρμογών, ενώ άλλες είναι πιο εξειδικευμένες και χρησιμοποιούνται για πιο ειδικά επιστημονικά προβλήματα (ανώτερων μαθηματικών, μηχανικής, προσομοίωσης πειραμάτων κ.λπ.) και εξειδικευμένες εφαρμογές (προγραμματισμός ιστοσελίδων, διαχείριση εμπορικών δεδομένων κ.λπ.). Μερικές γνωστές γλώσσες προγραμματισμού είναι η Visual Basic, η Logo, η Pascal, η C++, η Java και άλλες.



Εικόνα 1.4. Ο κώδικας για την άθροιση δύο αριθμών στο προγραμματιστικό περιβάλλον Γλωσσομάθεια

Όπως και οι φυσικές γλώσσες, έτσι και κάθε γλώσσα προγραμματισμού έχει ως βασικά χαρακτηριστικά:

- **το αλφάβητο**,
- **το λεξιλόγιο** και
- **το συντακτικό**.

Το **αλφάβητο** μιας γλώσσας προγραμματισμού είναι το σύνολο των χαρακτήρων που χρησιμοποιούνται από τη γλώσσα.

Το **λεξιλόγιο** μιας γλώσσας είναι το σύνολο των λέξεων που αναγνωρίζει η γλώσσα και έχουν συγκεκριμένη και μοναδική σημασία. Στις γλώσσες προγραμματισμού το λεξιλόγιο είναι πολύ περιορισμένο (μερικές δεκάδες λέξεις), ώστε να μπορούμε να το μάθουμε εύκολα.

Το **συντακτικό** μιας γλώσσας προγραμματισμού είναι το σύνολο των κανόνων που πρέπει να ακολουθούμε, για να συνδέουμε λέξεις σε προτάσεις. Σε μια γλώσσα προγραμματισμού η σύνδεση λέξεων δημιουργεί ολοκληρωμένες εντολές προς τον υπολογιστή. Αν δεν ακολουθήσουμε αυστηρά το συντακτικό μιας γλώσσας, είναι αδύνατο για τον υπολογιστή να καταλάβει ποια εντολή του δίνουμε.

Για να μάθουμε λοιπόν μία γλώσσα προγραμματισμού, πρέπει να μάθουμε σταδιακά το λεξιλόγιο που χρησιμοποιεί και το συντακτικό που ακολουθεί, ώστε να γράφουμε κατάλληλα τις εντολές. Κάθε εντολή προκαλεί συγκεκριμένες ενέργειες, αν εκτελεστεί από τον υπολογιστή. Για παράδειγμα, στη γλώσσα Logo η εντολή «**ΤΥΠΩΣΕ "Καλημέρα**» έχει ως αποτέλεσμα την εμφάνιση της λέξης «Καλημέρα» στην οθόνη του υπολογιστή.

Εικόνα 1.5. Το αποτέλεσμα της εντολής «Τύπωσε "Καλημέρα» στο περιβάλλον MWorldsPro

Το ολοκληρωμένο προγραμματιστικό περιβάλλον

Οι σύγχρονες γλώσσες προγραμματισμού μάς προσφέρουν ένα φιλικό περιβάλλον, έτσι ώστε γρήγορα να αναπτύσσουμε τα προγράμματά μας. Ένα περιβάλλον προγραμματισμού αποτελείται από διάφορα εργαλεία που βοηθάνε τον προγραμματιστή να γράψει και να διορθώσει το πρόγραμμά του.

Τα κύρια εργαλεία είναι:

➢ ένας εξειδικευμένος κειμενογράφος, που χρησιμεύει για τη σύνταξη και τη διόρθωση του προγράμματος και

➢ ένα πρόγραμμα-μεταφραστής που μετατρέπει τις οδηγίες μας στη μορφή που τις καταλαβαίνει ο επεξεργαστής, δηλαδή σε μια σειρά από 0 και 1 (Σχήμα 1.3).

Αυτή τη μετατροπή μπορούμε να την παρομοιάσουμε με τη διαδικασία επικοινωνίας μας με ένα κάτοικο της Κίνας. Αν δεν ξέρουμε Κινέζικα και έχουμε έναν Άγγλο μεταφραστή που μιλάει Κινέζικα, μπορούμε να του μιλήσουμε Αγγλικά και αυτός να μεταφράσει αυτό που θέλουμε στα Κινέζικα. Βέβαια μια τέτοια διαδικασία προϋποθέτει ότι ξέρουμε Αγγλικά. Παρόμοια, αν θέλουμε να επικοινωνήσουμε με τον υπολογιστή, πρέπει να μάθουμε μία γλώσσα προγραμματισμού με την οποία μπορεί να γίνει η απαραίτητη μετατροπή των οδηγιών μας σε σειρά από 0 και 1 (γλώσσα μηχανής).

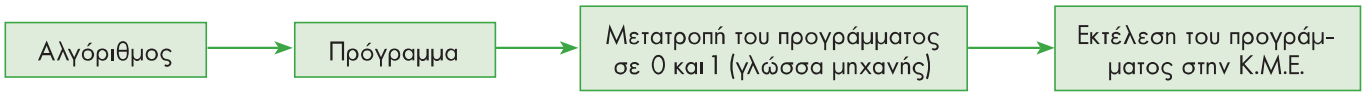
Αν σε κάποια οδηγία έχουμε κάνει λάθος στο αλφάβητο, στο λεξιλόγιο ή στο συντακτικό τότε το πρόγραμμα που μετατρέπει τις οδηγίες μας σε σειρά από 0 και 1 θα μας δώσει ένα κατάλληλο μήνυμα λάθους, ώστε να μας βοηθήσει να διορθώσουμε το λάθος μας. Τα λάθη αυτά ονομάζονται **συντακτικά λάθη**.

Τα προγράμματα που μετατρέπουν τις οδηγίες μας σε 0 και 1 μπορούν να χωριστούν σε δύο κατηγορίες:

- στους μεταγλωττιστές και
- στους διερμηνείς.

Η διαφορά τους είναι ότι οι **μεταγλωττιστές (compilers)** θα ελέγξουν όλο το πρόγραμμα για συντακτικά λάθη και μετά θα το μετατρέψουν όλο σε μια κατάλληλη σειρά από 0 και 1, ώστε να μπορεί να εκτελεστεί από τον επεξεργαστή του υπολογιστή.

Αντίθετα οι **διερμηνείς (interpreters)** ελέγχουν μία οδηγία κάθε φορά, την εκτελούν και μετά ελέγχουν την επόμενη οδηγία. Η γλώσσα προγραμματισμού Logo, που θα δούμε στο επόμενο κεφάλαιο, χρησιμοποιεί διερμηνέα.

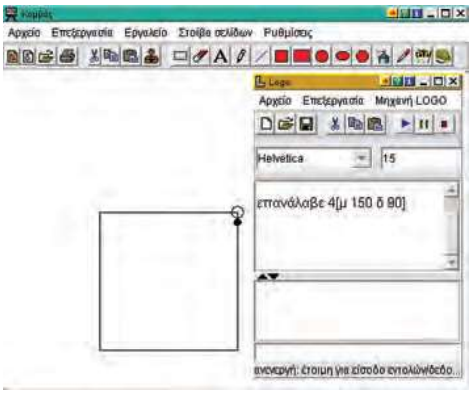


Σχήμα 1.3. Στάδια για την εκτέλεση ενός αλγορίθμου από την Κ.Μ.Ε. του υπολογιστή

Δεν πρέπει να ξεχνάμε ότι ο υπολογιστής εκτελεί πιστά, όποιες συντακτικά ορθές εντολές και αν του δώσουμε. Αν το αποτέλεσμα, που τελικά προκύπτει από την εκτέλεση του προγράμματος, δεν είναι το αναμενόμενο, τότε το πρόβλημα δε βρίσκεται στον τρόπο εκτέλεσης, αλλά στον αλγόριθμο που κατασκευάσαμε για τη λύση του προβλήματός μας. Στην περίπτωση αυτή λέμε ότι έχουμε κάνει ένα **λογικό λάθος** και πρέπει να ελέγξουμε ένα προς ένα τα βήματα-εντολές του αλγορίθμου μας, ώστε να διαπιστώσουμε, αν δίνουμε τις κατάλληλες εντολές με τη σωστή σειρά.

Ένα δεύτερο σημείο που πρέπει να γνωρίζουμε, όταν προγραμματίζουμε, είναι ότι για τον υπολογιστή τίποτα δεν είναι αυτονόητο. Ενώ εμείς οι άνθρωποι έχουμε την ικανότητα να συμπληρώνουμε τις οδηγίες κάποιου με τη λογική και την εμπειρία μας, ο υπολογιστής χρειάζεται να περιγράψουμε με μεγάλη ακρίβεια τις εντολές μας στον υπολογιστή, για να τις εκτελέσει. Αν, για παράδειγμα, του δώσουμε μία εντολή να υπολογίσει ένα άθροισμα, δεν είναι αυτονόητο ότι θα μας εμφανίσει και το αποτέλεσμα. Αν φαίνεται ότι οι υπολογιστές επιλύουν πολύ «έξυπνα» διάφορα προβλήματα, είναι, γιατί κάποιοι άνθρωποι τους προγραμμάτισαν γι' αυτό και όχι γιατί οι μηχανές είναι «έξυπνες». Για να φτιάξουμε λοιπόν ένα καλό πρόγραμμα, πρέπει πρώτα να έχουμε σχεδιάσει έναν καλό αλγόριθμο. Ο ρόλος του αλγορίθμου είναι θεμελιώδης.

Στο κεφάλαιο που ακολουθεί θα ασχοληθούμε με μία εκπαιδευτική γλώσσα με αρκετές δυνατότητες: τη **Logo**. Με τη γλώσσα Logo έχουμε τη δυνατότητα να μάθουμε πολύ γρήγορα πώς να δίνουμε εντολές στον υπολογιστή και να φτιάχνουμε δικά μας προγράμματα. Υπάρχουν πολλές εκδόσεις της γλώσσας Logo. Καθεμία μας προσφέρει ένα φιλικό περιβάλλον προγραμματισμού, για να γράφουμε και να δοκιμάζουμε τα προγράμματά μας. Τη «Berkeley Logo», το «Χελωνόκοσμος» (Εικόνα 1.4) και τη «MSWLogo» μπορείτε να τις βρείτε δωρεάν στο Διαδίκτυο, ενώ τη «Multi-Logo» μπορείτε να τη βρείτε στο λογισμικό Πληροφορικής Γυμνασίου (Παιδαγωγικό Ινστιτούτο 2000).



Εικόνα 1.6. Το περιβάλλον «Χελωνόκοσμος» (Παιδαγωγική Σχεδίαση: Εργαστήριο Εκπαιδευτικής Τεχνολογίας, Φ.Π.Ψ.).

ΕΡΩΤΗΣΕΙΣ

1. Γιατί πρέπει να κατανοούμε καλά ένα πρόβλημα, πριν να το επιλύσουμε;
2. Ποιες διαδικασίες μας βοηθούν στην κατανόηση ενός προβλήματος;
3. Τι είναι ένας αλγόριθμος;
4. Ποιες είναι οι βασικές ιδιότητες ενός Αλγορίθμου;
5. Τι είναι πρόγραμμα;
6. Ποιο είναι το αλφάβητο της γλώσσας μηχανής του υπολογιστή;
7. Ποια είναι τα βασικά χαρακτηριστικά μιας γλώσσας προγραμματισμού;
8. Ποια είναι τα στάδια για την εκτέλεση ενός αλγορίθμου από την Κ.Μ.Ε;