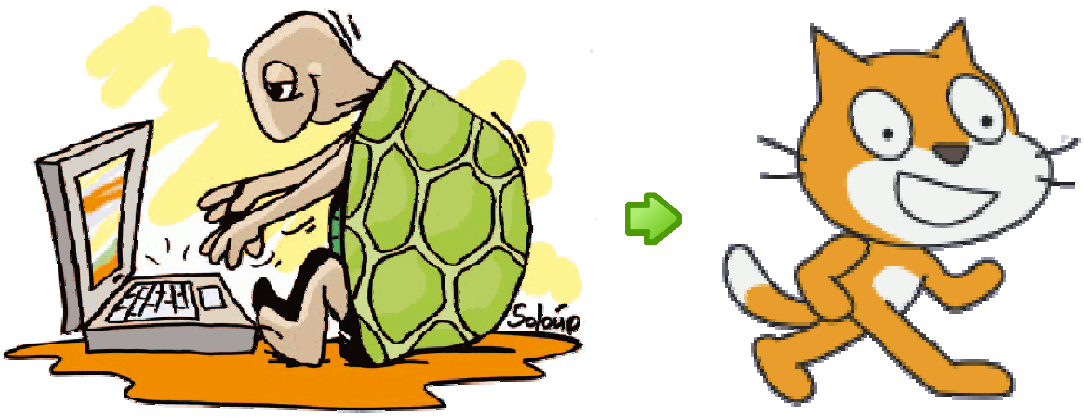
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ, ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗΣ ΚΑΙ ΘΡΗΣΚΕΥΜΑΤΩΝ
ΠΑΙΔΑΓΩΓΙΚΟ ΙΝΣΤΙΤΟΥΤΟ
Αριστείδης Αράπογλου, Χρίστος Μαβόγλου, Ηλίας Οικονομάκος, Κωνσταντίνος Φύτρος

ΠΛΗΡΟΦΟΡΙΚΗ

Γ' ΓΥΜΝΑΣΙΟΥ



Προσαρμογή του σχολικού βιβλίου
στο περιβάλλον προγραμματισμού Scratch

Επιμέλεια προσαρμογής

Στυλιανός-Μαρίνος Χαραλαμπίδης
Διπλ. Ηλεκτρολόγος Μηχανικός και
Μηχανικός Υπολογιστών Α.Π.Θ.
Εκπαιδευτικός ΠΕ19

Ζάκυνθος, Μάιος 2010

Περιεχόμενα

ΚΕΦΑΛΑΙΟ 1

Εισαγωγή στην Έννοια του Αλγορίθμου και στον Προγραμματισμό

Εισαγωγή

Στις προηγούμενες τάξεις αναφέρθηκε ότι ο υπολογιστής μπορεί να μας υποστηρίζει σε διάφορες δραστηριότητες 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;

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

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!

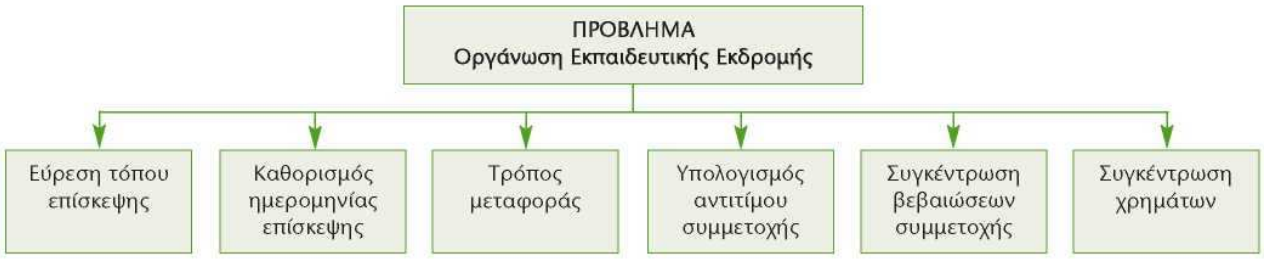
Έτσι, για να επιλύσουμε ένα πρόβλημα πρέπει αρχικά να το **κατανοήσουμε**. Πρέπει δηλαδή να καταλάβουμε καλά το περιεχόμενο του, να διακρίνουμε τα δεδομένα που έχουμε στη διάθεσή μας και τα ζητούμενά του. Είναι σημαντικό, όμως, να προσδιορίσουμε και το «**περιβάλλον**» ή το πλαίσιο μέσα στο οποίο εντάσσεται 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 προβλήματα. Αντιμετωπίζοντας καθένα από τα απλούστερα προβλήματα ξεχωριστά, στο τέλος θα καταφέρουμε να ε-



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

πιλύσουμε 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.

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.

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.

Εισαγωγική Δραστηριότητα

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

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;

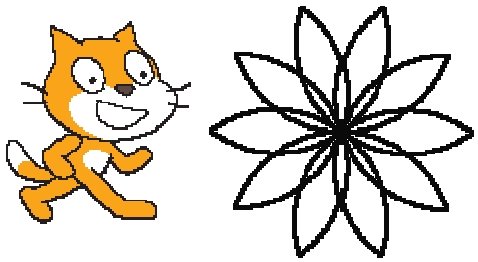
1.3 Ιδιότητες ενός Αλγορίθμου

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.

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).

Αλγόριθμος δημιουργίας ενός λουλουδιού με 10 πέταλα	Το αποτέλεσμα υλοποίησης του Αλγορίθμου
	

Αντίθετα η οδηγία:

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 περιγράψουμε 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 κλπ.

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 μία 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 στο 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 στο 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 αναπαράσταση 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 το 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;

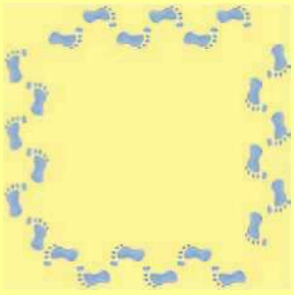
Κατ' αρχάς πρέπει να αναλύσουμε την έννοια «τετράγωνο»:

- 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 που 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**.

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

Οι πύργοι του Ανόι

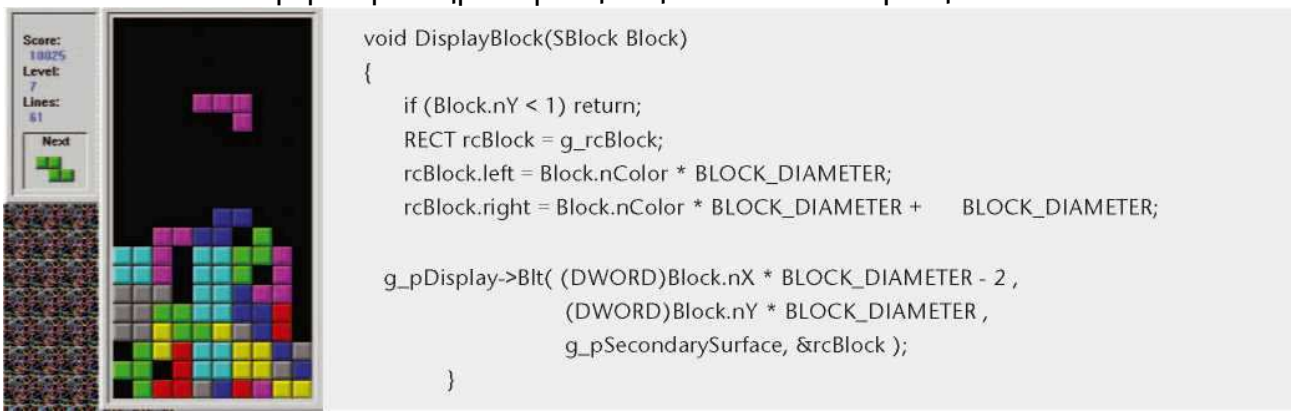
Ο Θρύλος: Σε κάποιους Ινδούς μοναχούς δόθηκε η δοκιμασία να μετακινήσουν 64 εύθραυστους δίσκους από μία τοποθεσία σε μια άλλη, έναν κάθε φορά, αποφεύγοντας την τοποθέτηση ενός μεγαλύτερου δίσκου πάνω σε έναν μικρότερο. Υπήρχε μόνο μια ακόμα ενδιάμεση τοποθεσία, πέρα από τις δύο, που ένας δίσκος μπορούσε 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 στην νέα τοποθεσία, τότε ο ναός τους θα καταρρεύσει και θα μετατραπεί σε σκόνη και ακόμα ο κόσμος θα καταστραφεί

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.



Εικόνα 1.2. Το γνωστό παιχνίδι ΤΕΤΡΙΣ είναι ένα πρόγραμμα το οποίο περιλαμβάνει μία 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 κλπ.) 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 δυο 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

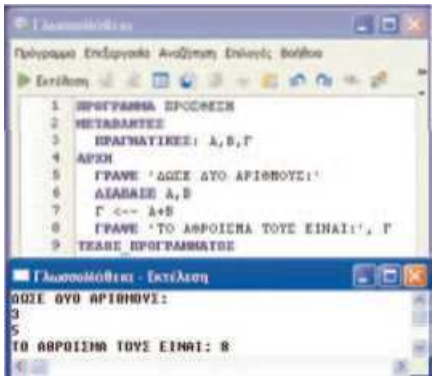
```
00000000
00000001
00000010
00000110
00000000
00100000
```

Εικόνα 1.3. Τμήμα Προγράμματος σε γλώσσα μηχανής.

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.

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 κλπ.) και εξειδικευμένες εφαρμογές (προγραμματισμός ιστοσελίδων, διαχείριση εμπορικών δεδομένων κλπ.). 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.

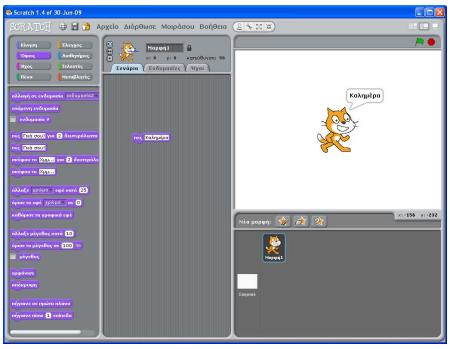
Όπως και οι φυσικές γλώσσες, έτσι και κάθε γλώσσα προγραμματισμού έχει ως βασικά χαρακτηριστικά:

- **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.


Εικόνα 1.5. Το αποτέλεσμα της εντολής *πες "Καλημέρα"* στο περιβάλλον Scratch

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 Scratch η εντολή 

έχει ως αποτέλεσμα την εμφάνιση της λέξης «Καλημέρα» από το γάτο στην οθόνη του υπολογιστή.

Το ολοκληρωμένο προγραμματιστικό περιβάλλον

Οι σύγχρονες γλώσσες προγραμματισμού μάς προσφέρουν ένα φιλικό περιβάλλον, έτσι ώστε γρήγορα να αναπτύσσουμε τα προγράμματα 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 κατα-

λαβαίνει 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).

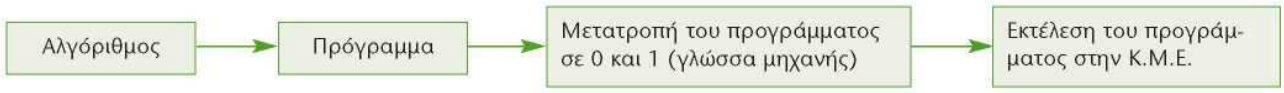
Αν σε κάποια οδηγία έχουμε κάνει λάθος στο αλφάβητο, στο λεξιλόγιο ή στο συντακτικό τότε το πρόγραμμα που μετατρέπει τις οδηγίες μας σε σειρά από 0 και 1 θα μας δώσει ένα κατάλληλο μήνυμα λάθους, ώστε να μας βοηθήσει να διορθώσουμε το λάθος μας. Τα λάθη αυτά ονομάζονται **συντακτικά λάθη**.

Τα προγράμματα που μετατρέπουν τις οδηγίες μας σε 0 και 1 μπορούν να χωριστούν σε δυο 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 Scratch, 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, 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 ένα εκπαιδευτικό περιβάλλον προγραμματισμού με αρκετές δυνατότητες: το Scratch. Με το Scratch έχουμε τη δυνατότητα να δημιουργούμε διαδραστικές ιστορίες, παιχνίδια και κινούμενα σχέδια. Το Scratch αναπτύσσεται από το Lifelong Kindergarten Group στο MIT Media Lab και διανέμεται δωρεάν από το δικτυακό τόπο http://scratch.mit.edu/ .

Ερωτήσεις

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;

ΚΕΦΑΛΑΙΟ 2

Ο Προγραμματισμός στην Πράξη

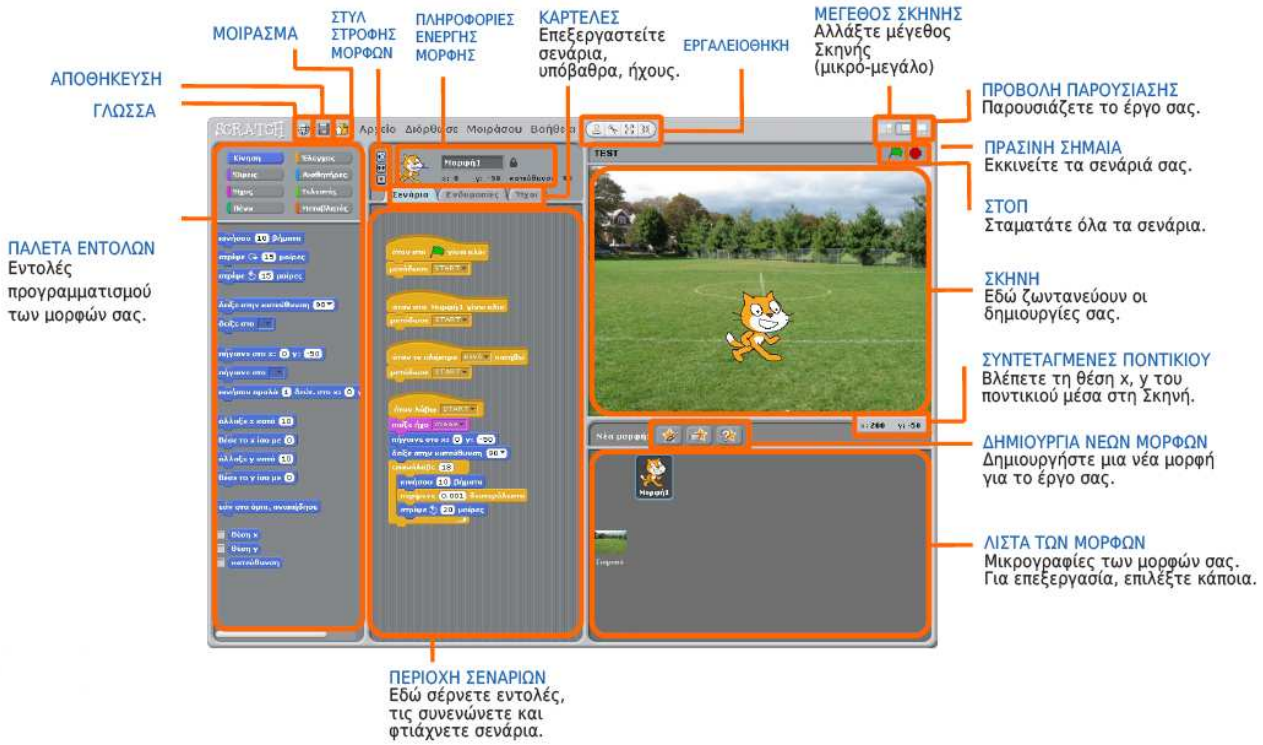
Εισαγωγή

Το Scratch είναι μια νέα γλώσσα προγραμματισμού που μας δίνει τη δυνατότητα να δημιουργήσουμε διαδραστικές ιστορίες, παιχνίδια και κινούμενα σχέδια, καθώς και να τις μοιραστούμε με άλλους στο διαδίκτυο. Στον ιστοχώρο του Scratch υπάρχουν και πολλές άλλες πηγές μάθησης και υποστήριξης όπως: εκπαιδευτικά βίντεο, κάρτες Scratch και συχνές ερωτήσεις (*http://info.scratch.mit.edu/Support/*). Ο παρών Οδηγός Χρήσης αφορά την έκδοση 1.4 (Ιούλιος 2009) του Scratch. Νεότερες εκδόσεις διατίθενται στη διεύθυνση: *http://info.scratch.mit.edu/Support/* .

Τα έργα (project) στο Scratch οικοδομούνται από αντικείμενα που λέγονται **μορφές** (sprite). Μπορούμε να αλλάξουμε την εμφάνιση μιας μορφής δίνοντάς της μια διαφορετική **ενδυμασία** (costume). Μπορούμε να την κάνουμε να μοιάζει με άνθρωπο, με τρένο, με πεταλούδα ή με οτιδήποτε άλλο. Για ενδυμασία μπορούμε να χρησιμοποιήσουμε οποιαδήποτε εικόνα: μπορούμε να δημιουργήσουμε μία εικόνα στον Επεξεργαστή Ζωγραφικής, να εισάγουμε μια από τον σκληρό μας δίσκο ή μία από το διαδίκτυο. Μπορούμε να δώσουμε οδηγίες σε μια μορφή ώστε να κινηθεί, να παίξει μουσική ή να αλληλεπιδράσει με άλλες μορφές. Για να πούμε στη μορφή τι να κάνει, συνενώνουμε εικονικές **εντολές** (που μοιάζουν με τουβλάκια – block) μεταξύ τους σε στήλες που ονομάζονται **σενάρια ενεργειών** (κώδικες, script). Όταν κάνουμε κλικ σε ένα σενάριο, το Scratch «τρέχει» τις εντολές από την κορυφή του σεναρίου έως τον πάτο.

2.1 Το περιβάλλον προγραμματισμού Scratch

Την πρώτη φορά που παρατηρούμε το περιβάλλον προγραμματισμού του Scratch βλέπουμε ότι η οθόνη χωρίζεται σε τέσσερεις περιοχές: *Σκηνή*, *Περιοχή σεναρίων*, *Παλέτα εντολών* και *Λίστα των μορφών* (Εικόνα 2.1).



Εικόνα 2.1. Το περιβάλλον του Scratch

Η **Σκηνή** είναι ο χώρος όπου ζωντανεύουν οι ιστορίες μας, τα παιχνίδια και τα κινούμενα σχέδια. Οι μορφές κινούνται και αλληλεπιδρούν μεταξύ τους επάνω στη Σκηνή. Η Σκηνή έχει μήκος 480 μονάδες και ύψος 360 μονάδες. Είναι χωρισμένη σε άξονες x και y. Το κέντρο της Σκηνής έχει συντεταγμένες x:0 και y:0.

Η **Λίστα των μορφών** παρουσιάζει μικρογραφίες όλων των μορφών του έργου. Το όνομα της κάθε μορφής εμφανίζεται κάτω από τη μικρογραφία της. Για να δούμε ή να επεξεργαστούμε τα σενάρια, τις ενδυμασίες και τους ήχους μιας μορφής, κάνουμε κλικ πάνω στη μικρογραφία της στη Λίστα των μορφών ή κάνουμε διπλό κλικ πάνω στην ίδια τη μορφή μέσα στη Σκηνή. (Η επιλεγμένη μορφή είναι μαρκαρισμένη με μπλε περίγραμμα μέσα στη Λίστα των μορφών)

Για να προγραμματίσουμε μια μορφή, σέρνουμε εντολές (τουβλάκια) από την **Παλέτα εντολών** προς την **Περιοχή σεναρίων**. Για να εκτελεστεί μια εντολή, κάνουμε κλικ επάνω της. Δημιουργούμε σενάρια ενεργειών (προγράμματα) συνενώνοντας εντολές μεταξύ τους σε στήλες και κάνουμε κλικ οπουδήποτε στη στήλη για να εκτελεστεί ολόκληρο το σενάριο, από την αρχή ως το τέλος. Για να καταλάβουμε τι ακριβώς κάνει μια εντολή, κάνουμε δεξί κλικ πάνω της και επιλέγουμε τη βοήθεια από το αναδυόμενο μενού. Όταν σέρνουμε μία εντολή μέσα στη Περιοχή σεναρίων, μία λευκή υπογράμμιση υποδεικνύει πού μπορούμε να την αφήσουμε ώστε να δημιουργήσει μια σωστή ένωση με άλλη εντολή.

2.2 Οι πρώτες εντολές

Εντολή εμφάνισης (εξόδου) και αριθμητικές πράξεις

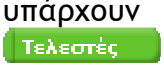
Μια βασική βοήθεια που μπορεί να μας προσφέρει ο υπολογιστής είναι η εκτέλεση σύνθετων αριθμητικών πράξεων. Αν ανατρέξουμε στην ιστορία των υπολογιστών, θα διαπιστώσουμε ότι οι πρώτοι ηλεκτρονικοί υπολογιστές στη δεκαετία του '40 είχαν κατασκευαστεί, για να βοηθήσουν στην εκτέλεση διάφορων υπολογισμών. Όλες λοιπόν οι διαδεδομένες γλώσσες προγραμματισμού έχουν σχεδιαστεί, ώστε να μπορούμε να εκτελούμε αριθμητικές πράξεις.

Όπως αναφέρθηκε στο πρόβλημα της εκδρομής (1ο Κεφάλαιο της Α' Γυμνασίου), τα δύο παιδιά αφού κατανόησαν το πρόβλημα της συγκέντρωσης χρημάτων για την εκπαιδευτική εκδρομή, συγκέντρωσαν τα απαραίτητα δεδομένα και βρήκαν ως λύση ότι έπρεπε να διαιρέσουν το κόστος ενοικίασης του λεωφορείου με το πλήθος των μαθητών που επρόκειτο να συμμετάσχουν στην εκδρομή. Συγκεκριμένα, έπρεπε να κάνουν τη διαίρεση 200 : 25 = ;. Μπορούμε να δώσουμε μια εντολή με τη γλώσσα προγραμματισμού Scratch και να μας εμφανίσει το αποτέλεσμα της διαίρεσης;

Η κατάλληλη εντολή είναι η 

. Η εντολή αυτή εκτελεί την πράξη 200 : 25 και εμφανίζει το αποτέλεσμα στη Σκηνή. Η εντολή 

είναι μία εντολή **εξόδου**, καθώς έχει ως αποτέλεσμα την εμφάνιση ενός αριθμού ή ενός μηνύματος στην οθόνη του υπολογιστή.

Ο υπολογιστής μπορεί να κάνει όλες τις αριθμητικές πράξεις. Για τα σύμβολα των πράξεων χρησιμοποιούμε τα σύμβολα που υπάρχουν στην Παλέτα εντολών 

(στα αριστερά της Επιφάνειας εργασίας του Scratch): 

για πρόσθεση, 

για αφαίρεση, 

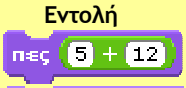
για πολλαπλασιασμό και 

για διαίρεση.

Ας δοκιμάσουμε τώρα λίγο πιο σύνθετες πράξεις. Δώστε στον υπολογιστή τις εντολές:

Εισαγωγική Δραστηριότητα

Δοκιμάστε τις παρακάτω εντολές και συμπληρώστε τα αποτελέσματα στον πίνακα. Στη συνέχεια προσπαθήστε να κάνετε διάφορους υπολογισμούς δοκιμάζοντας διάφορα νούμερα.

Εντολή	Αποτέλεσμα
	
	
	
	

α. πες (12 / 2) * 3 

β. πες 12 / (2 * 3) 

1. Ποιο είναι το αποτέλεσμα στις περιπτώσεις α) _____ και β) _____ ;
2. Με ποια σειρά εκτελέστηκαν οι πράξεις στις δύο αυτές εντολές;

Η Εμφάνιση Μηνυμάτων

Η εντολή 

επιτρέπει, εκτός από αριθμούς, να εμφανίζεται στη Σκηνή και κάποιο μήνυμα. Αν, για παράδειγμα, θέλουμε να εμφανίσουμε το όνομα μας, τότε μπορούμε να γράψουμε 

.

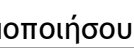
Αν μετά την εντολή 

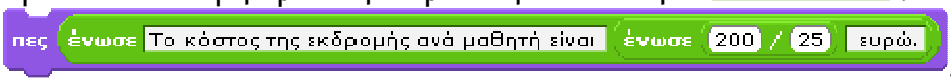
γράψουμε την αριθμητική έκφραση χωρίς να χρησιμοποιήσουμε τελεστές, τότε η εκτέλεση της εντολής θα έχει ως αποτέλεσμα την εμφάνιση της έκφρασης όπως εμφανίζεται μέσα στο πλαίσιο. Η εντολή 

εμφανίζει το «2+3» και όχι το αποτέλεσμα της πράξης, γιατί ο υπολογιστής εκλαμβάνει το 2+3 ως μία λέξη και όχι ως αριθμούς με τους οποίους πρέπει να κάνει πρόσθεση. Αν θέλουμε να εμφανίσουμε το μήνυ-

μα «Το όνομα μου είναι Στέλιος», τότε πρέπει να γράψουμε 

. Ο υπολογιστής θα εμφανίσει όλες τις λέξεις που περικλείονται στο πλαίσιο. Πειραματιστείτε εμφανίζοντας τα δικά σας μηνύματα στον υπολογιστή.

Πώς μπορούμε, όμως, να εμφανίζουμε μηνύματα μαζί με τα αποτελέσματα αριθμητικών πράξεων; Για παράδειγμα, ποια εντολή θα δίναμε, για να εμφανιστεί στον Κωστή και στη Χρύσα το μήνυμα: «Το κόστος της εκδρομής ανά μαθητή είναι 8 €», όπου το 8 είναι το αποτέλεσμα της πράξης 200 / 25; Αν θέλουμε να ενώσουμε δυο μηνύματα μεταξύ τους, τότε πρέπει να χρησιμοποιήσουμε την εντολή 

. Δοκιμάστε την εντολή 

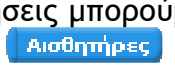
. Τι εμφανίζεται στην οθόνη;

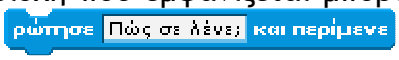
2.3 Συνομιλία με τον υπολογιστή. Περισσότερα για τις εντολές εισόδου-εξόδου

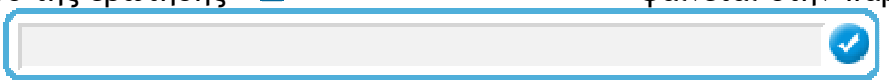
Στις προηγούμενες παραγράφους είχαμε την ευκαιρία να γνωρίσουμε την εντολή εξόδου 

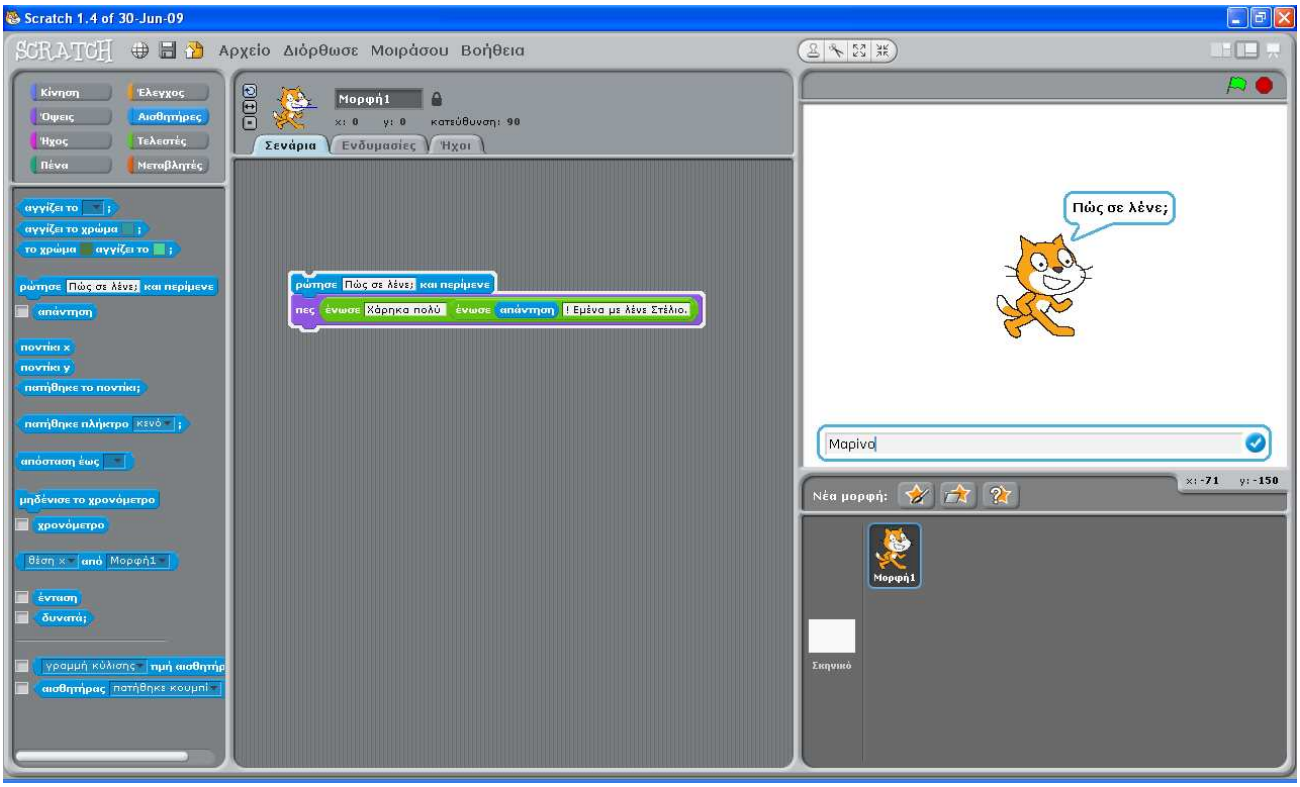
, με την οποία εμφανίζουμε δεδομένα στη Σκηνή της Επιφάνειας εργασίας του Scratch. Το περιβάλλον προγραμματισμού Scratch μας δίνει τη δυνατότητα να πραγματοποιούμε διαλόγους χρησιμοποιώντας ερωταποκρίσεις.

Ερωτήσεις μπορούμε να κάνουμε με την εντολή 

από την Παλέτα εντολών 

και στην κενή περιοχή που εμφανίζεται μπορούμε να δώσουμε μιαν απάντηση. Το παράθυρο της ερώτησης 

φαίνεται στην παρακάτω εικόνα. Το πλαίσιο  χρησιμεύει, για να πληκτρολογήσουμε την απάντηση μας.

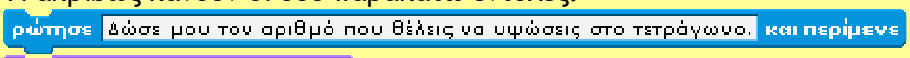


Η εντολή  είναι μια εντολή **εισόδου**, γιατί μας επιτρέπει να δώσουμε μία τιμή (μια λέξη, ένα σύνολο λέξεων, δηλαδή μια λίστα, ή έναν αριθμό) στον υπολογιστή, ώστε στη συνέχεια να την επεξεργαστεί ή να την εμφανίσει στην οθόνη. Αν θέλουμε να χρησιμοποιήσουμε ξανά την τιμή που δίνουμε στο πλαίσιο της ερώτησης, αυτή αποθηκεύεται προσωρινά και μπορούμε να την ανακτήσουμε χρησιμοποιώντας τη λέξη  , όπως στο επόμενο παράδειγμα.

Για να καταλάβετε καλύτερα τη χρήση των εντολών  και  , φτιάξτε τις δικές σας συνομιλίες.

<u>Δραστηριότητα</u>: Ας πειραματιστούμε λίγο και με τους αριθμούς

1. Τι ακριβώς κάνουν οι δυο παρακάτω εντολές:



2. Ποιο είναι το αποτέλεσμα της εκτέλεσης των παραπάνω εντολών, αν δώσουμε την τιμή 3456; Δοκιμάστε το στον υπολογιστή και στη συνέχεια δώστε και άλλες τιμές πατώντας κάθε φορά Enter μετά την πληκτρολόγηση του αριθμού.
3. Πώς μπορούν να τροποποιηθούν οι παραπάνω εντολές, ώστε να υπολογίζουμε τον κύβο ενός αριθμού;

2.4 Η γλώσσα Scratch και ο σχεδιασμός γεωμετρικών σχημάτων

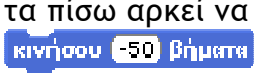
Κάνοντας τις πρώτες δοκιμές με τη γάτα

Οι **μορφές** είναι ίσως το πιο βασικό χαρακτηριστικό της γλώσσας Scratch. Για να δημιουργήσουμε μια μορφή στη *Σκηνή*, χρησιμοποιούμε τα εικονίδια με την ετικέτα «Νέα Μορφή»:

Πολλές εντολές στη γλώσσα Scratch μετακινούν και χειρίζονται τις μορφές στη Σκηνή. Η μορφή της γάτας που εμφανίζεται είναι αυτή που εικονίζεται στο πλάι:

Το ίχνος που αφήνει η γάτα, με την κατάλληλη μετακίνηση της, μας επιτρέπει να δημιουργήσουμε διάφορα σχέδια και γεωμετρικά σχήματα. Οι βασικές εντολές που μπορούμε να δώσουμε στη γάτα, ώστε να την κατευθύνουμε, είναι:

- 

: Με την εκτέλεση της εντολής αυτής η γάτα προχωράει μπροστά τόσα βήματα όσα έχουμε ορίσει (στο συγκεκριμένο παράδειγμα 10). Για να κινηθεί η γάτα προς τα πίσω αρκεί να εισάγουμε έναν αρνητικό αριθμό, π.χ.


- 

: Η γάτα στρίβει προς τα δεξιά τόσες μοίρες όσες έχουμε ορίσει (στο παράδειγμα 15).
- 

: Η γάτα στρίβει προς τα αριστερά τόσες μοίρες όσες έχουμε ορίσει (στην περίπτωσή μας 60).
- 

: Δίνει εντολή στη γάτα να αφήνει ίχνος από κάθε σημείο της οθόνης που περνάει. Αν δεν έχουμε δώσει στην αρχή αυτή την εντολή, η γάτα μετακινείται με τις κα-

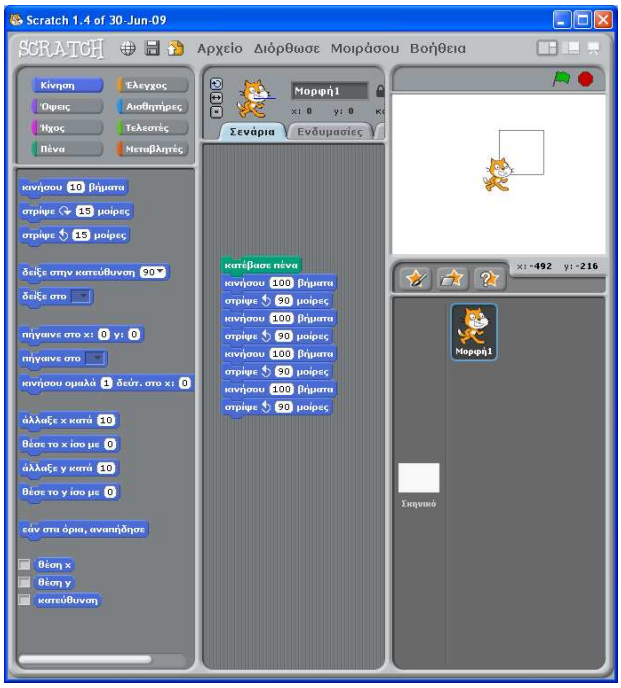
Εισαγωγική Δραστηριότητα

Τοποθετήστε ένα ψάρι στην **Σκηνή** επιλέγοντας το εικονίδιο *«Διάλεξε νέα μορφή από αρχείο»*. Πειραματιστείτε μετακινώντας το ψάρι στη Σκηνή και δοκιμάστε τις διπλανές εντολές στην *Περιοχή Σεναρίων*. Στη συνέχεια προσπαθήστε να δημιουργήσετε ένα ευθύγραμμο τμήμα μήκους 100 βημάτων.

τάλληλες εντολές στην οθόνη, χωρίς να σχεδιάζει τίποτε.

-  : Δίνει εντολή στη γάτα να σταματήσει να αφήνει ίχνος καθώς προχωράει.
-  : Σβήνει τα σχέδια που έχουμε δημιουργήσει στη σκηνή
-  : Μεταφέρει τη γάτα στο κέντρο της σκηνής
-  : Στρέφει τη γάτα προς την αρχική κατεύθυνση (προς τα δεξιά).

Ας θυμηθούμε λίγο τον αλγόριθμο του τετραγώνου που παρουσιάσαμε στο κεφάλαιο των αλγορίθμων. Ο αλγόριθμος αυτός περιέγραφε τα βήματα που πρέπει να ακολουθήσει ένα μικρό παιδί, ώστε να φτιάξει ένα τετράγωνο στην άμμο. Με μία μικρή παραλλαγή μπορούμε να υλοποιήσουμε τον αλγόριθμο αυτό, για να κατασκευάσουμε ένα τετράγωνο με μήκος πλευράς 100 βήματα, δίνοντας εντολές στη γάτα.

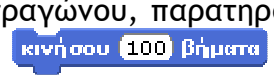


Εικόνα 2.3. Δημιουργία ενός τετραγώνου με τη βοήθεια της γάτας.

Όπως βλέπουμε και στη διπλανή εικόνα ο συνδυασμός των εντολών:

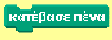
δημιουργεί ένα τετράγωνο στην οθόνη μας (η τελευταία εντολή απλά επαναφέρει τη γάτα στην αρχική κατεύθυνση).

Δομή Επανάληψης

Αν μελετήσουμε καλύτερα το παραπάνω πρόγραμμα του τετραγώνου, παρατηρούμε ότι οι εντολές  και  επαναλήφθηκαν **τέσσερις φορές** με την ίδια σειρά. Θα μπορούσαμε να έχουμε το ίδιο αποτέλεσμα ομαδοποιώντας τις δυο εντολές και δίνοντας μια εντολή που να τις επαναλαμβάνει τέσσερις φορές. Η εντολή αυτή είναι: 

Με την εντολή αυτή μπορούμε να κατασκευάσουμε το ίδιο τετράγωνο ως εξής:

Ερώτηση

Ποιο αποτέλεσμα θα προέκυπτε, αν δε γράφαμε την εντολή  στην αρχή;

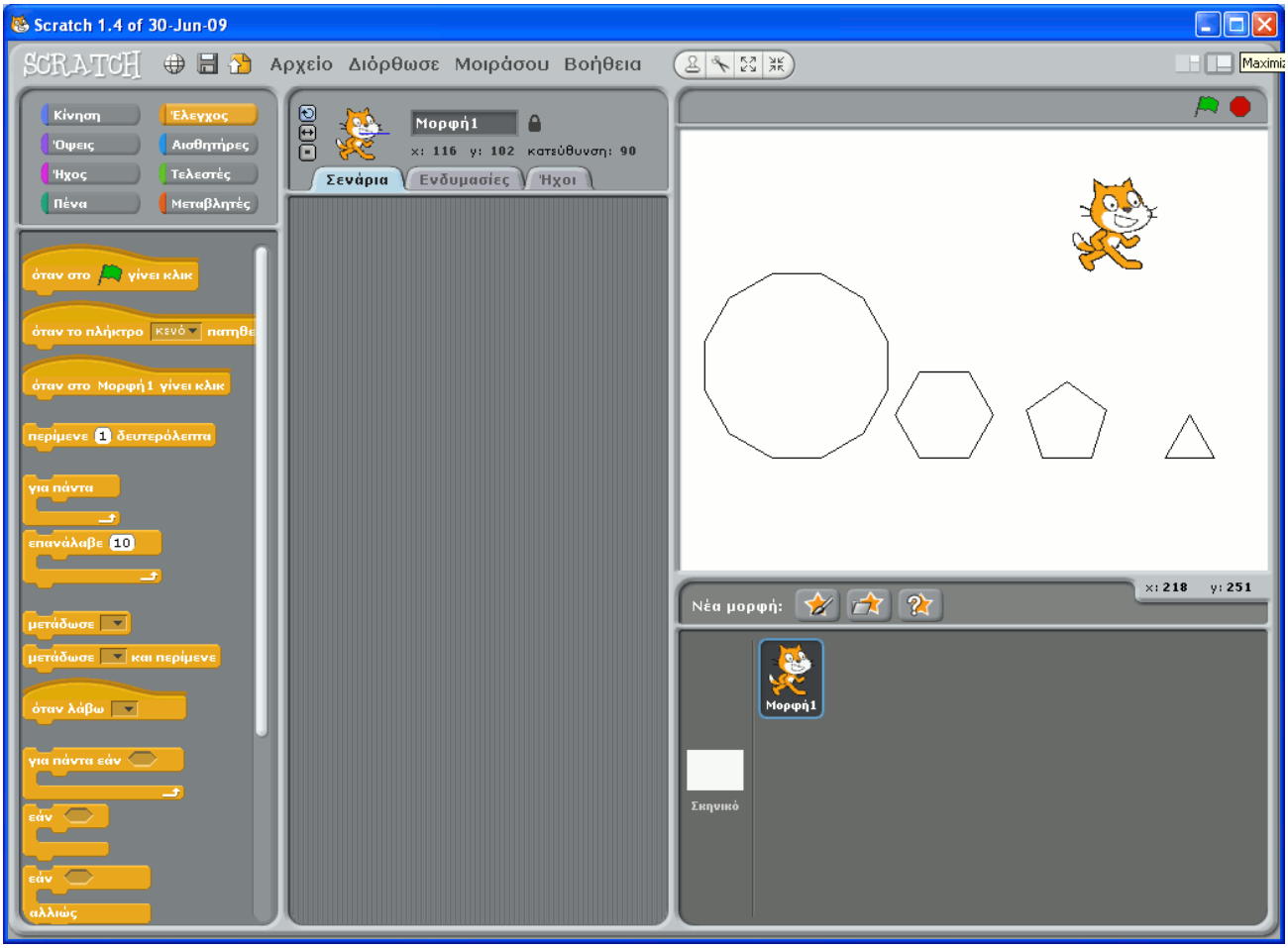
Η **δομή της επανάληψης** είναι πολύ χρήσιμη στον προγραμματισμό. Χρησιμοποιώντας τις εντολές επανάληψης ο υπολογιστής μπορεί να εκτελεί τις επαναλαμβανόμενες ενέργειες (υπολογισμούς, εμφανίσεις στην οθόνη κλπ.) και μάλιστα πολύ πιο γρήγορα από εμάς.

Δραστηριότητες:

1. Να γράψετε την κατάλληλη εντολή, ώστε να εμφανιστεί το όνομά σας 200 φορές στην οθόνη του υπολογιστή:
2. Στην εντολή που χρησιμοποιήσαμε, για να σχεδιάσουμε ένα τετράγωνο:

συνολικά στο σχήμα μας κάναμε στροφή 360 μοιρών σε 4 βήματα. Δηλαδή σε κάθε βήμα στρίψαμε 360:4=90 μοίρες.
Να δώσετε τις κατάλληλες εντολές στη γάτα, ώστε να σχεδιάσει ένα ισόπλευρο τρίγωνο, ένα πεντάγωνο, ένα εξάγωνο ή ένα δωδεκάγωνο, όπως τα σχήματα της Εικόνας 2.4.

3. Με τι μοιάζει το σχήμα που δημιουργεί η επόμενη ομάδα εντολών;



Εικόνα 2.4. Δημιουργία γεωμετρικών σχημάτων με τη γάτα της γλώσσας Scratch.

2.5 Δημιουργώντας Σενάρια

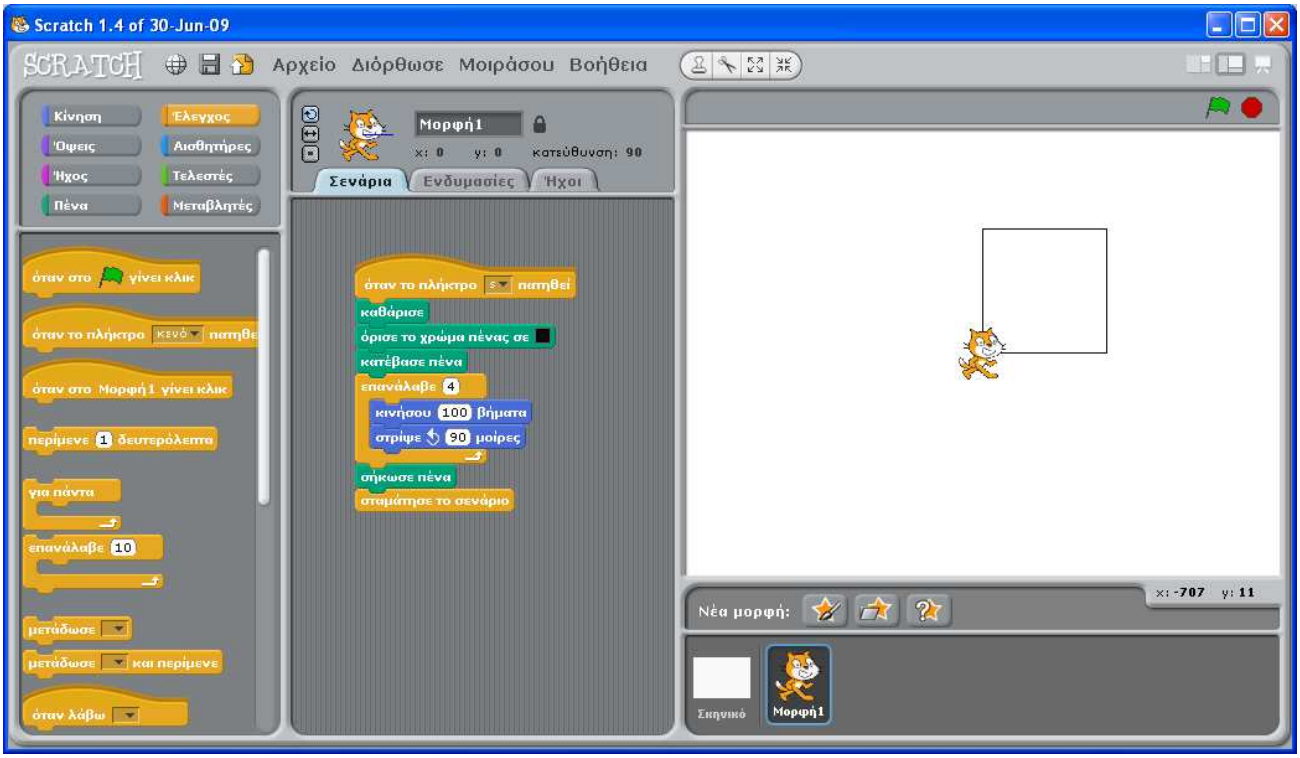
Εντολές που επιθυμούμε να εκτελεστούν με το πάτημα ενός πλήκτρου μπορούν να ομαδοποιηθούν. Η ομαδοποίηση αυτή των εντολών καλείται **Σενάριο**. Η εκτέλεση ενός σεναρίου μπορεί να χρησιμεύσει στον υλοποίηση μίας πιο σύνθετης εφαρμογής, στη βελτίωση της επικοινωνίας με το χρήστη ή στη διαδραστικότητα μεταξύ δύο ή περισσοτέρων μορφών. Το πλεονέκτημα του σεναρίου είναι ότι μπορούμε να το εκτελέσουμε με το πάτημα ενός πλήκτρου όποτε το χρειαστούμε.

Για τη δημιουργία ενός σεναρίου εισάγουμε ως πρώτη την εντολή 

από την ομάδα 

επιλέγοντας το πλήκτρο που μας εξυπηρετεί:

Στο επόμενο παράδειγμα (Εικόνα 2.5) έχουμε γράψει ένα παράδειγμα ενός σεναρίου, με το οποίο κάθε φορά που πιέζεται το πλήκτρο s, η γάτα σχεδιάζει ένα τετράγωνο πλευράς 100.

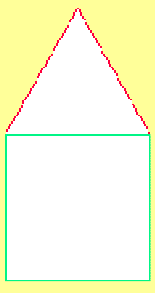


Εικόνα 2.5. Το σενάριο *s* έχει δημιουργήσει ένα τετράγωνο

Κάθε φορά που πιέζουμε το γράμμα *s* στο πληκτρολόγιο, θα σχηματίζεται ένα τετράγωνο. Ουσιαστικά με αυτόν τον τρόπο η γλώσσα Scratch μας επιτρέπει να δημιουργούμε τα δικά μας σενάρια-εντολές.

1η Δραστηριότητα

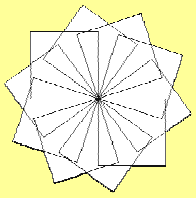
Δημιουργήστε δύο σενάρια: ένα που θα σχεδιάζει ένα τρίγωνο με το πάτημα του πλήκτρου *t* και ένα που θα σχεδιάζει ένα τετράγωνο με το πάτημα του πλήκτρου *s*. Χρησιμοποιώντας τα νέα σενάρια που μόλις δημιουργήσατε, προσπαθήστε να δημιουργήσετε ένα σπίτι, όπως το διπλανό σχήμα.



Στη συνέχεια δημιουργήστε ένα σενάριο που θα σχεδιάζει με το πάτημα του πλήκτρου *v* πολλά σπίτια σε τυχαίες θέσεις.

2η Δραστηριότητα

Προσπαθήστε να περιγράψετε την πορεία της γάτας ακολουθώντας τις παρακάτω εντολές. Το αποτέλεσμα των εντολών φαίνεται στην παρακάτω εικόνα. Αφού έχετε κατανοήσει τη σημασία των αριθμών 10 και 36 στις εντολές, προσπαθήστε να τους αλλάξετε δημιουργώντας τα δικά σας σχήματα.

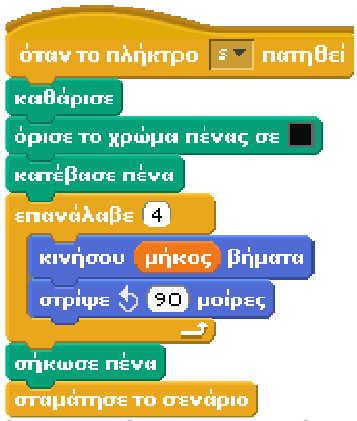


2.6 Μεταβλητές

Σύμφωνα με το παραπάνω σενάριο (Εικόνα 2.5), όταν θέλουμε να δημιουργούμε τετράγωνα με διαφορετικό μήκος πλευράς, πρέπει να επεμβαίνουμε κάθε φορά στην εντολή  και να αλλάζουμε το μήκος της πλευράς. Έτσι, αν θέλουμε να σχεδιάσουμε ένα τετράγωνο πλευράς 50 βημάτων, θα δώσουμε τις εντολές:

ενώ, αν θέλουμε ένα τετράγωνο πλευράς 150, θα δώσουμε τις εντολές:

Μπορούμε να χρησιμοποιούμε το ίδιο πάντα σενάριο για το σχεδιασμό τετραγώνων διαφορετικών πλευρών; Η απάντηση είναι καταφατική. Το σενάριο μπορούμε να το τροποποιήσουμε ως εξής:



Αφού έχουμε γράψει το παραπάνω σενάριο, μπορούμε να σχεδιάσουμε ένα τετράγωνο πλευράς 50 δίνοντας, στο ίδιο ή σε κάποιο άλλο σενάριο, την εντολή 

Δηλαδή, πριν την εκτέλεση του σεναρίου s ορίζουμε το επιθυμητό μήκος της πλευράς του τετραγώνου. Η τιμή 50 αποθηκεύεται προσωρινά στο  και η εντολή  μετακινεί τη γάτα μπροστά κατά 50 βήματα. Κάθε φορά που εκτελούμε το σενάριο s, στο  αποθηκεύεται προσωρινά μία διαφορετική τιμή. Το  ονομάζεται **μεταβλητή**.

Ερώτηση

Πώς θα ενεργοποιήσουμε το σενάριο *s*, ώστε να σχεδιάσει ένα τετράγωνο με πλευρά μήκους 80;

Το περιεχόμενο μιας **μεταβλητής** μπορεί να μεταβάλλεται κατά την εκτέλεση ενός προγράμματος. Μια μεταβλητή αντιστοιχεί σε μία θέση της μνήμης του υπολογιστή και γίνεται αναφορά σε αυτή με το όνομα που της δίνουμε εμείς. Μία θέση μνήμης μπορεί να έχει **μόνο μία τιμή** κάθε φορά, αλλά μπορούμε να την αλλάζουμε, όποτε είναι απαραίτητο, με μία άλλη τιμή.

Φανταστείτε τη μεταβλητή σα μια φωλιά, η οποία χωράει μόνο ένα αβγό. Όπως μπορούμε να αντικαθιστούμε το αυγό στη φωλιά με ένα άλλο, έτσι μπορούμε να αντικαθιστούμε την τιμή μιας μεταβλητής με μία άλλη τιμή.

Η προηγουμένη τιμή της μεταβλητής, όμως, χάνεται και δεν μπορούμε να τη χρησιμοποιήσουμε ξανά. Ωστόσο, μπορούμε να χρησιμοποιήσουμε περισσότερες μεταβλητές, για να αποθηκεύσουμε διαφορετικές τιμές.

Στη γλώσσα Scratch, για να αναφερθούμε στην τιμή της μεταβλητής, χρησιμοποιούμε το «τουβλάκι» με το όνομα της μεταβλητής, π.χ. μήκος. Για να δημιουργήσουμε μία νέα μεταβλητή, κάνουμε κλικ στην εντολή Δημιούργησε μια μεταβλητή από την ομάδα Μεταβλητές. Για να δώσουμε (εκχωρήσουμε) τιμή σε μία μεταβλητή, μπορούμε να χρησιμοποιήσουμε την εντολή όρισε το μήκος σε 50, όπως αναφέρθηκε και στο παραπάνω παράδειγμα, επιλέγοντας με το βελάκι το όνομα της μεταβλητής που μας ενδιαφέρει και συμπληρώνοντας στο λευκό πλαίσιο την τιμή που θέλουμε να αποθηκευτεί. Αν θέλουμε, για παράδειγμα, να δώσουμε στη μεταβλητή με όνομα x την τιμή 2 γράφουμε όρισε το x σε 2, ενώ αν θέλουμε να δώσουμε την τιμή Γάτα γράφουμε όρισε το x σε Γάτα.

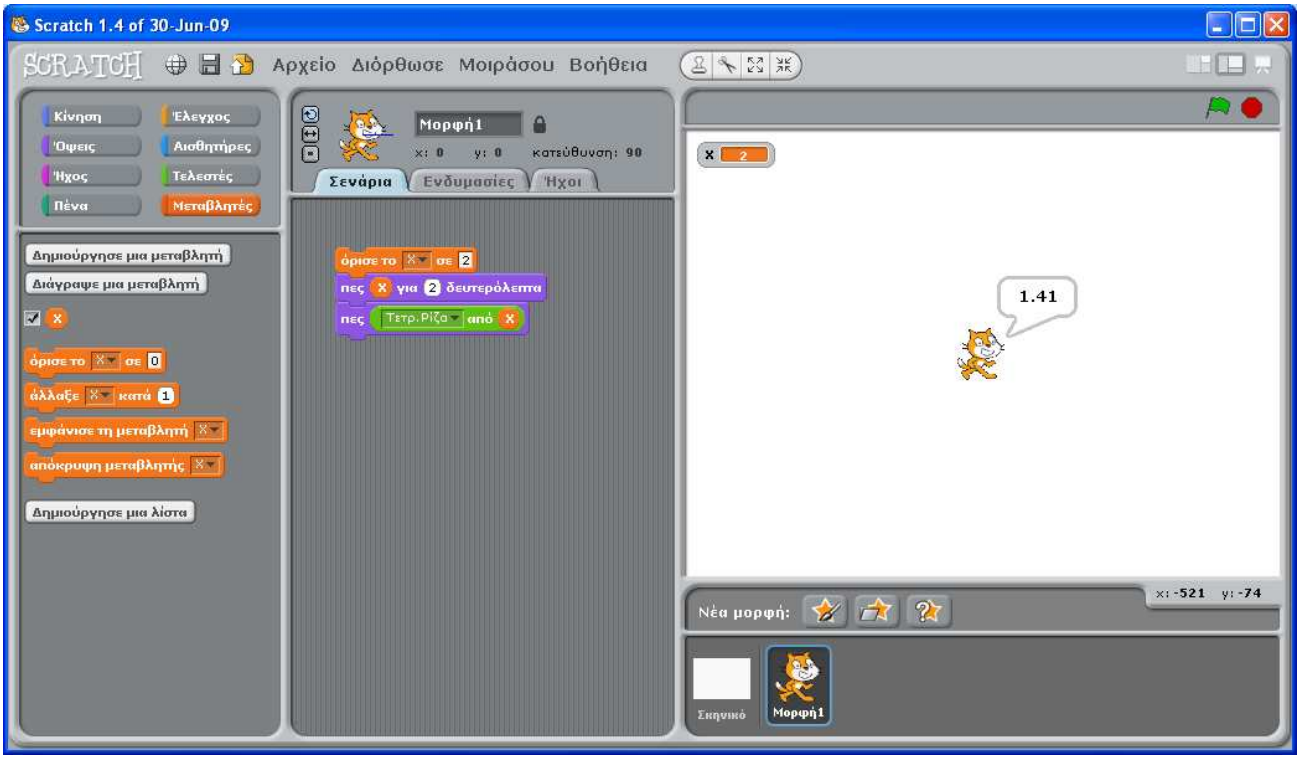
Πολλές φορές κάνουμε το λάθος και λέμε ότι η τιμή του x είναι ίση με 2.

Ποια εντολή θα χρησιμοποιήσουμε, για να εμφανίσουμε την τιμή που περιέχει η μεταβλητή x;

Πώς μπορούμε να υπολογίσουμε στη συνέχεια την τετραγωνική ρίζα του x και να εμφανιστεί στην οθόνη;

Η απάντηση φαίνεται στην παρακάτω Εικόνα 2.6.

Η μεταβλητή στον προγραμματισμό δεν έχει την ίδια έννοια που έχει η μεταβλητή στα Μαθηματικά. Στον προγραμματισμό σε μία μεταβλητή Χ τοποθετούμε (εκχωρούμε) μία τιμή, δηλαδή, στη θέση μνήμης που αντιστοιχεί στη μεταβλητή Χ αποθηκεύουμε προσωρινά μία τιμή.



Εικόνα 2.6. Εκχώρηση τιμής σε μεταβλητή και εμφάνισή της στην οθόνη

Δραστηριότητες

1. α) Προσπαθήστε να δώσετε το όνομά σας σε μία μεταβλητή ΟΝΟΜΑ και στη συνέχεια εμφανίστε το στη Σκηνή.
   β) Προσπαθήστε να εμφανίσετε στην οθόνη το όνομά σας, χωρίς να το ξαναγράψετε με το συνοδευτικό μήνυμα «Το όνομα μου είναι ... ».
2. Γράψτε δίπλα από τις εντολές εξόδου τι θα εμφανιστεί στην οθόνη μετά την εκτέλεση των εντολών;

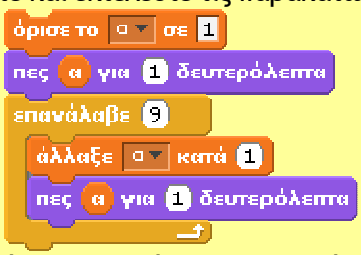
   Α.

```
όρισε το ζώο σε λιοντάρι
πες ζώο
πες λιοντάρι
πες ζώο
όρισε το ζώο σε σκύλο
πες ένωσε έχω ένα ζώο
```

   Β.

```
όρισε το X σε 3
πες 12 + 5 * X
πες 2 * 5 - X * 4
όρισε το X σε 8
πες 14 + 2 + X / 2
```

3. Γράψτε και εκτελέστε τις παρακάτω εντολές:

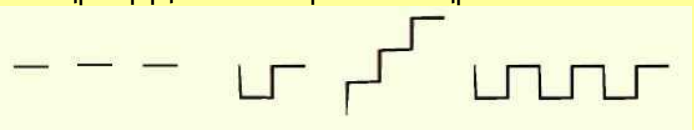


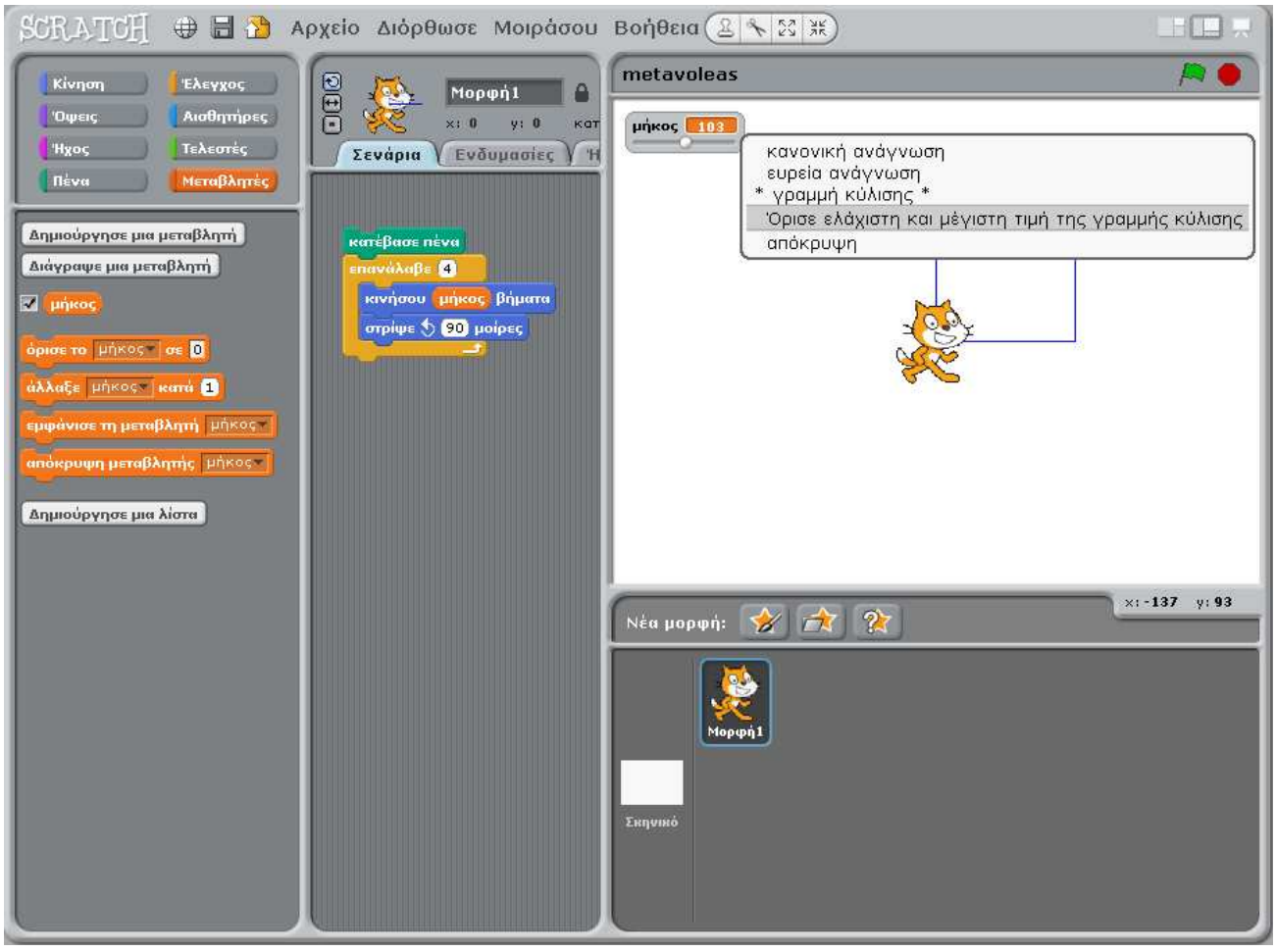
Ποιο είναι το αποτέλεσμα της εκτέλεσης της διαδικασίας;

Βρείτε ποια είναι η λειτουργία της εντολής 

, ώστε να μπορείτε να τη χρησιμοποιήσετε και στις επόμενες ασκήσεις.

4. Να δημιουργήσετε μία διαδικασία που να κατασκευάζει ένα ορθογώνιο παραλληλόγραμμο δίνοντάς του τα εκάστοτε μήκη των πλευρών.
5. Να καταγραφεί η διαδικασία που να υπολογίζει το εμβαδόν ενός τριγώνου πλευράς α και ύψους υ.
6. Δημιουργήστε τα 4 παρακάτω σχήματα

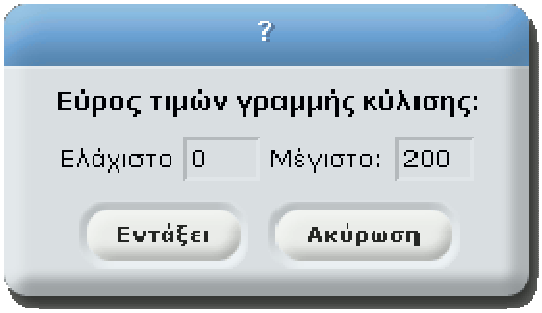




Εικόνα 2.7. Το σενάριο τετράγωνο παίρνει τιμές από το μεταβολέα «μήκος».

Μία παραλλαγή του σεναρίου τετράγωνο ή παίζοντας με τη γραμμή κύλισης μεταβλητών

Αντί να δίνουμε κάθε φορά το μήκος της πλευράς στο σενάριο *s* («τετράγωνο»), θα μπορούσαμε να χρησιμοποιήσουμε τη γραμμή κύλισης μίας μεταβλητής (Εικόνα 2.7). Κάνουμε δεξί κλικ στην ένδειξη της μεταβλητής επάνω στη Σκηνή και επιλέγουμε *γραμμή κύλισης* για να εμφανιστεί ο μεταβολέας. Για να αλλάξουμε τα όρια των τιμών που παίρνει ο μεταβολέας, ξανακάνουμε δεξί κλικ, επιλέγουμε *Όρισε ελάχιστη και μέγιστη τιμή της γραμμής κύλισης* και εμφανίζεται στην οθόνη μας η Εικόνα 2.8.



Εικόνα 2.8. Παράθυρο καθορισμού τιμών του μεταβολέα.

Αν εκτελέσουμε το σενάριο «τετράγωνο», η μεταβλητή μήκος παίρνει αυτόματα τιμή από το μεταβολέα (Εικόνα 2.7).

2.7 Επιλέγοντας...

Αρκετές φορές η περιγραφή της λύσης ενός προβλήματος δεν είναι μία ακολουθία βημάτων που πρέπει να εκτελεστούν όλα σε σειρά το ένα μετά το άλλο. Υπάρχουν προβλήματα που, για να λυθούν, πρέπει να επιλέγουμε ποια βήματα θα εκτελεστούν.

1ο Παράδειγμα

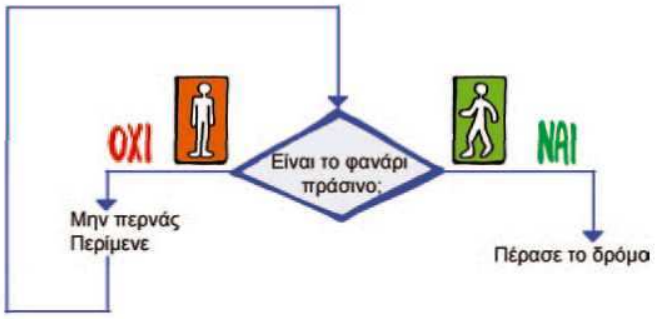
Αν θέλουμε να περάσουμε το δρόμο, επιλέγουμε τι θα κάνουμε ανάλογα με το τι δείχνει το φανάρι. Αν το φανάρι για τους πεζούς είναι πράσινο, περνάμε το δρόμο. Αν είναι κόκκινο, σταματάμε και περιμένουμε. Όμοια, για να ξέρουμε, αν πρέπει να πάρουμε μαζί μας ομπρέλα ή γυαλιά ηλίου, οφείλουμε να κάνουμε τους παρακάτω ελέγχους:

- Αν βρέχει, τότε θα πάρουμε μαζί μας ομπρέλα.
- Αν ο ήλιος είναι δυνατός, τότε πρέπει να φορέσουμε τα γυαλιά ηλίου.

Στην παράγραφο αυτή θα έχουμε την ευκαιρία να μάθουμε πώς να δίνουμε εντολές στον υπολογιστή, ώστε να επιλέγει, ανάλογα με τις συνθήκες που ισχύουν, ποια βήματα πρέπει να εκτελέσει.

Αν θέλουμε να γράψουμε έναν αλγόριθμο που να τον εκτελεί ένα μικρό παιδί, ώστε να διασχίσει με ασφάλεια το δρόμο, πρέπει να συμπεριλάβουμε τον έλεγχο του φαναριού (Εικόνα 2.10). Ο αλγόριθμος μπορεί να περιγραφεί με βήματα ως εξής:

1. Περπάτησε μέχρι την άκρη του πεζοδρομίου.
2. Έλεγξε το σηματοδότη για τους πεζούς.
3. Αν ο σηματοδότης είναι πράσινος, **τότε** πέρασε προσεκτικά το δρόμο· **διαφορετικά** (δηλ. αν είναι κόκκινος), περίμενε στην άκρη του πεζοδρομίου μέχρι το φανάρι να γίνει πράσινο.



Εικόνα 2.10. Σχηματική αναπαράσταση του αλγορίθμου έλεγχος φαναριού πεζών

2ο Παράδειγμα

Να γραφεί ένας αλγόριθμος που θα μας δίνει την απόλυτη τιμή ενός αριθμού.

Αν θυμηθούμε λίγο τα Μαθηματικά, η απόλυτη τιμή ενός αριθμού x ισούται με:

- x, αν x > 0,
- 0, αν x = 0 και

• -x, αν x < 0.

Επομένως, ο αλγόριθμος για την εύρεση της απόλυτης τιμής ενός αριθμού, με μια μικρή μετατροπή, μπορεί να διαμορφωθεί ως εξής:

• Μάθε την τιμή του x.

• Αν το x είναι μικρότερο από το 0 τότε υπολόγισε την τιμή -x (δηλαδή -1 * x) και εμφάνισέ την

• διαφορετικά εμφάνισε το x.

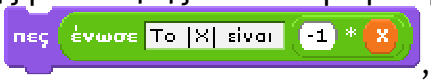
Η εντολή της γλώσσας Scratch που χρησιμεύει για την εκτέλεση του παραπάνω αλγορίθμου από τον υπολογιστή είναι η:

Με την εντολή αυτή ο υπολογιστής ελέγχει αρχικά, αν ισχύει η συνθήκη. Στη συνέχεια ανάλογα με το αν ισχύει (είναι αληθής), εκτελεί την πρώτη εντολή ή ομάδα εντολών, διαφορετικά εκτελεί τη δεύτερη.

Η συνθήκη είναι μία λογική πρόταση και εικονίζεται πάντα ως «κουτάκι» με γωνίες. Χρησιμοποιεί συνήθως τα παρακάτω σύμβολα, που ονομάζονται και λογικοί τελεστές.

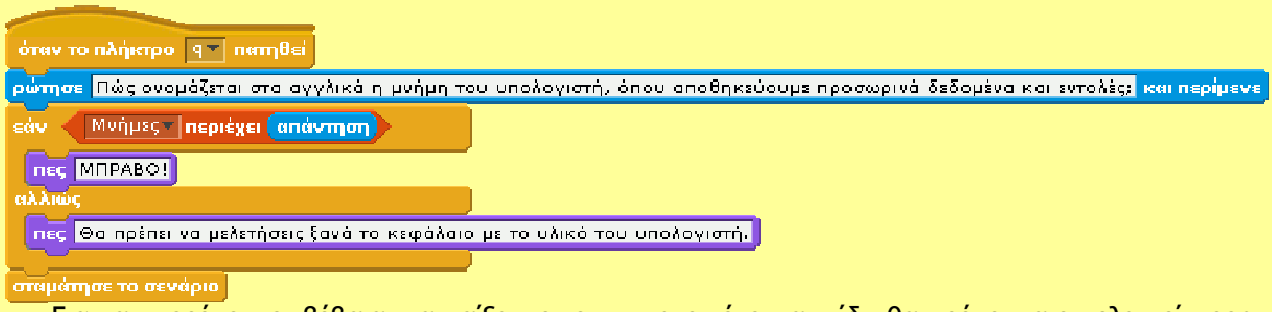
Σύμβολο	Σημασία	Παράδειγμα
	ισότητα	
	μεγαλύτερο	
	μικρότερο	
	αν μία τιμή περιέχεται σε μία λίστα τιμών	

Από τον αλγόριθμο της εύρεσης της απόλυτης τιμής ενός αριθμού μπορεί να προκύψει το εξής σενάριο:

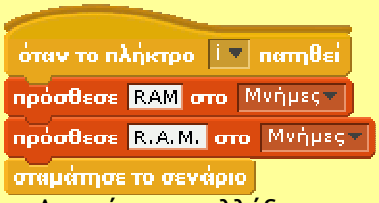
Στο σενάριο του παραδείγματος η συνθήκη ελέγχου είναι η  και χρησιμοποιείται για να ελέγξει αν η τιμή της μεταβλητής  είναι μικρότερη από το μηδέν. Αν είναι, τότε εκτελείται η πρώτη εντολή  , η οποία ανακοινώνει στην οθόνη την τιμή του  με το συνοδευτικό μήνυμα «το |X| είναι ». Σε διαφορετική περίπτωση, εκτελείται η δεύτερη εντολή, που εμφανίζει το ίδιο μήνυμα και στη συνέχεια την αρχική τιμή του  .

Δραστηριότητες

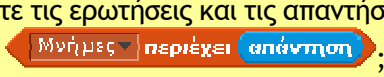
1. Φτιάξτε τα δικά σας παιχνίδια γνώσεων.
   Χρησιμοποιώντας τις εντολές επιλογής μπορούμε να φτιάξουμε τα δικά μας παιχνίδια γνώσεων και να παίζουμε με τους φίλους μας



Για να μπορέσουμε, βέβαια, να παίξουμε το συγκεκριμένο παιχνίδι, θα πρέπει να εκτελεστεί προηγουμένως το παρακάτω σενάριο, προκειμένου να εισαχθούν οι σωστές απαντήσεις στη λίστα Μνήμες.



α. Δοκιμάστε να αλλάξετε τις ερωτήσεις και τις απαντήσεις του παιχνιδιού γνώσεων.

β. Τι ελέγχει η συνθήκη 

;

2. *Μετρήστε πόσο χρόνο χρειαστήκατε, για να υπολογίσετε σωστά το γινόμενο δυο τυχαίων αριθμών μεταξύ του 1 και του 100.*

Για να μετρήσουμε το χρόνο, θα χρησιμοποιήσουμε την εντολή 

ώστε να μηδενίζεται ο χρόνος στην αρχή του σεναρίου. Με την ενεργοποίηση της παραπάνω εντολής ο χρόνος μετριέται σε δευτερόλεπτα και αποθηκεύεται στη μεταβλητή με το όνομα χρονόμετρο. Στη δραστηριότητα θα εμφανίζεται ο χρόνος, μόνο αν το αποτέλεσμα είναι σωστό, διαφορετικά θα εμφανίζεται ένα μήνυμα λάθους. Το σενάριο μπορεί να γραφεί ως εξής:

α. Τροποποιήστε την παραπάνω δραστηριότητα, ώστε να παίζουν δυο παίκτες. Με την τροποποίηση που θα κάνετε πρέπει το νέο πρόγραμμα να συγκρίνει τους δυο χρόνους και να εμφανίζει σε μήνυμα ποιος νίκησε.

β. Τροποποιήστε τη δραστηριότητα, ώστε το πρόγραμμα να συγκρίνει πέντε διαφορετικούς χρόνους και να εμφανίζει τον παίκτη που νίκησε τις περισσότερες φορές.

3. Δημιουργήστε ένα σενάριο που να υπολογίζει την τιμή του x στη συνάρτηση α*x+β=0.

2.8 Δημιουργώντας πιο σύνθετες εφαρμογές με τη γλώσσα Scratch

Δημιουργία μιας αριθμομηχανής

1η Δραστηριότητα

Στη δραστηριότητα αυτή θα κατασκευάσουμε μια αριθμομηχανή που να κάνει τις τέσσερις βασικές αριθμητικές πράξεις.

Βήμα 1: Στην Παλέτα εντολών δημιουργούμε δυο μεταβλητές με τα ονόματα α και β, και τις εμφανίζουμε στη Σκηνή. Ορίζουμε τις μεταβλητές έτσι ώστε να εμφανίζονται οι αντίστοιχες γραμμές κύλισης και να παίρνουν τιμές από 1 μέχρι και 100. Για να το πετύχουμε αυτό κάνουμε δεξί κλικ σε κάθε μία από τις μεταβλητές στη Σκηνή και από το μενού που εμφανίζεται επιλέγουμε *γραμμή κύλισης*. Στη συνέχεια ξανακάνουμε δεξί κλικ και επιλέγουμε *Όρισε ελάχιστη και μέγιστη τιμή της γραμμής κύλισης* για να καθορίσουμε τα όρια των τιμών που θα παίρνουν οι μεταβλητές.

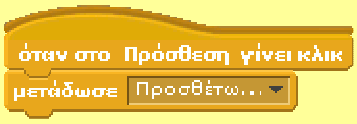
Βήμα 2: Δημιουργούμε τέσσερα κουμπιά, ένα για την κάθε πράξη, με τα ονόματα «Πρόσθεση», «Αφαίρεση», «Πολλαπλασιασμός» και «Διαίρεση». Τα κουμπιά εισάγονται ως νέες μορφές με το εικονίδιο  από τις έτοιμες που υπάρχουν στο φάκελο Things. Στη συνέχεια επιλέγοντας 

από την καρτέλα *Ενδυμασίες* της κάθε μορφής μπορούμε να προσθέσουμε το κατάλληλο σύμβολο.

Βήμα 3: Κάθε κουμπί στη Σκηνή δημιουργείται για κάποιο σκοπό˙ συνήθως με την ενεργοποίησή του εκτελείται ένα σενάριο ή ομάδα εντολών. Κάνοντας κλικ στο κάθε κουμπί θέλουμε να ενεργοποιείται η γάτα και να εμφανίζει το αποτέλεσμα της αντίστοιχης πράξης. Για το λόγο αυτό θα χρειαστεί να μεταδοθεί ένα διαφορετικό μήνυμα για κάθε κουμπί στο οποίο θα κάνουμε κλικ. Το μήνυμα αυτό θα το λάβει η γάτα και θα εκτελέσει το αντίστοιχο σενάριο υπολογισμού του αποτελέσματος.

Βήμα 4: Δημιουργούμε ένα σενάριο για κάθε πράξη που θα εκτελεί η γάτα με το οποίο θα εμφανίζεται και το αποτέλεσμα.

Το πρώτο ζεύγος σεναρίων που αντιστοιχεί στο κουμπί της Πρόσθεσης είναι

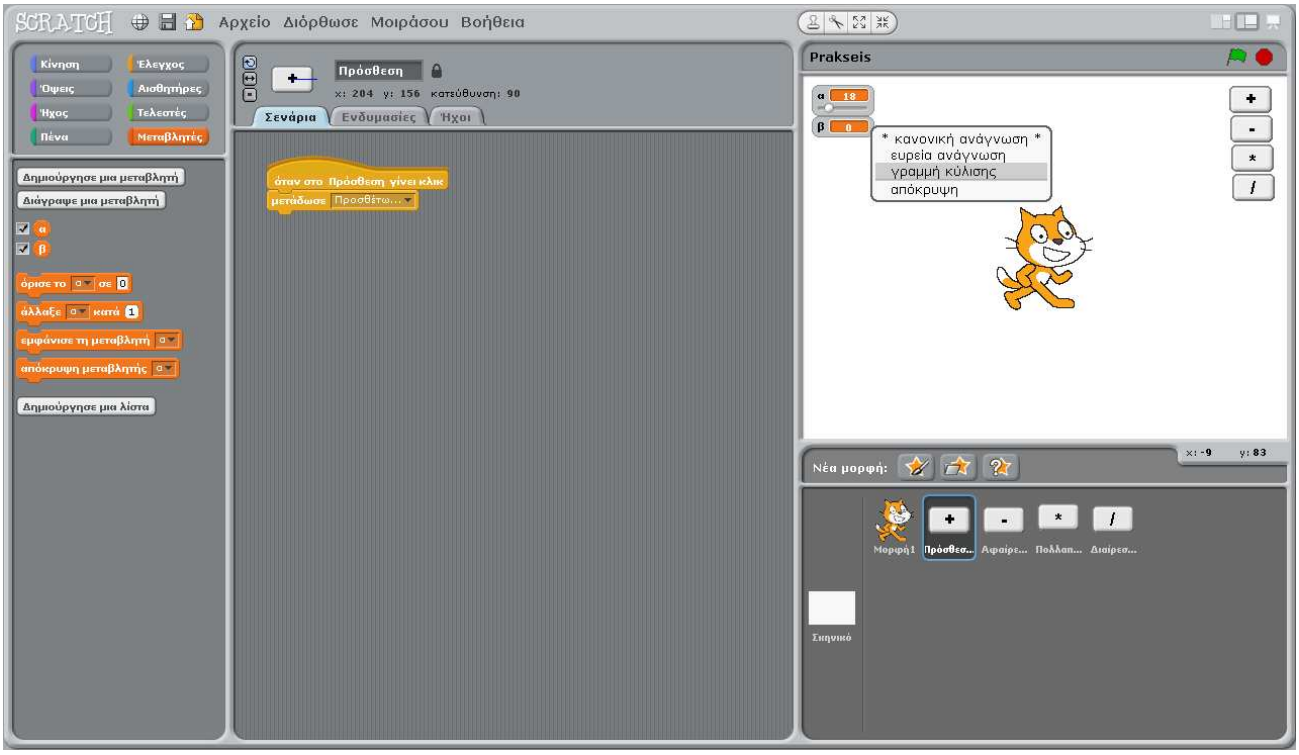


και


για τις μορφές  και  αντίστοιχα.

Στην παρακάτω εικόνα φαίνεται τόσο το σενάριο της μορφής  , όσο και το μενού που εμφανίζεται κάνοντας δεξί κλικ στη μεταβλητή β.

Άσκηση: Να δημιουργήσετε τα κατάλληλα σενάρια, ώστε να λειτουργούν και τα τέσσερα κουμπιά στην αριθμομηχανή μας.



Κυνηγητό σκύλου γάτας

2η Δραστηριότητα

Μία πιο σύνθετη δραστηριότητα περιλαμβάνει περισσότερες κινούμενες μορφές. Επίσης οι μορφές μπορούν να αλλάζουν ενδυμασίες, ώστε να δημιουργούμε πιο ελκυστικά σενάρια.

Στην παρακάτω εικόνα έχουμε βάλει δυο μορφές, της γάτας και του σκύλου. Η εναλλαγή στην ενδυμασία μίας μορφής γίνεται ως εξής:

Βήμα 1: Επιλέγουμε την μορφή που θέλουμε από τη Λίστα των Μορφών.

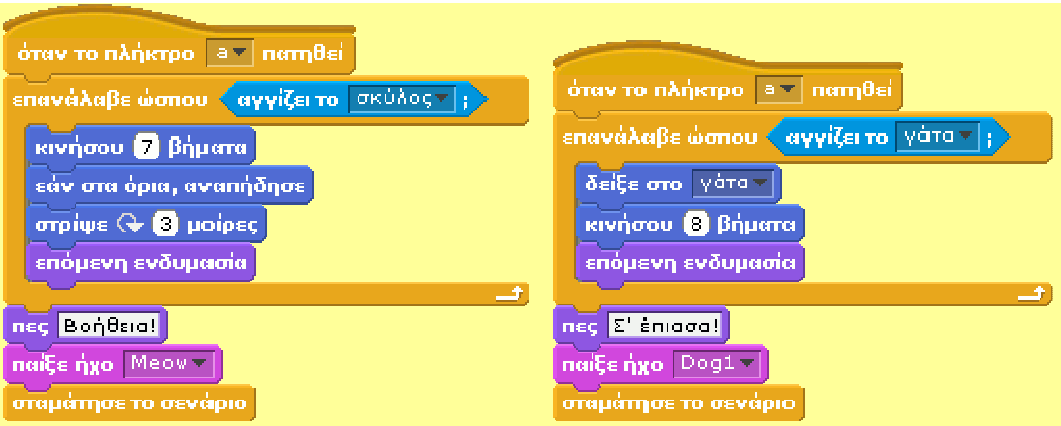
Βήμα 2: Κάνουμε κλικ στην καρτέλα Ενδυμασίες και με το κουμπί Εισαγωγή εμφανίζεται ένα παράθυρο από όπου μπορούμε να επιλέξουμε την ενδυμασία που μας εξυπηρετεί.

Βήμα 3: Στην περίπτωση που θέλουμε οι ενδυμασίες της μορφής να εναλλάσσονται, ώστε να δημιουργείται η ψευδαίσθηση της κίνησης, εισάγουμε σε μία εντολή επανάληψης την εντολή επόμενη ενδυμασία από την ομάδα Όψεις.

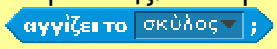
Οι μορφές αρχικά παίρνουν όνομα τύπου Μορφή1, Μορφή2, Μορφή3 ... Μπορούμε, όμως, να αλλάζουμε τα ονόματά τους με επιλογή πάνω στη μορφή και κλικ στο πλαίσιο πάνω από τις καρτέλες της Περιοχής σεναρίων.

Στη συνέχεια μπορούμε να καλούμε τη μορφή με το νέο όνομα, όπως π.χ.: γάτα, σκύλος

Η δημιουργία των σεναρίων έχει ως αποτέλεσμα ένα «κυνηγητό» μεταξύ γάτας και σκύλου, μέχρι ο σκύλος να ακουμπήσει τη γάτα. Τότε τα σενάρια σταματούν και ο σκύλος εμφανίζει το μήνυμα «Σ' έπιασα», ενώ η γάτα ζητά απεγνωσμένα βοήθεια. Παρακάτω φαίνονται τα δύο σενάρια, για τη γάτα και το σκύλο αντίστοιχα.



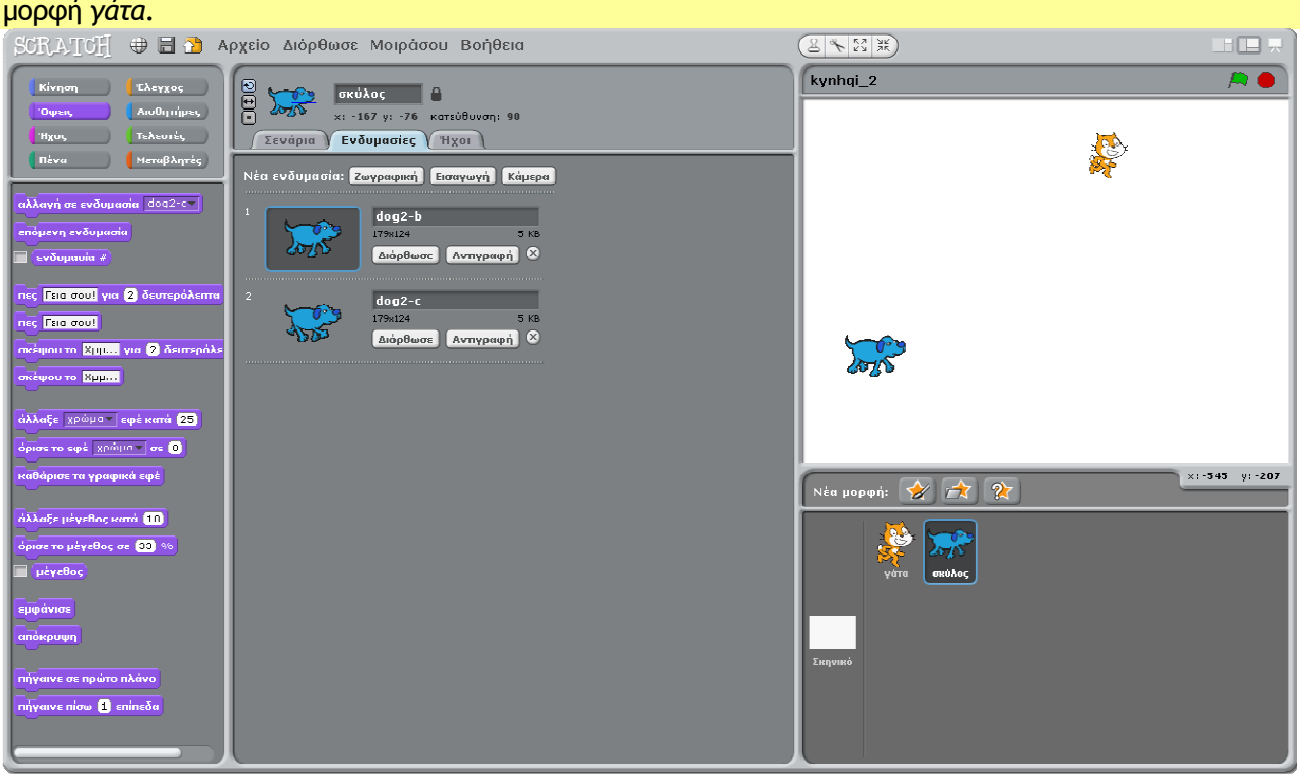
Η εντολή χρησιμοποιεί μια λογική πρόταση για να ελέγξει την επανάληψη(την εκτέλεση)/τον τερματισμό της εκτέλεσης των εντολών στο εσωτερικό της. Η λογική πρόταση είναι η: 

για το σενάριο που εκτελεί ο σκύλος, και 

για το σενάριο της γάτας. Αυτές οι λογικές προτάσεις είναι ΑΛΗΘΕΙΣ όταν οι δύο μορφές σκύλος και γάτα έρχονται σε επαφή.

Η εντολή 

αλλάζει συνεχώς την κατεύθυνση της μορφής *σκύλος*, ώστε να κοιτάει προς τη μορφή *γάτα*.



Άσκηση: Τροποποιήστε την παραπάνω δραστηριότητα, ώστε η γάτα να κινείται σε μία διεύθυνση μόνο. Ποια εντολή πρέπει να διαγράψετε;

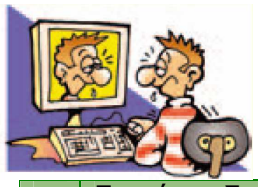
Στη συνέχεια δημιουργήστε δυο σενάρια που το καθένα απ' αυτά θα ενεργοποιείται από τα βελάκια «←» και «→» στο πληκτρολόγιο. Το κουμπί «→» θα αλλάζει τη κατεύθυνση της μορφής με το όνομα γάτα προς τα δεξιά κατά 90 μοίρες. Αντίστοιχα το κουμπί «←» θα αλλάζει τη κατεύθυνση της μορφής με το όνομα γάτα προς τα αριστερά κατά 90 μοίρες.

ΑΝΑΚΕΦΑΛΑΙΩΣΗ

Στη ζωή μας πολλές φορές καλούμαστε να λύσουμε πολλά και ποικίλα προβλήματα. Μερικά από αυτά τα προβλήματα μπορούμε να τα λύσουμε και με τη βοήθεια υπολογιστή. Η περιγραφή της λύσης ενός προβλήματος με λογικά βήματα ονομάζεται «**αλγόριθμος**». Για να υλοποιήσουμε έναν αλγόριθμο στον υπολογιστή, πρέπει να τον γράψουμε με μια γλώσσα προγραμματισμού. Η συγγραφή ενός προγράμματος σε μια γλώσσα προγραμματισμού γίνεται συνήθως σε ειδικά περιβάλλοντα που μας διευκολύνουν στη σύνταξή του. Επίσης μετατρέπουν τον **κώδικα** με τον οποίο γράφουμε σε μορφή κατάλληλη, ώστε να τον εκτελέσει ο υπολογιστής.

Υπάρχουν πολυάριθμες **γλώσσες προγραμματισμού**. Η καθεμία έχει δικά της χαρακτηριστικά (αλφάβητο, λεξιλόγιο, συντακτικό) και δυνατότητες. Η επιλογή της κατάλληλης γλώσσας προγραμματισμού εξαρτάται σε μεγάλο βαθμό από τις λειτουργίες του προγράμματος που σχεδιάζουμε.

Σε αυτές τις σημειώσεις παρουσιάζεται το περιβάλλον προγραμματισμού Scratch.



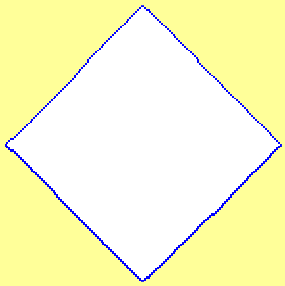
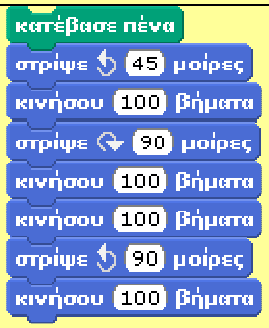
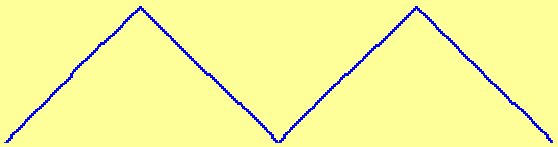
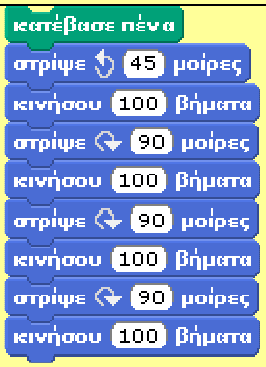
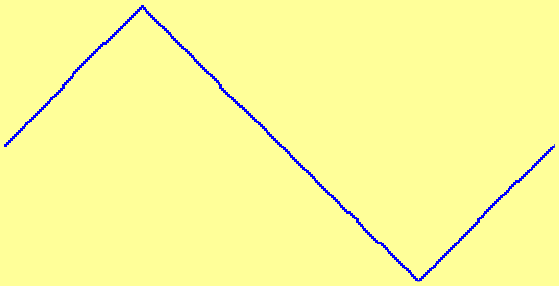
ΑΣΚΗΣΕΙΣ ΑΥΤΟ-ΑΞΙΟΛΟΓΗΣΗΣ

1. Χαρακτηρίστε τις παρακάτω προτάσεις ως σωστές ή λάθος βάζοντας δίπλα στα αντίστοιχα κελιά Σ ή Λ. Στην περίπτωση που πιστεύετε ότι είναι λάθος, σκεφτείτε ποια θα μπορούσε να είναι η αντίστοιχη σωστή πρόταση.

	Προτάσεις Σωστού-Λάθους	Σ ή Λ
1	Ένα πρόβλημα μπορεί να λυθεί πάντα με μαθηματικούς υπολογισμούς.	
2	Η επίλυση ενός προβλήματος προηγείται της κατανόησής του.	
3	Πρέπει να καθορίσουμε τα ζητούμενα ενός προβλήματος, για να μπορέσουμε να το επιλύσουμε.	
4	Όλα τα προβλήματα έχουν λύση.	
5	Ένας αλγόριθμος πρέπει πάντοτε να «εξασφαλίζει» το ότι θα τερματίσει.	
6	Η εντολή «Πες ένα αστείο» είναι αυστηρά καθορισμένη.	
7	Ένα πρόγραμμα είναι η γραφή ενός αλγορίθμου σε μια γλώσσα προγραμματισμού.	
8	Υπάρχουν πολλές διαφορετικές γλώσσες, για να προγραμματίσουμε έναν υπολογιστή.	
9	Ο μεταφραστής βρίσκει τα λογικά λάθη ενός προγράμματος.	
10	Η γλώσσα που καταλαβαίνει ένας υπολογιστής είναι η γλώσσα μηχανής.	

2. Σχεδίασε το αποτέλεσμα που προκύπτει από τις παρακάτω εντολές στο περιβάλλον της γλώσσας Scratch. (Σημείωση: στο περιβάλλον έχουμε τοποθετήσει μία μορφή, ώστε να μπορούν να εκτελεστούν οι εντολές που ακολουθούν.)

3. Αντιστοιχίστε τα σχήματα στα δεξιά με τα τμήματα του κώδικα στα αριστερά.

<table>
<tr>
<td>A.
 
</td>
<td>1.
  </td>
</tr>
<tr>
<td>B.
 
</td>
<td>2.
  </td>
</tr>
<tr>
<td>Γ.
 
</td>
<td>3.
  </td>
</tr>
</table>

ΔΡΑΣΤΗΡΙΟΤΗΤΕΣ

Κεφάλαιο 1: Εισαγωγή στην Έννοια του Αλγορίθμου και στον Προγραμματισμό

1. Σε ποια απλούστερα προβλήματα μπορεί να χωριστεί το πρόβλημα των μαθητικών εκλογών; Σε τι μας βοηθάει η ανάλυση του προβλήματος σε επιμέρους προβλήματα; Ποιος είναι ο χώρος του προβλήματος;
2. Να αναλύσετε το πρόβλημα «αγορά Η/Υ» σε απλούστερα προβλήματα.
3. Προσπαθήστε να επιλύσετε το παρακάτω πρόβλημα: Έστω ότι δεν διαθέτουμε ρολόι αλλά δύο κλεψύδρες, των 7 λεπτών η μία και των 4 λεπτών η άλλη. Πώς μπορούμε να ξέρουμε πότε ακριβώς περάσανε 9 λεπτά;
4. Φτιάξτε ένα διάγραμμα με τα βήματα που πρέπει να ακολουθήσουμε, για να κατασκευάσουμε ένα βαρκάκι ή ένα αεροπλανάκι με ένα τετράγωνο χαρτί.

5. Να γραφεί ένας αλγόριθμος που να περιγράφει σε κάποιον που δεν ξέρει, πώς μπορεί να κάνει ποδήλατο. Τι πρόβλημα μπορεί να έχει αυτός ο αλγόριθμος;
6. Να γραφεί ένας αλγόριθμος που να περιγράφει σε ένα μικρό παιδί πώς να σχηματίσει ένα ορθογώνιο παραλληλόγραμμο με τα βήματά του στην άμμο.
7. Στον παρακάτω αλγόριθμο:
   Δώσε μου το έτος που έχουμε σήμερα.
   Δώσε μου το έτος που γεννήθηκες.
   Η ηλικία σου υπολογίζεται με το άθροισμα του έτους που γεννήθηκες και του έτους που έχουμε σήμερα.
   Εμφάνιση της ηλικίας.
   Πώς θα χαρακτηρίζαμε το λάθος που υπάρχει;

ΘΕΜΑΤΑ ΓΙΑ ΣΥΖΗΤΗΣΗ

1. Να αναφέρετε, από τις εγκυκλοπαιδικές σας γνώσεις, μερικά προβλήματα που παραμένουν άλυτα ή που έχει αποδειχτεί ότι δεν έχουν λύση.
2. Γιατί νομίζετε ότι υπάρχουν πολλές γλώσσες προγραμματισμού; Θα μπορούσαν να αντικατασταθούν όλες οι γλώσσες προγραμματισμού με μία;

Βιβλιογραφία

Αράπογλου Α. κ.α., (2007). Πληροφορική, Ο.Ε.Δ.Β., Αθήνα.

Lifelong Kinderkarten Group, (2009). Οδηγός χρήσης Scratch – έκδοση 1.4, MIT Media Lab, http://info.scratch.mit.edu/Languages & http://scratch.mit.edu (η.τ.ε. 07/05/2010).