Κεφ.14: Μεταβλητές & Λίστες

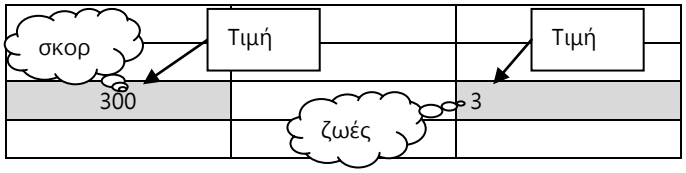
Σε αυτό το κεφάλαιο:

«Το πιο απίθανο πράγμα με τη ζωή είναι ότι πάντα θα υπάρχουν μεταβλητές. Θα πρέπει να τις δεις και να τις αναγνωρίσεις για να μάθεις πώς να αλληλεπιδράς μ'αυτές».
(Suzanne Farrell)

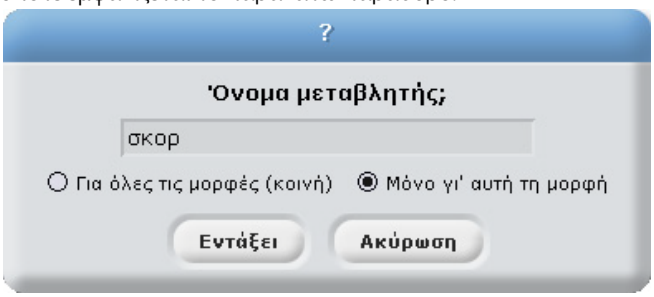
14.1. Εισαγωγή στην έννοια των μεταβλητών

14.1.1 Οι μεταβλητές

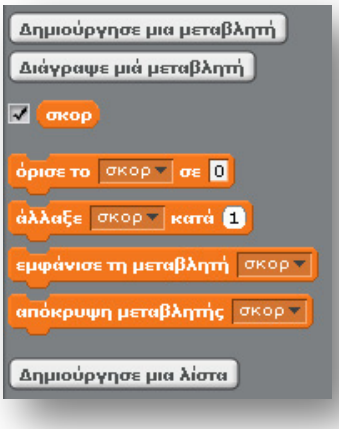
Θα πρέπει να έχετε παρατηρήσει ότι έχουμε φτιάξει τόσα παιχνίδια μέχρι αυτό το σημείο και δεν έχουμε αναφερθεί πουθενά για το πως μπορούμε να δημιουργήσουμε τα πιο σημαντικά στοιχεία όλων των παιχνιδιών: ζωές και σκορ! Θα μπορούσαμε με κάποιο τρόπο να κρατάμε το σκορ στις προσπάθειες που κάναμε για μεταφερθεί το ελικοπτεράκι από την κίτρινη πλατφόρμα στην κόκκινη; Θα μπορούσαμε να έχουμε ένα μέγιστο αριθμό προσπαθειών πέρα από τον οποίο θα σταματούσε η εκτέλεση του έργου μας, δηλαδή θα τελείωνε το παιχνίδι; Θα μπορούσαμε να εμφανίσουμε τις ζωές του pacman που μας απομένουν και το σκορ που μαζεύει τρώγοντας φρουτάκια;
Προφανώς! Είπαμε ότι στο τέλος του βιβλίου θα γνωρίζετε να δημιουργείτε ολοκληρωμένα παιχνίδια. Δεν θα ξελέμε τώρα!
Πως, λοιπόν, θα διατηρούμε το σκορ καθ' όλη τη διάρκεια εκτέλεσης ενός έργου; Θα το αποθηκεύσουμε ως μια μεταβλητή. Που; Στη μνήμη του υπολογιστή.
Οι μεταβλητές είναι συμβολικά ονόματα που αντιστοιχούν σε θέσεις μνήμης του υπολογιστή. Στις θέσεις μνήμης αυτές, αποθηκεύουμε διάφορες τιμές (π.χ. το σκορ, τον αριθμό ζωών, την απάντηση ενός χρήστη κτλ). Οι τιμές αυτές μπορούν να αλλάξουν, να διαβαστούν ή να αντιγραφούν. Μπορείτε να τις φανταστείτε σαν κουτάκια που έχουν ένα όνομα και μια τιμή. Στην παρακάτω εικόνα, θεωρώντας ότι ο πίνακας είναι η μνήμη του υπολογιστή, βλέπουμε δυο θέσεις στη μνήμη που αντιστοιχίζονται στις μεταβλητές «σκορ» και «ζωές» που έχουν τιμές 300 και 3 αντίστοιχα.



Άρα, όταν δημιουργούμε μια μεταβλητή, ουσιαστικά δεσμεύουμε μια θέση στη μνήμη του υπολογιστή στην οποία μπορούμε να αποθηκεύσουμε όποια τιμή θελήσουμε.
Στο Scratch, για να δημιουργήσουμε μια μεταβλητή πρέπει αρχικά να μεταφερθούμε στην παλέτα εντολών **μεταβλητές** και να πατήσουμε στην επιλογή «Δημιούργησε μια μεταβλητή» οπότε εμφανίζεται το παρακάτω παράθυρο:



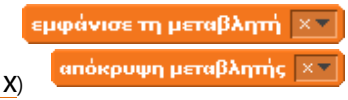
Στο παράθυρο αυτό πρέπει να προσδιορίσουμε:
α) **το όνομα της μεταβλητής**: με την ονομασία της μεταβλητής προσδιορίζουμε τον τρόπο με τον οποίο θα αναφερόμαστε στη συγκεκριμένη θέση μνήμης και θα αλλάζουμε τα περιεχόμενά της, δηλαδή την τιμή της. Καλό είναι στις μεταβλητές που δημιουργούμε να δίνουμε ονόματα με νόημα, δηλαδή ονόματα που προσδιορίζουν τη χρησιμότητα της μεταβλητής. Στο συγκεκριμένο παράδειγμα θα ονομάσουμε τη μεταβλητής μας «σκορ».
β) **το εύρος χρήσης της μεταβλητής**: θα πρέπει επίσης να προσδιορίσουμε αν η νέα μεταβλητή θέλουμε να είναι ορατή και να τροποποιείται από όλα τα αντικείμενα του έργου μας ή αν θέλουμε να αξιοποιείται μόνο από ένα συγκεκριμένο αντικείμενο. Στη δεύτερη περίπτωση, η μεταβλητή αυτή μπορεί να χρησιμοποιηθεί μόνο σε σενάρια του συγκεκριμένου αντικειμένου ενώ στην πρώτη, όλα τα αντικείμενα μπορούν να αλλάξουν την τιμή της μεταβλητής.
Αφού δημιουργηθεί η μεταβλητή, ως δια μαγείας, η παλέτα **μεταβλητές** εμφανίζει νέες εντολές!



Η πρώτη εντολή **όρισε το Χ σε 0**, είναι η σημαντικότερη, καθώς μας δίνει τη δυνατότητα να δώσουμε μια τιμή στη μεταβλητή μας συμπληρώνοντας το λευκό κουτάκι. Όταν δημιουργούμε μια μεταβλητή, η αρχική της τιμή είναι κενή. Με τη χρήση της εντολής **όρισε το Χ σε 0** μπορούμε να της δώσουμε αρχική τιμή αλλά και να αλλάξουμε την τιμή αυτή. Π.χ. σε ένα παιχνίδι θα ορίζαμε τη μεταβλητή «ζωές» σε 3 αρχικά ενώ τη μεταβλητή «σκορ» σε 0. Οι μεταβλητές μπορούν να πάρουν και αρνητικές τιμές.

Η δεύτερη εντολή που γίνεται διαθέσιμη είναι η **άλλαξε Χ κατά 1**, η οποία αλλάζει την τρέχουσα τιμή της μεταβλητής κατά την τιμή που προσδιορίζουμε στο λευκό κουτάκι. Άλλαξε την τιμή κατά 1, σημαίνει αύξηση της τιμής της μεταβλητής κατά 1 μονάδα ενώ αύξηση κατά -5 σημαίνει μείωση της αρχικής τιμής κατά 5 μονάδες. Το μεγάλο πλεονέκτημα της συγκεκριμένης εντολής είναι μας επιτρέπει να αλλάζουμε την τιμή της μεταβλητής χωρίς να ενδιαφερόμαστε για το ποια είναι η τρέχουσα τιμή της. Αν εκτελούσαμε τη συγκεκριμένη εντολή για τη μεταβλητή «σκορ», τότε το σκορ θα αυξανόταν κατά 1 ενώ αν εκτελούσαμε την εντολή για τη μεταβλητή «ζωές» με παράμετρο -1, η τιμή της αντίστοιχης μεταβλητής θα γινόταν 2.

Η τρίτη και η τέταρτη εντολή (**εμφάνισε τη μεταβλητή Χ**, **απόκρυψη μεταβλητής Χ**) αναφέρονται στην εμφάνιση και στην απόκρυψη μια μεταβλητής από την οθόνη του Scratch. Σε αντίθεση με άλλες μεταβλητές κατάστασης, όπως η **θέση Χ** και η **ενδυμασία #**, η εμφάνιση των τιμών των μεταβλητών στην οθόνη του Scratch ελέγχεται προγραμματιστικά. Έτσι, για παράδειγμα, όταν ο χρήστης μας χάσει μια ζωή, μπορούμε να εμφανίσουμε τη μεταβλητή «ζωές» για λίγα δευτερόλεπτα και στη συνέχεια να την αποκρύψουμε.



Ας δούμε πιο πρακτικά τη χρησιμότητα και τη λειτουργία των μεταβλητών μέσα από παραδείγματα. Δεν θα παρουσιάσουμε ολοκληρωμένα παραδείγματα αλλά μόνο τα τμήματά τους που αφορούν τη λειτουργία των μεταβλητών.

14.1.2 Παραδείγματα εφαρμογών με μεταβλητές

Παράδειγμα: Ο «κ.Γατίδης μετράει ως το 10»: Τι θα λέγατε να βάλουμε τον κ.Γατίδη να μετράει μέχρι το 10 με τη χρήση των μεταβλητών. Η πρώτη λύση που μας έρχεται στο μυαλό:

Τι θα συνέβαινε όμως αν θέλαμε να μετρήσει μέχρι το 30; Θα δημιουργούσαμε 30 εντολές; Θα μπορούσαμε αντί για αυτή τη λύση, να δημιουργήσουμε μια μεταβλητή, να αυξάνουμε κατά μια μονάδα την τιμή της και να βάλουμε το χαρακτήρα μας να αναφέρει την τιμή αυτή μέσα σε μια επανάληψη; Δεν θα ήταν πιο εύκολο;

Χρειαζόμαστε λοιπόν μια μεταβλητή που θα την ονομάσουμε «αριθμός» και την οποία θα κάνουμε διαθέσιμη μόνο στον κ.Γατίδη. Από την παλέτα **Μεταβλητές** επιλέγουμε «δημιούργησε μια μεταβλητή», προσδιορίζουμε ως όνομα της μεταβλητής το «αριθμός» και επιλέγουμε η μεταβλητή να είναι διαθέσιμη μόνο για το συγκεκριμένο αντικείμενο.

Θα δώσουμε στη μεταβλητή, αρχική τιμή 1 και στη συνέχεια μέσα σε μια επανάληψη θα βάλουμε τον κ.Γατίδη να λέει την τιμή της μεταβλητής ενώ θα αλλάζουμε την τιμή της κατά 1. Μελετήστε το παρακάτω σενάριο:

[14_π01.sb]

Όπως παρατηρούμε από το συγκεκριμένο παράδειγμα, θα μπορούσαμε πολύ εύκολα να βάλουμε τον κ.Γατίδη να μετράει μέχρι το 1000 απλά αλλάζοντας τον αριθμό των επαναλήψεων στην εντολή **επανέλαβε Χ**!

Για να μεταφέρουμε τη μεταβλητή μέσα σε ένα λευκό κουτάκι, επιλέγουμε και σέρνουμε την αντίστοιχη μεταβλητή κατάστασης που εμφανίζεται στην παλέτα **Μεταβλητές**.
Επίσης, είναι σημαντικό να θυμόμαστε:
α) ότι για κάθε μεταβλητή στην αρχή ενός προγράμματος θα πρέπει να προσδιορίζουμε **την αρχική τιμή της**,
β) η μεταβλητή μπορεί να χρησιμοποιηθεί σε οποιοδήποτε **λευκό κουτάκι των εντολών του Scratch** (π.χ. στην κίνηση, στη ζωγραφική, στον ήχο, στις συνθήκες κτλ.).

Παράδειγμα: χρονομετρητής: Έστω λοιπόν ότι θέλουμε να δημιουργήσουμε ένα χρονόμετρο για ένα παιχνίδι μας. Ο παίκτης θα μπορεί να παίξει το παιχνίδι για 60 δευτερόλεπτα. Πως θα μπορούσαμε να το καταφέρουμε;

[14_π02.sb]

Όπως βλέπετε στο προηγούμενο σενάριο, ο προγραμματιστής μας, δημιούργησε μια μεταβλητή που την ονόμασε «ρολόι».

Στόχος του ήταν η μεταβλητή αυτή να κρατά κάθε στιγμή το χρόνο που απομένει στον παίκτη. Στη συνέχεια, ο προγραμματιστής μας προσδιόρισε ως αρχική τιμή της μεταβλητής το 60, δηλαδή 60 δευτερόλεπτα παιχνιδιού. Χρησιμοποιώντας την εντολή **εμφάνισε μεταβλητή...** εμφάνισε τη μεταβλητή αυτή στην οθόνη του χρήστη ώστε ο παίκτης να γνωρίζει το χρόνο που του απομένει. Στη συνέχεια ζήτησε κάθε ένα δευτερόλεπτο να αλλάζει η τιμή της μεταβλητής «ρολόι», μέχρι η τιμή της να γίνει μηδέν. Αν η μεταβλητή μηδενιστεί, η επανάληψη σταματά, εκτελούνται οι δυο τελευταίες εντολές του σεναρίου και το παιχνίδι σταματά!

Προσέξτε ότι:

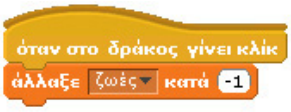
- ✓ έχουμε αρνητική τιμή στην εντολή **άλλαξε ρολόι κατά...**
- ✓ με την εντολή **εμφάνισε μεταβλητή...** παρουσιάζεται η τιμή της μεταβλητής στην οθόνη του Scratch,
- ✓ η μεταβλητή χρησιμοποιήθηκε μέσα σε μια συνθήκη.

Αν θέλαμε κάθε φορά που ο χρήστης έτρωγε ένα φρουτάκι να κερδίζει έξτρα χρόνο, τι θα έπρεπε να κάνουμε; Πολύ απλά, θα μπορούσαμε να χρησιμοποιήσουμε την εντολή **άλλαξε ρολόι κατά...** σε κάποιο διαφορετικό σενάριο και κάθε φορά που έτρωγε ο πρωταγωνιστής μας ένα φρουτάκι να προσθέταμε τον αντίστοιχο χρόνο που κέρδιζε στη μεταβλητή «ρολόι». Το παραπάνω σενάριο θα λειτουργούσε χωρίς κανένα πρόβλημα. Αυτό είναι και το πλεονέκτημα της εντολής **επανάλαβε ώσπου...** αντί της **επανάλαβε Χ** στη συγκεκριμένη περίπτωση. Μπορείτε να το εξηγήσετε καλύτερα;

Παράδειγμα: «Μακριά από το δράκο»: Έστω, λοιπόν, ότι έχουμε δημιουργήσει ένα παιχνίδι στο οποίο ο χρήστης προσπαθεί να κάνει κλικ σε όσο το δυνατόν περισσότερα Στρουμφάκια για να τα απελευθερώσει από το κάστρο του Δρακουμέλ. Αυτό όμως πρέπει να γίνει χωρίς να κάνουμε κλικ πάνω στο Δρακουμέλ που κινείται μανιωδώς προς το δείκτη του ποντικιού. Έστω ότι έχουμε 3 ζωές. Πως θα μπορούσαμε να υλοποιήσουμε το συγκεκριμένο παιχνίδι. Μελετήστε το παρακάτω σενάριο:

Στο συγκεκριμένο παράδειγμα, δημιουργήσαμε μια μεταβλητή που ονομάσαμε «ζωές» και η οποία προσδιορίσαμε ότι θέλουμε να είναι ορατή από όλα τα αντικείμενα του έργου μας. Στην αρχή του έργου μας, ο ήρωας έχει 3 ζωές και στη συνέχεια ζητήσαμε να συνεχίσει να εκτελείται το έργο μας μέχρι οι ζωές να γίνουν 0. Τότε εμφανίζεται μήνυμα τέλους του παιχνιδιού και σταματά η εκτέλεση του έργου. Πως όμως θα μειώνονται οι ζωές μας; Από το σενάριο του Δρακουμέλ. Μη ξεχνάτε ότι και ο Δρακουμέλ μπορεί να πειράζει αυτή τη μεταβλητή αφού είναι ορατή σε αυτόν.

Αρκεί να προσθέσουμε το παρακάτω σενάριο στο αντικείμενο δράκος:



Οι μεταβλητές, επομένως, μπορούν να χρησιμοποιηθούν και ως ένας τρόπος έμμεσης επικοινωνίας μεταξύ των αντικειμένων όταν είναι ορατές σε αυτά. Πολλαπλά αντικείμενα μπορούν να αλλάζουν τις τιμές μιας μεταβλητής, ενώ όλα τα αντικείμενα μπορεί να περιέχουν στα σενάριά τους συνθήκες που εξαρτώνται από αυτήν.

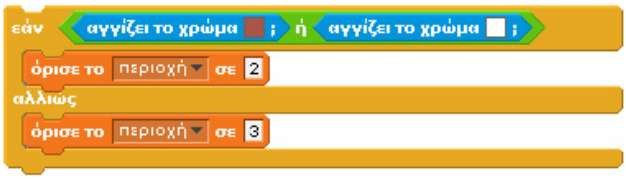
Παράδειγμα: Μπάσκετ: Έστω ότι θέλουμε να φτιάξουμε ένα πρόγραμμα με το οποίο να παίζουμε μπάσκετ. Στόχος του παιχνιδιού είναι να συγκεντρώσουμε όσους περισσότερους πόντους μπορούμε. Στο παιχνίδι αυτό λοιπόν θα προσομοιώσουμε ένα παίκτη που σουτάρει από διαφορετικές θέσεις στο γήπεδο! Για να το πετύχουμε αυτό, θα χρειαστούμε ένα αντικείμενο-χέρι για τον παίκτη που σουτάρει, ένα αντικείμενο-μπάλα και ένα αντικείμενο-καλάθι! Ο χρήστης μετακινεί το αντικείμενο-χέρι με τα βελάκια του πληκτρολογίου ενώ σουτάρει τη μπάλα πατώντας το πλήκτρο κενό για κατάλληλο χρονικό διάστημα. Όποτε η μπάλα αγγίζει τη στεφάνη του καλαθιού, θεωρούμε ότι ο παίκτης πέτυχε καλάθι. Κάθε φορά που ο παίκτης ρίχνει μία βολή και πετυχαίνει καλάθι από την περιοχή κάτω από το καλάθι, το σκορ του αυξάνεται κατά 2 πόντους. Κάθε φορά που πετυχαίνει καλάθι ενώ βρίσκεται έξω από την περιοχή, το σκορ του αυξάνεται κατά 3 πόντους. Στο τέλος του παιχνιδιού αυτό που θέλουμε να ξέρουμε είναι τους πόντους που έχει πετύχει ο παίκτης.

Είναι φανερό ότι κατά τη διάρκεια του παιχνιδιού πρέπει με κάποιο τρόπο να «κρατάμε» - αποθηκεύουμε τους πόντους του χρήστη και κάθε φορά που ο χρήστης πετυχαίνει καλάθι, να ενημερώνουμε το σκορ. Στο μυαλό μας έρχονται αμέσως οι μεταβλητές. Θα χρειαστούμε μία μεταβλητή στην οποία θα αποθηκεύουμε το σκορ του παίκτη κατά τη διάρκεια του παιχνιδιού. Πότε θα αλλάζουμε τη μεταβλητή «σκορ»; Όποτε το αντικείμενο «μπάλα» αγγίζει το αντικείμενο «στεφάνη». Σε ποιο αντικείμενο θα δημιουργήσουμε τη μεταβλητή; Σε ένα από τα δυο. Έστω ότι επιλέγουμε το πρώτο και ότι η μεταβλητή είναι ορατή μόνο στο συγκεκριμένο αντικείμενο.

Επιπλέον, θέλουμε να γνωρίζουμε από ποια περιοχή προήλθε η βολή που κατέληξε σε καλάθι έτσι ώστε να αυξήσουμε το σκορ κατά 2 ή κατά 3 πόντους.

Το πρόβλημά μας είναι πως το αντικείμενο-χέρι θα «μιλήσει» με το αντικείμενο-μπάλα ώστε να προσθέσει τον κατάλληλο αριθμό πόντων. Στο σημείο αυτό πρέπει να σκεφτούμε ότι μπορούμε να δημιουργήσουμε μία μεταβλητή που θα αποθη-

κεύει σε ποια περιοχή βρισκόταν το αντικείμενο «παίκτης» την ώρα της βολής. Έστω ότι την ονομάζουμε «περιοχή» και την ενημερώνει το αντικείμενο «παίκτης» ανάλογα με το που βρίσκεται την ώρα που επιχειρεί ο χρήστης το σουτ. Η μεταβλητή θα είναι ορατή από όλα τα αντικείμενα ώστε να τη λαμβάνει υπόψη του το αντικείμενο-μπάλα που μετράει το σκορ. Θα παίρνει τιμές με τον εξής απλό τρόπο:



Η περιοχή κάτω από τη στεφάνη έχει χρώμα μπορντό ή λευκό. Αν λοιπόν το αντικείμενο «παίκτης» αγγίζει αυτά τα χρώματα τότε ορίζουμε τη τιμή της μεταβλητής «περιοχή» σε 2. Διαφορετικά καταλαβαίνουμε ότι βρίσκεται έξω από τη περιοχή των δίποντων, οπότε και ορίζουμε τη μεταβλητή «περιοχή» σε 3 χρησιμοποιώντας την εντολή **όρισε το...σε...**

Η μεταβλητή «σκορ» ελέγχεται και τροποποιείται μόνο από το αντικείμενο «μπάλα». Αρχικά το σκορ είναι μηδέν αφού δεν έχει μπει κανένα καλάθι.

Έπειτα, κάθε φορά που το αντικείμενο μπάλα αγγίζει το αντικείμενο «στεφάνη», πρέπει το αντικείμενο «μπάλα» να ελέγχει τη μεταβλητή «περιοχή». Αν η μεταβλητή αυτή έχει τιμή 2 τότε πρέπει να αυξήσει τη μεταβλητή «σκορ» κατά 2. Αυτό το πετυχαίνουμε με την εντολή **άλλαξε σκορ κατά 2**. Διαφορετικά, η μεταβλητή «περιοχή» θα έχει τιμή 3, καθώς μπορεί να πάρει μόνο αυτές τις δύο τιμές.

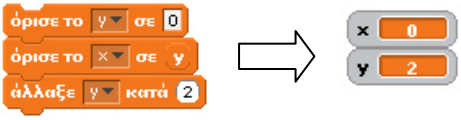
Μήπως όμως ο προγραμματιστής μας μπορούσε να κάνει κάτι πιο αποδοτικό; Μελετήστε προσεκτικά την παρακάτω εναλλακτική:

Αυτό ήταν ακόμη ένα παράδειγμα στο οποίο οι μεταβλητές χρησιμοποιήθηκαν για την έμμεση επικοινωνία μεταξύ αντικειμένων.

14.1.3 Γενικές παρατηρήσεις για τις μεταβλητές

Υπάρχουν δυο σημαντικά στοιχεία που πρέπει να αποσαφηνίσουμε για τις μεταβλητές στον προγραμματισμό.

Α) Πρώτον, οι μεταβλητές που χρησιμοποιούμε στον προγραμματισμό έχουν διαφορετική έννοια από αυτές που χρησιμοποιούμε στα μαθηματικά. Στον προγραμματισμό, το περιεχόμενο μιας μεταβλητής αλλάζει σύμφωνα με τη σειρά εκτέλεσης των εντολών ενός έργου-προγράμματος. Αυτό σημαίνει ότι η τιμή που θα είναι αποθηκευμένη στη μεταβλητή θα είναι η τελευταία τιμή την οποία εκχωρήσαμε σε αυτή. Αν για παράδειγμα έχουμε τις μεταβλητές «x» και «y», και αρχικά αποθηκεύσουμε στη μεταβλητή «x» το περιεχόμενο της μεταβλητής «y» και στη συνέχει αλλάξουμε την τιμή της μεταβλητής «y», αυτό δε σημαίνει ότι θα αλλάξει και η τιμή της μεταβλητής «x».



Στο πρόγραμμα της παραπάνω εικόνας οι μεταβλητές θα πάρουν τις εξής τιμές μετά από την εκτέλεση κάθε εντολής:

Όρισε το y σε μηδέν	y=0, x=κενό
Όρισε το x σε y	y=0, x=0
Άλλαξε y κατά 2	y=2, x=0

Δηλαδή το **όρισε το x σε y** ΔΕΝ σημαίνει εξίσωση των δυο τιμών αλλά ότι απλά θα βάλουμε ως τιμή του x, την τιμή του y τη στιγμή που εκτελείται η συγκεκριμένη εντολή. Αν στη συνέχεια η τιμή του y αλλάξει, η τιμή του x δε θα επηρεαστεί.

Β) Δεύτερον, τα λευκά κουτάκια που περιέχουν οι εντολές των μεταβλητών, μπορούν να περιέχουν άλλες μεταβλητές.

Έτσι για παράδειγμα μπορούμε να αναθέσουμε μια σύνθετη μαθηματική έκφραση ως τιμή μιας μεταβλητής π.χ.

Σε αυτές τις αναθέσεις τιμών, υπολογίζεται ο τύπος που βρίσκεται στο δεύτερο πεδίο της εντολής και το νούμερο που θα βρεθεί γίνεται η τιμή της μεταβλητής. Είναι πολύ ενδιαφέρον ότι μπορούμε να χρησιμοποιήσουμε σε μια τέτοια έκφραση την ίδια τη μεταβλητή στην οποία αναθέτουμε την τιμή! Π.χ.:

Στο τέλος του προηγούμενου σεναρίου το x θα πάρει την τιμή 2. Το Scratch δε βλέπει τη συγκεκριμένη έκφραση ως λανθασμένη (παρότι φαίνεται παράξενη), καθώς υπολογίζει πρώτα την τιμή της μαθηματικής έκφρασης βάσει της προηγούμενης τιμής του x και στη συνέχεια αναθέτει την νέα υπολογισμένη τιμή, στο x. Φυσικά, η συγκεκριμένη έκφραση είναι περιττή καθώς μπορεί να αντικατασταθεί από την εντολή **άλλαξε...κατά...** Θα δούμε όμως αργότερα παραδείγματα που κάτι ανάλογο είναι απαραίτητο.

14.2 Λίστες

14.2.1 Τι είναι οι λίστες;

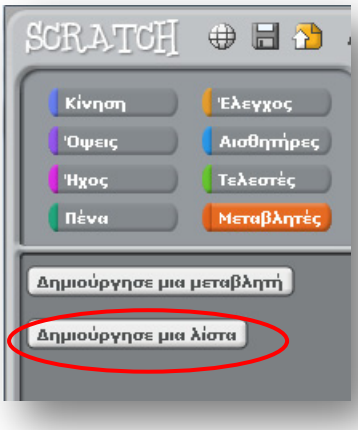
Πολλές φορές στην καθημερινή μας ζωή, χωρίς να το συνειδητοποιούμε, χρησιμοποιούμε λίστες. Τέτοια παραδείγματα είναι η λίστα του super market η οποία είναι ένας κατάλογος αντικειμένων που θέλουμε να αγοράσουμε. Ένα άλλο παράδειγμα είναι ένας τηλεφωνικός κατάλογος, ο οποίος περιλαμβάνει το

όνομα και το τηλέφωνο των κατοίκων μιας περιοχής, όπως επίσης και η κατάσταση με τα ονόματα και τους βαθμούς που αναρτάται στο σχολείο μετά τις τελικές εξετάσεις της πληροφορικής. **Η *λίστα* είναι μία συλλογή αντικειμένων με κοινά χαρακτηριστικά ή κοινό σκοπό**. Για παράδειγμα, ο κοινός σκοπός των αντικειμένων της λίστας του supermarket είναι να αγοράσουμε όλα τα υλικά που χρειαζόμαστε για μία συνταγή μαγειρικής. Όμοια, το κοινό χαρακτηριστικό της λίστας με τα ονόματα και τις βαθμολογίες των μαθητών είναι ότι αφορούν το ίδιο μάθημα, δηλαδή αυτό της πληροφορικής.

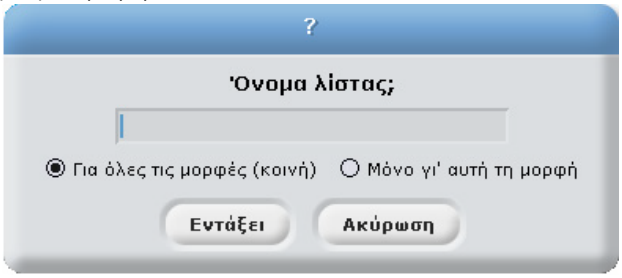
Αντίστοιχα και στον προγραμματισμό συναντάμε συχνά περιπτώσεις που οι λίστες είναι χρήσιμες για την υλοποίηση ενός προγράμματος. Σκεφτείτε για παράδειγμα ότι θέλουμε να δημιουργήσουμε μια εφαρμογή αγώνων ταχύτητας (Formula 1) και θα πρέπει ο χρήστης να μπορεί να διαλέξει πιο όχημα θα οδηγήσει μέσα από μια λίστα από 20 διαφορετικά μονοθέσια! Αντίστοιχα, αναλογιστείτε πόσο μεγάλο ενδιαφέρον θα είχε για το χρήστη να διαλέγει ακόμα και τις πίστες στις οποίες θέλει να αγωνιστεί οι οποίες θα βρίσκονται αποθηκευμένες σε μια δεύτερη λίστα παρόμοια με αυτή των μονοθέσιων. Το σίγουρο είναι ότι με χρήση των λιστών μπορούμε να δημιουργήσουμε πάρα πολλά παιχνίδια από πολύ απλά μέχρι αρκετά πολύπλοκα, παιχνίδια που συναντούμε στην καθημερινότητά μας και δεν φανταζόμασταν ποτέ ότι θα μπορούσαμε να γίνουμε οι δημιουργοί τους!

14.2.2 Η λίστα στο Scratch

Πως θα μπορούσαμε όμως στο Scratch να δημιουργήσουμε μία λίστα; Παρατηρούμε ότι στην παλέτα «Μεταβλητές» υπάρχει η επιλογή «Δημιούργησε μια λίστα».



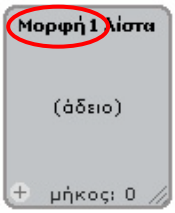
Πατώντας την επιλογή αυτή, ανοίγει ένα παράθυρο που έχει την ίδια μορφή με αυτό που εμφανίζεται όταν δημιουργούμε μία μεταβλητή.



Στο παράθυρο αυτό προσδιορίζουμε το όνομα που θέλουμε να δώσουμε στη λίστα και καθορίζουμε αν η λίστα αυτή θα είναι κοινή για όλα τα αντικείμενα ή μόνο για κάποιο, όπως ακριβώς και στις μεταβλητές. Η διαφορά των κοινών λιστών με τις λίστες που αφορούν ένα μόνο αντικείμενο μπορεί να φανεί καλύτερα μέσα ένα παράδειγμα. Σκεφτείτε για παράδειγμα μια εφαρμογή, όπως το γνωστό σε όλους μας παιχνίδι «Scrabble». Έστω ότι η εφαρμογή έχει δημιουργηθεί για τέσσερεις χρήστες. Στο παιχνίδι υπάρχει μια λίστα που περιέχει όλα τα πιθανά γράμματα που μπορεί να τραβήξει κάθε παίχτης, η οποία πρέπει να είναι κοινή, δηλαδή έχει δημιουργηθεί με την επιλογή «Για όλες τις μορφές (κοινή)», αφού όλοι οι χρήστες πρέπει να μπορούν να επέμβουν πάνω της, δηλαδή να διαλέγουν γράμματα από αυτή. Ο κάθε παίχτης ξεχωριστά όμως πρέπει να κρατάει σε μια προσωπική του λίστα τα γράμματα που έχει επιλέξει σε κάθε γύρο. Επομένως είναι αναγκαία η δημιουργία τεσσάρων ακόμη λιστών οι οποίες θα δημιουργηθούν με την επιλογή «Μόνο γι' αυτή τη μορφή» αφού θα αφορούν κάθε χρήστη αποκλειστικά.

Πατάμε «Εντάξει» στο προηγούμενο παράθυρο και η λίστα μας έχει δημιουργηθεί. Παρατηρήστε ότι εμφανίζεται ένα μικρό ορθογώνιο για τη λίστα στην οθόνη του Scratch, όπως φαίνεται στην παρακάτω εικόνα.

Αν η λίστα δημιουργήθηκε με το ίδιο όνομα αλλά με την επιλογή «Μόνο γι' αυτή τη μορφή» τότε εμφανίζεται και το όνομα του αντικειμένου στον τίτλο του παραθύρου, όπως φαίνεται στην επόμενη εικόνα.



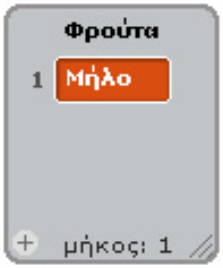
Παρατηρούμε ότι αμέσως μετά τη δημιουργία της λίστας, στην παλέτα «Μεταβλητές» εμφανίζονται εντολές μέσω των οποίων μπορούμε να επέμβουμε στη λίστα μας (όπως συνέβη με τις μεταβλητές).

Ας συζητήσουμε τη χρήση των εντολών της λίστας μέσα από ρεαλιστικά παραδείγματα.

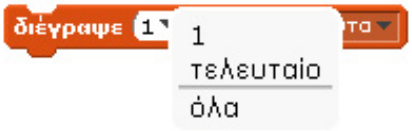
Η πρώτη εντολή που φαίνεται είναι η **πρόσθεσε [αντικείμενο] στο [λίστα]**. Με την εντολή αυτή, μπορούμε να προσθέσουμε ένα αντικείμενο στη λίστα μας, το οποίο προστίθεται στην τελευταία θέση της λίστας. Σκεφτείτε, για παράδειγμα, μια παραλλαγή της εφαρμογής «φιδάκι» όπου το φίδι μας

δεν τρέφεται με τελίτσες αλλά με φρούτα! Έστω ότι θέλουμε να ξέρουμε κάθε στιγμή ποια φρούτα έχει ήδη φάει το φίδι κρατώντας τα σε μια λίστα. Τότε σε κάθε διαφορετικό φρούτο της εφαρμογής μας (θεωρώντας ότι έχουμε δημιουργήσει τόσα αντικείμενα όσα είναι και τα φρούτα που θέλουμε στο παιχνίδι), πρέπει να δημιουργήσουμε μια συνθήκη που αφορά το αν αγγίζει το φρούτο το αντικείμενο-φίδι. Αν συνθήκη γίνεται αληθής και τότε θέλουμε το όνομα του αντικειμένου να προστίθεται στη λίστα (την οποία ενδεικτικά μπορούμε να ονομάσουμε «Φρούτα»). Το παρακάτω σενάριο δείχνει την ομάδα των εντολών που προσθέτουν το όνομα «μήλο» στη λίστα «Φρούτα» αν αυτό αγγίξει το αντικείμενο «φίδι».

Μετά την εκτέλεση της εντολής **πρόσθεσε [Μήλο] στο [Φρούτα]** η λίστα των φρούτων θα έχει τη μορφή της επόμενης εικόνας.



Συνεχίζουμε με τη δεύτερη εντολή η οποία είναι η **διέγραψε [1] από το [λίστα]**. Με την εντολή αυτή μπορούμε να διαγράψουμε το πρώτο στοιχείο της λίστας μας. Παρατηρούμε όμως ότι στην επιλογή 1 υπάρχει ένα βελάκι το οποίο αν το πατήσουμε μας δίνει τρεις επιλογές σχετικά με το τι θέλουμε να διαγράψουμε από τη λίστα μας.



Με τη επιλογή «1» διαγράφουμε το πρώτο στοιχείο της λίστας μας, με την επιλογή «τελευταίο» όπως είναι φανερό διαγράφουμε το τελευταίο στοιχείο της, ενώ με την επιλογή «όλα» διαγράφουμε όλα τα αντικείμενά της. Μπορούμε επίσης κάνοντας κλικ στο «1» και όχι στο βελάκι να δώσουμε εμείς πιο αντικείμενο επιθυμούμε να διαγραφεί από τη λίστα π.χ. μπορούμε να εκτελέσουμε την εντολή **διέγραψε [4] από το [λίστα]**. Σκεφτείτε για παράδειγμα την προηγούμενη εφαρμογή με το φιδάκι που τρώει φρούτα. Έστω ότι ο πρώτος χρήστης έχασε όλες τις ζωές του και είναι η σειρά μας να παίξουμε. Αν εκκινήσουμε το πρόγραμμά μας τότε η λίστα «Φρούτα» θα περιέχει αρχικά όλα τα φρούτα που ο προηγούμενος χρήστης είχε φάει πριν χάσει. Αφού δε θέλουμε κάτι τέτοιο, είναι λογικό στην αρχή του προγράμματος να διαγράφουμε όλα τα περιεχόμενα της λίστας.

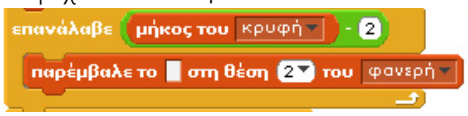
Γενικά είναι μια καλή στρατηγική κάθε φορά που χρησιμοποιούμε λίστες στην εφαρμογή μας να μην ξεχνάμε να διαγράφουμε όλα τα στοιχεία τους με το ξεκίνημα του προγράμματος.

Η τρίτη κατά σειρά εντολή της λίστας είναι η **παρέμβαλε το [αντικείμενο] στη θέση [1] του [λίστα]**. Αυτή η εντολή μας δίνει τη δυνατότητα να προσθέσουμε ένα αντικείμενο στη λίστα μας, σε όποια

θέση επιθυμούμε και όχι αναγκαστικά στη τελευταία θέση όπως κάνει η εντολή **πρόσθεσε [αντικείμενο] στο [λίστα]**. Η επιλογή της θέσης στην οποία θα τοποθετηθεί το αντικείμενο γίνετε κατά τον ίδιο τρόπο με την επιλογή της θέσης του αντικειμένου που θέλουμε να διαγράψουμε, όπως εξηγήσαμε στη προηγούμενη εντολή. Η μόνη διαφορά είναι ότι εδώ στη θέση της επιλογής «όλα» που υπήρχε στη διαγραφή, έχουμε την επιλογή «οποιοδήποτε» με την οποία το αντικείμενο προστίθεται τυχαία σε κάποια θέση της λίστας.

Έστω ότι θέλουμε να δημιουργήσουμε το πολύ γνωστό παιχνίδι κρεμάλα! Πως θα μας βοηθούσαν οι λίστες στην υλοποίησή τους; Μια λίστα θα μπορούσε να κρατούσε τα γράμματα της κρυμμένης λέξης, και μια δεύτερη λίστα θα μπορούσε να κρατάει τα γράμματα που δίνει ο χρήστης και είναι πετυχημένα στην προσπάθειά του να ανακαλύψει την κρυμμένη λέξη. Προφανώς, στη δεύτερη λίστα αρχικά θα εμφανίζονται το πρώτο και το τελευταίο γράμμα της λέξης. Τα δύο γράμματα θα εμφανίζονται στη πρώτη και στη τελευταία θέση της αντίστοιχης λίστας. Ανάμεσα στα δύο γράμματα πρέπει να παρεμβάλουμε τόσα κενά όσα είναι τα υπόλοιπα γράμματα της λέξης

έτσι ώστε ο δεύτερος παίχτης να έχει μια αντίληψη της λέξης που ψάχνει. Ένα σενάριο που κάνει κάτι τέτοιο:



Για να μπορέσουμε να γεμίσουμε τη φανερή λίστα μας με τον κατάλληλο αριθμό κενών, χρησιμοποιούμε όπως βλέπουμε μια ακόμα εντολή των λιστών την **μήκος του [λίστα]**. Αυτή η εντολή μας δίνει τη δυνατότητα να γνωρίζουμε πόσα στοιχεία είναι αποθηκευμένα στη λίστα μας κάθε στιγμή. Στο συγκεκριμένο παράδειγμα θέλουμε να γεμίσουμε τη φανερή λίστα με τόσα κενά όσα αντιστοιχούν στο συνολικό μήκος της κρυφής λίστας μείον 2 για τα γράμματα που θα φαίνονται κανονικά. Με τον τρόπο αυτό πετυχαίνουμε τις επιθυμητές επαναλήψεις. Παρατηρείστε ότι παρεμβάλουμε τα κενά επαναλαμβανόμενα στη θέση 2 της φανερής λίστας, δηλαδή ανάμεσα στο πρώτο και το τελευταίο γράμμα. Κάθε νέο κενό που προστίθεται σε κάθε επανάληψη μπαίνει στη δεύτερη θέση της λίστας «σπρώχνοντας» ουσιαστικά τα υπόλοιπα κενά και το τελευταίο γράμμα προς τα κάτω.

Η πέμπτη εντολή που συναντούμε στην παλέτα των μεταβλητών είναι η **αντικατέστησε στοιχείο [1] του [λίστα] με [αντικείμενο]**. Με την εντολή αυτή μπορούμε να αντικαταστήσουμε ένα στοιχείο της λίστας μας με ένα άλλο αντικείμενο. Η χρήση της είναι παρόμοια με την εντολή **παρέμβαλε το [αντικείμενο] στη θέση [1] του [λίστα]** που είδαμε πριν, με τη διαφορά ότι εδώ το νέο μας αντικείμενο δεν μπαίνει στη λίστα μετακινώντας άλλα στοιχεία αλλά αντικαθιστά το αντικείμενο που βρισκόταν στη θέση που προσδιορίζουμε στην εντολή. Οι επιλογές που έχουμε για το ποιο στοιχείο θέλουμε να αντικαταστήσει το καινούριο αντικείμενο είναι ίδιες με πριν. Μπορούμε δηλαδή να αντικαταστήσουμε το πρώτο αντικείμενο της λίστας, το τελευταίο της ή οποιοδήποτε άλλο. Με την επιλογή «οποιοδήποτε» η αντικατάσταση είναι τυχαία ενώ αν θέλουμε να αντικαταστήσουμε ένα συγκεκριμένο αντικείμενο αλλά όχι το πρώτο ή το τελευταίο, αρκεί να κάνουμε κλικ στον αριθμό 1 και να προσδιορίσουμε τον αριθμό της θέσης του στοιχείου προς αντικατάσταση.

Για να καταλάβουμε τη λειτουργία όμως αυτής της εντολής ας σκεφτούμε ξανά το προηγούμενο παράδειγμα της κρεμάλας. Έστω ότι ήρθε η ώρα του δεύτερου χρήστη να παίξει μαντεύοντας γράμματα της λέξης που έβαλε ο πρώτος. Αν το γράμμα που πληκτρολογεί ο παίχτης, ανήκει στη κρυφή λίστα, γεγονός που σημαίνει ότι είναι σωστή επιλογή, τότε πρέπει αυτό το γράμμα να αντικαταστήσει το κενό στην αντίστοιχη θέση στη φανερή λίστα. Για να ελέγξουμε αν το γράμμα που έδωσε ο χρήστης ανήκει στη κρυφή λίστα, θα χρησιμοποιήσουμε την εντολή των λιστών **[λίστα] περιέχει [αντικείμενο]**. Αυτή η εντολή πραγματοποιεί μια αναζήτηση στη λίστα μας και επιστρέφει «αληθές» αν το αντικείμενο υπάρχει ή «ψευδές» αν το αντικείμενο δεν υπάρχει σε αυτή.

Αυτό όμως από μόνο του δεν είναι αρκετό. Αν τελικά το γράμμα ανήκει στη κρυφή λίστα, πρέπει να ανακαλύψουμε σε ποια θέση ή θέσεις είναι ακριβώς, ώστε να αντικαταστήσουμε τα κενά με το γράμμα το οποίο πληκτρολογήθηκε. Αυτό δυστυχώς θα μας αναγκάσει να ελέγξουμε όλα τα στοιχεία της λίστας ένα προς ένα. Για να κάνουμε τον έλεγχο θα χρειαστούμε μια επανάληψη και τη χρήση της εντολής **στοιχείο [1] του [λίστα]**. Η εντολή αυτή επιστρέφει το όνομα του αντικειμένου της λίστας που βρίσκεται στη θέση που προσδιορίζουμε στην εντολή. Αν δηλαδή εκτελεστεί η εντολή **στοιχείο [3] του [λίστα]** τότε αυτή επιστρέφει το τρίτο αντικείμενο της λίστας. Το κομμάτι του σεναρίου της κρεμάλας που πραγματοποιεί αυτά που αναφέραμε:

Η μεταβλητή **απάντηση** περιέχει την απόκριση του χρήστη σε ερώτημα για το επόμενο γράμμα με τη χρήση της εντολής **ρώτησε...και περίμενε**. Στο σενάριο αυτό εξετάζουμε αρχικά αν υπάρχει το γράμμα μέσα στη λίστα. Εφόσον υπάρχει, τότε με μια επανάληψη ελέγχουμε όλα τα γράμματα της κρυφής λίστας ξεχωριστά για να βρούμε τις θέσεις του γράμματος. Κάθε φορά που βρίσκουμε μια θέση στην κρυφή λίστα στην οποία υπάρχει το γράμμα, αντικαθιστούμε στην αντίστοιχη θέση το κενό της φανερής λίστας!

Λείπει όμως κάτι από το σενάριο. Τι είναι το x; Το x είναι μια μεταβλητή που έχουμε δημιουργήσει και της οποίας αρχική τιμή είναι το 2. Παρατηρήστε ότι λόγω αυτής της τιμής ο πρώτος έλεγχος στη κρυφή λίστα θα γίνει για το δεύτερο στοιχείο της. Η μεταβλητή x είναι ουσιαστικά ένας μεταβλητός δείκτης στη λίστα. Στην πρώτη επανάληψη παίρνει την τιμή 2, στην επόμενη 3 κοκ. Τέτοιες μεταβλητές, που αυξάνονται με τις επαναλήψεις ονομάζονται **μετρητές** και είναι ιδιαίτερα χρήσιμες στον προγραμματισμό.

Ο αριθμός των επαναλήψεων είναι ίσος είναι το μήκος της κρυφής λίστας μείον 2 αφού το πρώτο και το τελευταίο γράμμα δεν χρειάζεται να τα ελέγχουμε.

14.2.3 Μεταβλητές vs. Λίστες

Ίσως, να μην είναι εντελώς ξεκάθαρο πότε πρέπει να χρησιμοποιούμε μια μεταβλητή και πότε μια λίστα. Από τις παραπάνω περιγραφές βλέπουμε ότι τα δύο αυτά στοιχεία έχουν αρκετές ομοιότητες μεταξύ τους. Παρατηρούμε ότι και τα δύο αποτελούν δομές αποθήκευσης. Παρόλα αυτά, δεν πρέπει να συγχέουμε τη χρήση και το λόγο ύπαρξης του καθενός.

Αρχικά, πρέπει να έχουμε ξεκαθαρίσει στο μυαλό μας ότι οι μεταβλητές χρησιμοποιούνται για την απόδοση τιμής σε ένα

χαρακτηριστικό ενός αντικειμένου, για παράδειγμα το «χρώμα_ματιών» του αντικειμένου «Κατερίνα». Ακόμη, οι μεταβλητές μπορούν να χρησιμοποιηθούν ως θέσεις προσωρινής αποθήκευσης για χαρακτηριστικά που η τιμή τους μπορεί να αλλάξει κατά την εκτέλεση ενός προγράμματος. Ένα τέτοιο παράδειγμα θα μπορούσε να είναι μεταβλητές που κρατούν το σκορ του αγώνα για τις ομάδες μας ή η μεταβλητή που κρατάει τις ελεύθερες θέσεις σε ένα parking για να δούμε αν υπάρχει χώρος για άλλα αυτοκίνητα.

Από την άλλη πλευρά, η λίστα μας προσφέρει τη δυνατότητα αποθήκευσης δεδομένων, χωρίς να απαιτείται να γνωρίζουμε εξ αρχής τον αριθμό των δεδομένων που θέλουμε να κρατήσουμε. Για παράδειγμα, σε έναν τηλεφωνικό κατάλογο δε γνωρίζουμε από την αρχή τον αριθμό των τηλεφώνων που θα καταχωρηθούν, οπότε για κάθε νέο αριθμό που θέλουμε να έχουμε στον κατάλογο μας, προσθέτουμε ακόμη μία θέση στη λίστα. Επίσης, μία ιδιότητα της λίστας που την καθιστά ιδιαίτερα χρήσιμη είναι ότι μπορεί να διαχειρίζεται μεγάλο πλήθος δεδομένων. Ας σκεφτούμε το προηγούμενο παράδειγμα, όπου ο τηλεφωνικός κατάλογος μπορεί να είναι κάποιας μεγάλης πόλης, για την οποία ο αριθμός τηλεφώνων των κατοίκων της είναι από την αρχή γνωστός. Αν θέλαμε να κρατήσουμε το κάθε τηλέφωνο σε μία ξεχωριστή μεταβλητή θα ήταν πολύ δύσκολο να δημιουργήσουμε και να διαχειριστούμε τόσες πολλές μεταβλητές. Επιπλέον. η λίστα μας προσφέρει τη δυνατότητα της ταξινόμησης των δεδομένων αλλά και τη δυνατότητα της αναζήτησης. Αυτές οι λειτουργίες θα ήταν χρήσιμες για παράδειγμα σε μία εφαρμογή για ένα κατάστημα ενοικίασης DVD ταινιών. Ο πωλητής θα μπορούσε να χειρίζεται το πρόγραμμα αυτό για να κάνει αναζήτηση στη λίστα των ταινιών που διαθέτει το κατάστημα έτσι ώστε να βρει αν υπάρχει η ταινία που ζήτησε κάποιος πελάτης. Ακόμη, θα έχει τη δυνατότητα να ταξινομεί τη λίστα με τις ταινίες με αλφαβητική σειρά.

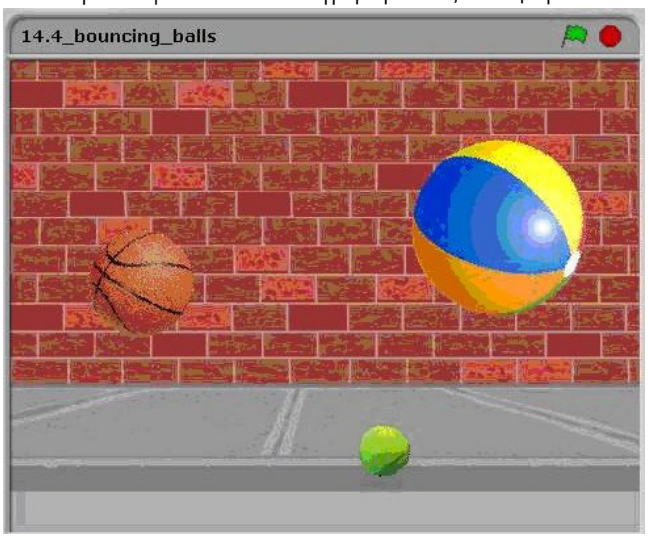
Ευτυχώς που σε αυτό το βιβλίο επιδιώκουμε την κατανόηση μέσα από παραδείγματα, και κυρίως παραδείγματα-παιχνίδια ☺

14.3 Παραδείγματα

14.3.1 Μεταβλητές

Παράδειγμα Bouncing Balls: Στη σκηνή υπάρχουν τρείς μπάλες και κάθε μία έχει διαφορετικό μέγεθος από τις άλλες. Όλες οι μπάλες χοροπηδούν ταυτόχρονα προς όλες τις κατευθύνσεις. Κάθε φορά που πετυχαίνεις μία μπάλα με το ποντίκι σου, το σκορ σου αυξάνεται. Όσο πιο μικρή η μπάλα, τόσο περισσότεροι και οι βαθμοί σου! Πόσο καλό στόχο έχεις;

Αντικείμενα: Αρχικά πρέπει να ανιχνεύσουμε ποια θα είναι τα αντικείμενά μας. Παρατηρούμε ότι πρέπει να υπάρχουν τρεις μπάλες οι οποίες θα κινούνται συνεχώς, επομένως και τρία αντικείμενα. Θα διαλέξουμε μπάλες με διαφορετικό μέγεθος. Μπορούμε να χρησιμοποιήσουμε μία μπάλα θαλάσσης ως τη μεγαλύτερη, μία μπάλα του μπάσκετ ως μεσαίου μεγέθους μπάλα και μία μπάλα του τένις ως τη μικρότερη μπάλα. Τα αντίστοιχα ονόματα των αντικειμένων θα είναι μπάλα_θαλάσσης, μπάλα_μπάσκετ και μπάλα_τένις. Όλες τις μπάλες μπορείτε να τις βρείτε μέσω διαδικτύου. Για σκηνικό, επιλέξαμε το υπόβαθρο «brick-wall1» της βιβλιοθήκης σκηνικών του Scratch για να φαίνεται ότι κυνηγάμε μπάλες στο δρόμο.



Ποιες είναι οι συμπεριφορές των αντικειμένων μας;

Μπάλα του μπάσκετ: Η μπάλα του μπάσκετ θα κινείται από την έναρξη του παιχνιδιού μέχρις ότου ο παίκτης να τερματίσει ο ίδιος το παιχνίδι. Όταν φτάσει στα όρια της σκηνής, θα πρέπει να συνεχίσει να κινείται αλλάζοντας κατεύθυνση.

Παρότι έχουμε αντιμετωπίσει πολλά παραπλήσια προβλήματα στα προηγούμενα κεφάλαια, ας θυμηθούμε ξανά όλες τις απαιτούμενες εντολές. Για την κίνηση της μπάλας θα χρησιμοποιήσουμε την εντολή **κινήσου...βήματα**. Εφόσον θέλουμε να μη σταματάει να κινείται, θα πρέπει η εντολή αυτή να συνδυαστεί με την εντολή **για πάντα**. Επίσης, για να αναπηδήσει και να αλλάξει κατεύθυνση στα όρια θα πρέπει να χρησιμοποιήσουμε την εντολή **εάν στα όρια, αναπήδησε**. Για να γίνει το παιχνίδι ενδιαφέρον και η θέση της μπάλας να μην είναι απολύτως προβλέψιμη, μπορούμε να χρησιμοποιήσουμε τον τελεστή **τυχαία επιλογή από...μέχρι...** για να προσδώσουμε μια τυχαία αρχική κατεύθυνση στο αντικείμενο και στη συνέχεια με στόχο τα βήματα της μπάλας να μην είναι σταθερά σε κάθε επανάληψη. Το σενάριο της κίνησης που προκύπτει για την μπάλα:

Μπάλα του τένις και μπάλα θαλάσσης: Οι μπάλες αυτές θα πρέπει να κινούνται πάνω στη σκηνή με τον ίδιο τρόπο που κινείται και η μπάλα του μπάσκετ ενώ όταν ο παίκτης πετύχει με το ποντίκι του κάποια από τις μπάλες αυτές, θα πρέπει να αυξάνεται το σκορ του με τη διαφορά όμως ότι για την μπάλα του τένις οι πόντοι του θα είναι περισσότεροι αφού είναι και η πιο μικρή.

Μη ξεχνάτε ότι για να κάνετε τις μπάλες να ξεκινήσουν από διαφορετικές κατευθύνσεις μπορείτε επίσης να στρίψετε την

μπλε γραμμή κατεύθυνσης που βρίσκεται πάνω στο κάθε αντικείμενο όπως φαίνεται στην παρακάτω εικόνα.

Πόντοι: Έχοντας ολοκληρώσει την κίνηση κάθε μπάλας, ήρθε η στιγμή να υλοποιήσουμε τον τρόπο με τον οποίο ο παίκτης θα συγκεντρώνει πόντους. Οι πόντοι θα πρέπει να αποθηκεύονται σε μία μεταβλητή. Συνεπώς δημιουργούμε μια νέα μεταβλητή με το όνομα «πόντοι» και επιλέγουμε να είναι «Κοινή για όλες τις μορφές» αφού θα πρέπει να μπορεί να τροποποιηθεί από όλες τις μπάλες, δηλαδή από όλα τα αντικείμενα του έργου μας.

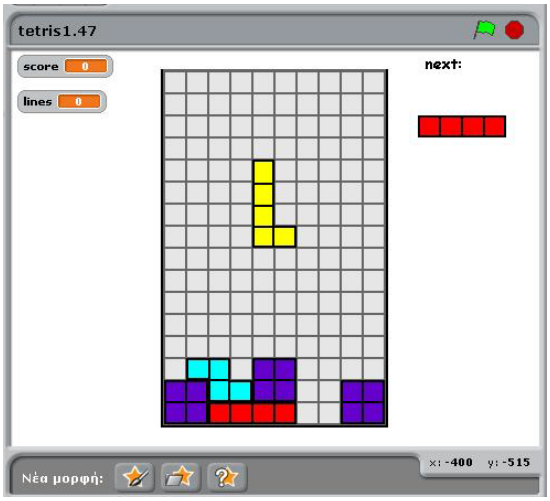
Κάθε φορά που ξεκινάει το πρόγραμμα η μεταβλητή πρέπει να έχει μηδενική τιμή για να είναι αξιόπιστο το σκορ του παίκτη! Άρα στα σενάρια για τις μπάλες προσθέτουμε την εντολή **όρισε το [score] σε [0]**, πριν τις εντολές τις κίνησης που είδαμε παραπάνω.

Κάθε φορά που γίνει κλικ πάνω σε μία μπάλα, οι πόντοι που συλλέγει ο παίκτης πρέπει να αυξάνονται. Αν πετύχει την μπάλα θαλάσσης κερδίζει 1 πόντο, την μπάλα του μπάσκετ 2 και την μπάλα του τένις 4 πόντους. Επομένως, πρέπει να χρησιμοποιήσουμε την εντολή **όταν στο...γίνει κλικ** για κάθε αντικείμενο, με στόχο να αλλάζουμε την τιμή της μεταβλητής ανάλογα με το ποια μπάλα έχει πετύχει ο παίκτης.

Για την αύξηση της τιμής της μεταβλητής θα χρειαστούμε την εντολή **άλλαξε [πόντοι] κατά [...]** , όπου ο αριθμός αύξησης εξαρτάται από το είδος της μπάλας (1 για την μπάλα θαλάσσης, 2 για την μπάλα του μπάσκετ και 4 για την μπάλα του τένις). Προσθέτουμε τα παρακάτω σενάρια στα αντικείμενα μας (ένα σε κάθε αντικείμενο) και το παιχνίδι μας έχει ολοκληρωθεί!!

[14_π05.sb]

Παράδειγμα μεταβλητών στο Tetris: Για όσους δε γνωρίζουν το Tetris, σκοπός του παιχνιδιού είναι να δημιουργούμε ολοκληρωμένες γραμμές χωρίς κενά, τοποθετώντας κατάλληλα τα τουβλάκια που μας έρχονται από το πάνω μέρος της οθόνης. Κάθε φορά που ολοκληρώνεται μια γραμμή, τότε διαγράφεται και όλα τα τουβλάκια μετακινούνται προς τα κάτω. Στόχος του παίχτη είναι να μην φτάσουν τα στοιβαγμένα τουβλάκια στο πάνω μέρος της οθόνης. Εδώ δεν θα αναλύσουμε την λειτουργία ολόκληρου του παιχνιδιού αλλά θα εστιάσουμε στον τρόπο με τον οποίο επιλέγουμε με τυχαίο τρόπο τα τουβλάκια που πέφτουν.

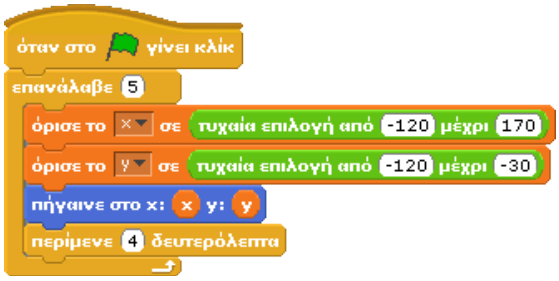


Από τη στιγμή που υπάρχουν 7 διαφορετικά τουβλάκια διαθέσιμα, προφανώς θα χρησιμοποιήσουμε την εντολή **τυχαία επιλογή από 1 μέχρι 7** ώστε να προκύψει ένας τυχαίος αριθμός από 1 έως 7. Ανάλογα με αυτό τον αριθμό θα στείλουμε μήνυμα στο κατάλληλο αντικείμενο ώστε να πάρει θέση και να αρχίσει να πέφτει μόλις έρθει η σειρά του. Από τη στιγμή που κάθε φορά που εκτελείται η εντολή **τυχαία επιλογή από 1 μέχρι 7** παράγει διαφορετικό αριθμό, για να κάνουμε ελέγχους με τον αριθμό που παράγει, πρέπει να αποθηκεύσουμε το αποτέλεσμά της σε μια μεταβλητή. Μη ξεχνάτε ότι κάθε φορά που τρέχει η **τυχαία επιλογή από 1 μέχρι 7** παράγει έναν διαφορετικό αριθμό. Έπειτα, η συγκεκριμένη μεταβλητή θα συμμετάσχει στους διάφορους ελέγχους που πρέπει να γίνουν. Μελετήστε τον παρακάτω μέρος του σεναρίου του βασικού αντικειμένου του Tetris:

Όλο το σενάριο βρίσκεται μέσα σε μια επανάληψη. Στη τελευταία γραμμή του σεναρίου δημιουργείται ένας τυχαίος αριθμός ο οποίος αποθηκεύεται στη μεταβλητή next, ενώ στην πρώτη γραμμή του σεναρίου, η μεταβλητή randomnum παίρνει την τιμή της μεταβλητής next και με βάση αυτή τη μεταβλητή στέλνεται το κατάλληλο μήνυμα. Δεν θα μπορούσαν να γίνουν οι έλεγχοι αν δεν αποθηκεύαμε τον τυχαίο αριθμό που παράχθηκε μέσα σε μια μεταβλητή και για αυτό η χρήση των μεταβλητών για την αποθήκευση τυχαίων αριθμών είναι πολύ συχνή.

Γιατί ο προγραμματιστής επέλεξε να φωλιάσει τις εντολές **εάν…αλλιώς** και δεν τις έβαλε απλώς τη μια κάτω από την άλλη; Αξιολογήστε την απόδοση των δυο αντίστοιχων προγραμμάτων. Πόσοι έλεγχοι πραγματοποιούνται κάθε φορά;

Αντίστοιχα, αν θέλαμε να μεταφέρετε ένα αντικείμενό μας μέσα σε ένα συγκεκριμένο τμήμα της οθόνης του Scratch με τυχαίο όμως τρόπο, θα δημιουργούσαμε ένα σενάριο σαν αυτό που φαίνεται στην επόμενη εικόνα.

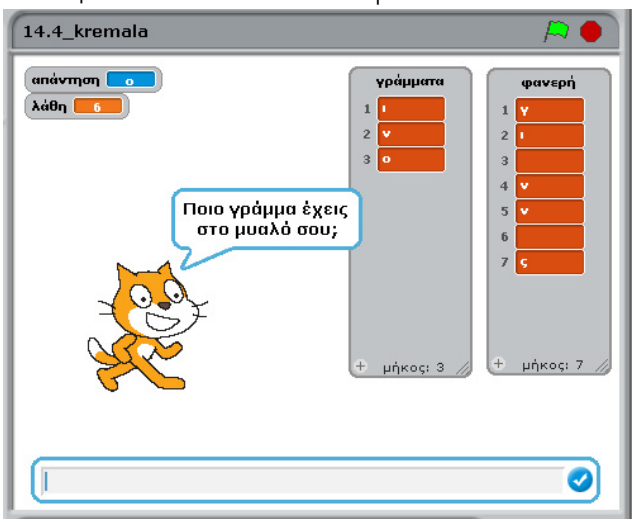


Στο παραπάνω σενάριο, τα άλλα αντικείμενα μπορούν να ελέγξουν τη νέα θέση του συγκεκριμένου αντικειμένου μέσω των μεταβλητών x και y.

Αν κουραστήκατε και θέλετε να παίξετε Tetris, μπορείτε να επισκεφτείτε τη διεύθυνση
http://scratch.mit.edu/projects/amyv/11292

14.3.2 Λίστες

Παράδειγμα «Κρεμάλα»: Τι θα λέγατε να παίξουμε μία κρεμάλα; Ας επαναλάβουμε τους κανόνες για να είμαστε σίγουροι ότι το παιχνίδι θα είναι δίκαιο! Στο παιχνίδι αυτό χρειαζόμαστε δύο παίκτες τουλάχιστον. Ο πρώτος παίκτης είναι αυτός που διαλέγει μια λέξη και φανερώνει μόνο το πρώτο και το τελευταίο γράμμα της καθώς και πόσες κενές θέσεις υπάρχουν ανάμεσά τους. Ο δεύτερος παίκτης ψάχνει να βρει τη λέξη προτείνοντας κάθε φορά κάποιο γράμμα. Όμως οι ευκαιρίες του δεν είναι απεριόριστες! Έτσι όταν τις εξαντλήσει, χάνει! Στην παρακάτω εικόνα, βλέπετε ένα στιγμιότυπο του προγράμματος, που ο πρώτος χρήστης έχει βάλει τη λέξη «γιάννης» και ο δεύτερος χρήστης έχει έως τώρα δοκιμάσει τα γράμματα ι,ν,ο και επιτρέπεται να κάνει άλλα 6 λάθη.



[14_p05.sb]

Αντικείμενα και Σκηνικά: Στο συγκεκριμένο παιχνίδι παρατηρούμε ότι δε χρειαζόμαστε παραπάνω από ένα αντικείμενο. Το αντικείμενο μας θα λειτουργεί ως παρουσιαστής του παιχνιδιού που ανακοινώνει στους παίκτες τους κανόνες. Στο παράδειγμά μας, θα αφήσουμε τον κ.Γατίδη ως παρουσιαστή.

Υλοποίηση του παιχνιδιού: Ας αναλύσουμε το παιχνίδι λίγο πιο προσεκτικά. Τι συμβαίνει στο παιχνίδι;

- ✓ Πραγματοποιείται η εισαγωγή μίας λέξης από τον πρώτο παίκτη.
- ✓ Αποκρύπτεται η λέξη αυτή ώστε ο δεύτερος παίκτης να μην τη γνωρίζει.

- ✓ Εμφανίζεται μια λίστα που περιέχει μόνο το αρχικό και τελευταίο γράμμα της λέξης με ενδιάμεσα κενά για τα γράμματα που λείπουν.
- ✓ Ο δεύτερος παίκτης αρχίζει να εισάγει γράμματα.
- ✓ Αν τα γράμματα αυτά υπάρχουν στη λέξη, προστίθενται στις σωστές θέσεις.
- ✓ Όλα τα γράμματα που εισάγει ο χρήστης αποθηκεύονται σε μια άλλη λίστα ώστε να γνωρίζει ποια έχει δοκιμάσει κάθε στιγμή.
- ✓ Το παιχνίδι συνεχίζεται είτε μέχρι είτε να βρει τη λέξη ο χρήστης, είτε να φτάσει το μέγιστο αριθμό λαθών που επιτρέπονται.

Όπως παρατηρείτε, το συγκεκριμένο έργο έχει μια ακολουθιακή δομή, στην οποία η μια ενέργεια διαδέχεται την επόμενη και συνεπώς δε θα χρειαστούμε πολλαπλά σενάρια αλλά μόνο 1! Μετά από πολλά κεφάλαια θα δημιουργήσουμε ένα παιχνίδι με ένα μόνο σενάριο!

Εισαγωγή λέξης από τον πρώτο παίκτη.

Επειδή όλο το παιχνίδι εκτυλίσσεται σε επίπεδο γραμμάτων, θα ζητήσουμε από το χρήστη να εισάγει τη λέξη προσδιορίζοντας τα γράμματά της ξεχωριστά. Ο μόνος τρόπος που έχουμε μάθει για την εισαγωγή δεδομένων από το χρήστη είναι μέσω της εντολής **ρώτησε...και περίμενε** και της μεταβλητής **απάντηση** από την παλέτα **Αισθητήρες**. Που όμως θα αποθηκεύσουμε τα γράμματα; Χρειαζόμαστε τη δημιουργία μιας λίστας την οποία θα ονομάσουμε «κρυφή» και στην οποία θα αποθηκεύουμε τα γράμματα της λέξης. Αυτή η λίστα δεν θα πρέπει να είναι ορατή στο χρήστη.

Αρχικά, κατανοούμε ότι εφόσον ο παίκτης θα δίνει τα γράμματα ξεχωριστά, θα χρειαστούμε κάποια εντολή επανάληψης έτσι ώστε να επαναλαμβάνεται η διαδικασία εισαγωγής γραμμάτων. Μέχρι πότε θα εκτελείτε η επανάληψη; Κάθε λέξη έχει διαφορετικό αριθμό γραμμάτων και συνεπώς δε μπορούμε να γνωρίζουμε εκ των προτέρων τον ακριβή αριθμό των επαναλήψεων. Μπορούμε όμως να δημιουργήσουμε εμείς μια συνθήκη τερματισμού, π.χ. να πούμε ότι όταν η λέξη ολοκληρωθεί, ο χρήστης μπορεί να εισάγει ως γράμμα το κενό, οπότε το πρόγραμμα αντιλαμβάνεται ότι τα γράμματα της λέξης τελείωσαν. Άρα από τη στιγμή που δεν γνωρίζουμε τον αριθμό των επαναλήψεων πρέπει να χρησιμοποιήσουμε την εντολή **επανέλαβε ώσπου...** από την παλέτα **Έλεγχος**. Στο εσωτερικό της επανάληψης αυτής θα χρειαστούμε την εντολή **ρώτησε...και περίμενε** και τη μεταβλητή **απάντηση** και κάθε γράμμα που προσδιορίζει ο χρήστης θα το εισάγουμε στην κρυφή λίστα μας μέσω της εντολής **πρόσθεσε [...] στο [κρυφή]**. Το παρακάτω σενάριο υλοποιεί αυτή τη διαδικασία:

Δυο παρατηρήσεις σχετικές με το προηγούμενο σενάριο:

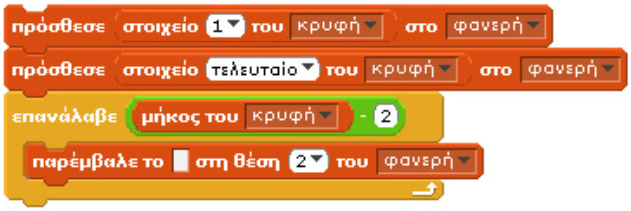
- ✓ Με βάση το προηγούμενο σενάριο, όταν ο παίκτης πατήσει το κενό, αυτό θα προστεθεί στη λίστα, κάτι που θέλουμε να αποφύγουμε αφού δεν αποτελεί μέρος της λέξης. Για αυτό αφαιρούμε από τη λίστα το τελευταίο στοιχείο που προστέθηκε με την εντολή **διέγραψε [τελευταίο] από το [κρυφή]**.
- ✓ Η συνθήκη της επανάληψης παίρνει τη μορφή «**[απάντηση] = [ ]**» με τη βοήθεια του τελεστή της ισότητας από την παλέτα Τελεστές. Το λευκό κουτάκι στον τελεστή φαίνεται άδειο αλλά εμείς έχουμε πατήσει το χαρακτήρα του κενού.

Από τη στιγμή που δεν υπάρχουν εντολές για την απόκρυψη και την εμφάνιση μιας λίστας στην οθόνη του Scratch, η απόκρυψη της κρυφής λίστας πρέπει να γίνει απενεργοποιώντας την αντίστοιχη μεταβλητή κατάστασης «κρυφή» που βρίσκεται στην παλέτα **Μεταβλητές**.

Δημιουργία λέξης με κενά

Ήρθε η στιγμή να προσπαθήσουμε να δημιουργήσουμε τη λέξη που θα εμφανίζεται στον δεύτερο παίκτη κα η οποία θα έχει ένα γράμμα στην αρχή και ένα στο τέλος της και κενά στις ενδιάμεσες θέσεις. Σκεφτόμενοι όπως και πριν, και αυτή η λέξη θα αποθηκευτεί σε μορφή λίστας έτσι ώστε να είμαστε σε θέση να αντικαταστήσουμε τα κενά της με τις επιλογές γραμμάτων του παίκτη. Θα δημιουργήσουμε λοιπόν μία λίστα με όνομα «φανερή».

Για την εισαγωγή στη λίστα «φανερή» του πρώτου και του τελευταίου στοιχείου της λίστας «κρυφή», θα χρειαστούμε τις εντολές **πρόσθεσε [...] στο [φανερή]**, **στοιχείο [τελευταίο] του [κρυφή]** και **στοιχείο [1] του [κρυφή]**. Οι δυο πρώτες εντολές του παρακάτω σεναρίου (που αποτελούν τη συνέχεια του προηγούμενου) προσθέτουν το πρώτο και τελευταίο γράμμα της κρυφής λίστας στη φανερή.



Τα στοιχεία αυτά θα εισαχθούν στην πρώτη και δεύτερη θέση της «φανερής» λίστας, ενώ δεν υπάρχουν προς το παρόν κενά στοιχεία για τα γράμματα που λείπουν. Πρέπει όμως να εισάγουμε αυτά τα κενά. Αυτό κάνει η επανάληψη που ακολουθεί στο προηγούμενο σενάριο. Βάζει όσα κενά χρειαζόμαστε μεταξύ των δυο γραμμάτων. Πόσα κενά χρειαζόμαστε; Όσο είναι

το πλήθος των γραμμάτων της λίστας «κρυφή» μείον 2, αφού έχουμε ήδη εισάγει δύο από τα γράμματα της λέξης. Το πλήθος των γραμμάτων της κρυφής λίστας θα το πάρουμε με την εντολή **μήκος του [κρυφή]**, ενώ η αφαίρεση θα πραγματοποιηθεί με τη βοήθεια του τελεστή αφαίρεσης **...– ....** Για την εισαγωγή των κενών χρησιμοποιήσαμε την εντολή **παρέμβαλε το [ ] στη θέση [2] του [φανερή]**. Με την εντολή αυτή ουσιαστικά παρεμβάλουμε κάθε φορά στη θέση 2 της φανερής λίστας ένα κενό μετατοπίζοντας τα στοιχεία που ακολουθούν προς τα κάτω και δημιουργώντας έτσι τα απαραίτητα κενά. Η λίστα «φανερή» θέλουμε να εμφανίζεται στον παίκτη, οπότε πρέπει να επιλέξουμε την εμφάνιση της από την αντίστοιχη μεταβλητή κατάστασης.

Εισαγωγή γραμμάτων από το δεύτερο παίκτη

Η συνέχεια του προγράμματός μας είναι μια επανάληψη. Ο δεύτερος παίκτης πρέπει να μπορεί να δίνει διαρκώς γράμματα και εμείς να εξετάζουμε τις επόμενες εναλλακτικές:

- ✓ το γράμμα να ανήκει μέσα στην κρυφή λίστα, οπότε πρέπει να το προσθέσουμε στη φανερή στις κατάλληλες θέσεις,
- ✓ το γράμμα να μην ανήκει μέσα στη κρυφή λίστα, οπότε πρέπει να σημειώσουμε ότι ο παίκτης έκανε ένα λάθος,
- ✓ η φανερή λίστα να μην περιέχει άλλα κενά, γεγονός που σημαίνει ότι ο δεύτερος παίκτης τα κατάφερε και βρήκε τη λέξη,
- ✓ ο χρήστης να φτάνει το όριο των λαθών που του επιτρέπονταν και συνεπώς το παιχνίδι να τερματίζεται.

Ας δούμε αυτές τις εναλλακτικές ξεχωριστά:

Α) Το γράμμα ανήκει μέσα στην κρυφή λίστα, οπότε πρέπει να το προσθέσουμε και στη φανερή λίστα

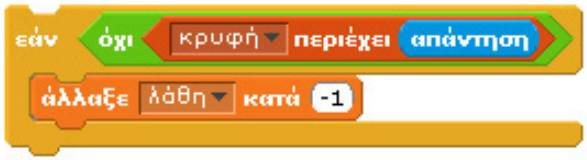
Εφόσον, πλέον, ο δεύτερος παίκτης μπορεί να δει τη λέξη με την οποία θα παίξει, πρέπει να μπορεί να εισάγει τα γράμματα που φαντάζεται ότι κρύβονται πίσω από τα κενά. Για να εισάγει ο δεύτερος παίκτης τα γράμματα θα χρησιμοποιήσουμε ξανά την εντολή **ρώτησε [ ] και περίμενε** σε συνδυασμό με τη μεταβλητή **απάντηση**.

Όταν ο παίκτης εισάγει ένα γράμμα, πρέπει να ελέγχεται αν υπάρχει μέσα στα γράμματα της κρυφής λίστας. Ο έλεγχος αυτός θα γίνει με τη βοήθεια της εντολής **[κρυφή] περιέχει [απάντηση]**. Εάν ο έλεγχος αυτός είναι αληθής, δηλαδή αν το γράμμα περιέχεται στη λέξη, τότε θα πρέπει να βρούμε σε ποια θέση της λέξης βρίσκεται. Για να βρούμε τη θέση/τις θέσεις, θα χρειαστούμε μια επανάληψη όπως αυτή που αναλύσαμε προηγουμένως στο κεφάλαιο. Στην επανάληψη θα εξετάσουμε αν οποιοδήποτε από τα ενδιάμεσα γράμματα της κρυφής λίστας ταιριάζει με την απάντηση του χρήστη. Αν ταιριάζει, θα αντικαταστήσουμε το κενό της φανερής λίστας, με την απάντηση του χρήστη, στη θέση που εντοπίσαμε το γράμμα στην κρυφή λίστα! Μελετήστε το παρακάτω σενάριο:

Θυμηθείτε ότι το x είναι ένας μια μεταβλητή-μετρητής για να ελέγχουμε διαφορετικές θέσεις στις λίστες και για αυτό πρέπει σε κάθε επανάληψη την αυξάνουμε.

Β) Το γράμμα δεν ανήκει μέσα στη κρυφή λίστα, οπότε πρέπει να σημειώσουμε ότι ο παίκτης έκανε ένα λάθος

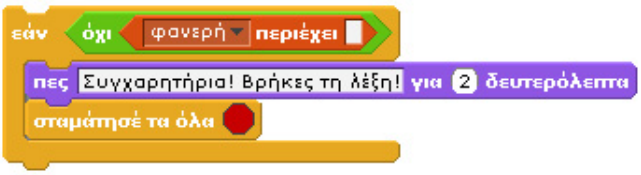
Τι συμβαίνει όμως αν το γράμμα δεν ανήκει μέσα στη κρυφή λίστα; Τότε ο δεύτερος παίκτης έχει κάνει ένα λάθος. Αλλά πόσα λάθη θα επιτρέψουμε στον παίκτη να κάνει; Έστω ότι δίνουμε στον παίκτη επτά ευκαιρίες. Αυτός ο αριθμός θα πρέπει να μειώνεται κάθε φορά που ο παίκτης κάνει λάθος και όταν μηδενιστεί, τότε να τερματίζεται και το παιχνίδι. Για να μπορούν όμως να γίνονται οι ενέργειες αυτές πρέπει ο αριθμός των λαθών να αποθηκεύεται σε μία μεταβλητή. Δημιουργείστε, λοιπόν, τη μεταβλητή με όνομα «λάθη» και στη συνέχεια προσθέστε στο προηγούμενο σενάριό μας, τις παρακάτω εντολές:



Οι εντολές αυτές απλά εξετάζουν αν η κρυφή λίστα δεν περιέχει το γράμμα που έδωσε ο χρήστης, και εφόσον κάτι τέτοιο ισχύει, μειώνουν τα διαθέσιμα λάθη του χρήστη.

Γ) Η φανερή λίστα δεν περιέχει άλλα κενά, γεγονός που σημαίνει ότι ο δεύτερος παίκτης τα κατάφερε και βρήκε τη λέξη.

Στην περίπτωση που ο χρήστης έχει βρει τη λέξη, η φανερή λίστα δεν θα περιέχει κενά, γεγονός που πρέπει να οδηγήσει στον τερματισμό του παιχνιδιού. Ο έλεγχος θα γίνει και σε αυτή την περίπτωση με τη βοήθεια του λογικού τελεστή άρνησης **όχι...** και της εντολής συνθήκης **[φανερή] περιέχει [ ]**. Όταν η συνθήκη αυτή γίνει αληθής, τότε θα πρέπει ο παρουσιαστής να δηλώνει στον παίκτη ότι κέρδισε και να τερματίζεται το παιχνίδι. Όλες αυτές οι εντολές εμφανίζονται στην παρακάτω εικόνα.



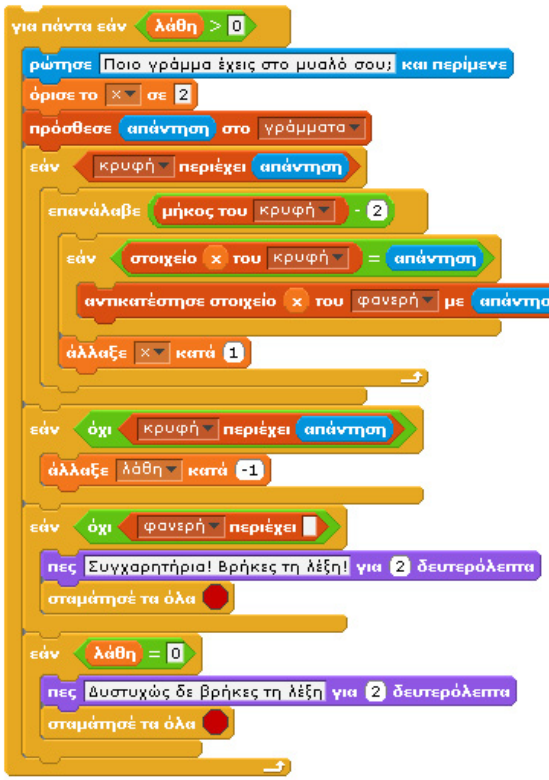
Δ) Ο χρήστης φτάνει το όριο των λαθών που τέθηκε από την αρχή

Στην περίπτωση, που ο παίκτης έχει εξαντλήσει όλες του τις ευκαιρίες, δηλαδή όταν η μεταβλητή «λάθη» έχει γίνει 0, το παιχνίδι πρέπει να τερματίζεται και να ανακοινώνεται στον παίκτη ότι δε βρήκε τη λέξη. Ο έλεγχος σε αυτή την περίπτωση είναι πιο απλός και υλοποιείται με τη βοήθεια του τελεστή ισότητας [λάθη] = [0].

Το μόνο που δεν έχουμε κάνει μέχρι αυτή τη στιγμή είναι να μπορεί ο χρήστης να βλέπει ποια γράμματα έχει εισάγει. Για να το καταφέρουμε και αυτό, θα δημιουργήσουμε μια νέα λίστα με όνομα «γράμματα», θα την εμφανίσουμε στην οθόνη του Scratch και θα προσθέτουμε σε αυτήν όλες τις απαντήσεις του δεύτερου παίκτη με τη χρήση της εντολής **πρόσθεσε «απάντηση» στα «γράμματα»**.

Επίσης δεν πρέπει να ξεχνάμε, ότι για να ξαναπαίξουμε το παιχνίδι θα πρέπει οι λίστες να είναι άδειες και για το λόγο αυτό, στην αρχή του σεναρίου μας πρέπει να διαγράφουμε όλα τα περιεχόμενά τους με την εντολή **διέγραψε [όλα] από το [φανερή/κρυφή/γράμματα]**.

Μελετήστε προσεκτικά το ολοκληρωμένο σενάριο.



Και φυσικά μια διαπίστωση για τους αποδοτικούς προγραμματιστές μας: υπάρχει μια εύκολη αλλαγή που μπορούμε να κάνουμε και η οποία θα κάνει το πρόγραμμά μας λίγο γρηγορότερο από όσο είναι τώρα. Ποια είναι αυτή; Αναζητήστε την απάντηση στις εντολές επανάληψης του σεναρίου.

Παράδειγμα «Love chase»:

Τι θα λέγατε για ένα κυνηγητό; Όταν ακούμε κυνηγητό μας έρχονται στο μυαλό διάφορες ιστορίες όπως ένα περιπολικό να κυνηγάει το αυτοκίνητο των ληστών ή ακόμη και το παιχνίδι όπου ένας παίκτης καλείται να πιάσει όλους τους υπόλοιπους. Εδώ θα δημιουργήσουμε ένα διαφορετικό είδος κυνηγητού. Πόσοι από εσάς δεν έχετε φανταστεί να σας κυνηγάει το αγόρι ή το κορίτσι των ονείρων σας; Αυτό όμως μερικές φορές μπορεί να καταλήξει ενοχλητικό! Στη δική μας ιστορία ο κύριος της «παρέας» δεν αντέχει άλλο το στενό μαρκάρισμα της θαυμάστριας του και τρέχει να γλιτώσει! Εκείνη όμως τον ακολουθεί κατά βήμα, αφού μπορεί και καταγράφει κάθε του κίνηση! Δημιουργήστε δύο φιγούρες της επιλογής σας, έτσι ώστε η μία να κινείται από τον παίκτη και η δεύτερη να ακολουθεί την πορεία της πρώτης. Η επόμενη εικόνα είναι αποκαλυπτική για το τι πρέπει να φαίνεται στο έργο σας! Έχει πολύ πλάκα, δοκιμάστε το.

[14_π06.sb]

Αντικείμενα και Σκηνικό. Στο κυνηγητό που θα δημιουργήσουμε συμμετέχουν δύο αντικείμενα. Το ένα αντικείμενο θα μοιάζει με αγόρι ενώ το άλλο με κορίτσι. Μπορείτε να επιλέξετε ό, τι θέλετε. Στο συγκεκριμένο παράδειγμα έχουν χρησιμοποιηθεί οι μορφές fantasy10 και fantasy14 από τον κατάλογο Fantasy της βιβλιοθήκης μορφών του Scratch. Στη μορφή fantasy14 έχουν γίνει τροποποιήσεις στο χρώμα με τη βοήθεια της ζωγραφικής, ενώ και στις δύο μορφές έχει γίνει σμίκρυνση αφού το κυνηγητό θέλει χώρο! Το σκηνικό επίσης δεν παίζει ιδιαίτερο ρόλο και για το λόγο αυτό χρησιμοποιούμε ένα απλό λευκό σκηνικό. Στα αντικείμενα μας έχουμε δώσει τα ονόματα Mr.Blue και Mrs.Pink. Ας δούμε τα δυο αντικείμενα ξεχωριστά.

Mr.Blue

Ας σκεφτούμε πιο αναλυτικά τι πρέπει να κάνει αρχικά το αντικείμενο Mr.Blue, ο ήρωας που τρέχει για να ξεφύγει από την κατά τα άλλα πολυαγαπημένη του. Ο χαρακτήρας αυτός ελέγχεται από το χρήστη και πρέπει να μετακινείται προς όλες τις κατευθύνσεις, χρησιμοποιώντας τα βελάκια του πληκτρολογίου. Όμως, εφόσον η Mrs.Pink πρέπει να τον ακολουθεί, σημαίνει ότι ο Mr.Blue αφήνει πίσω του ίχνη της κίνησής του, κάτι που μπορεί να γίνει με τη βοήθεια της πένας. Αλλά είναι αυτό από μόνο του αρκετό για να τον εντοπίσει; Μάλλον όχι. Για να μπορεί η Mrs.Pink να τον ακολουθεί κατά βήμα πρέπει να γνωρίζει τις ακριβείς συντεταγμένες της πορείας του κάθε στιγμή. Αυτό μπορεί να συμβεί μόνο αν ο Mr.Blue αποθηκεύει τις συντεταγμένες από τις οποίες έχει περάσει και η Mrs.Pink να έχει πρόσβαση σε αυτές. Μπορείτε να σκεφτείτε που θα καταγράφονται οι συντεταγμένες του Mr.Blue;

Mrs.Pink

Η Mrs.Pink, από την άλλη πλευρά, φροντίζει να ενημερώνεται για τις θέσεις από τις οποίες έχει περάσει ο Mr.Blue, διαβάζοντας τις καταχωρήσεις του και πηγαίνοντας προς αυτές ελπίζοντας ότι θα τον πετύχει. Όμως, ως φανατική θαυμάστρια που είναι, θέλει να σβήνει τα ίχνη του Mr.Blue έτσι ώστε καμία άλλη να μην μπορεί να τον εντοπίσει!

Κίνηση του Mr.Blue

Ας ξεκινήσουμε την υλοποίηση του παιχνιδιού δημιουργώντας την κίνηση του Mr.Blue από τον παίκτη. Για την κίνηση αυτή θα χρησιμοποιήσουμε τα βελάκια του πληκτρολογίου. Έχουμε δει πολλαπλά τέτοια παραδείγματα σε προηγούμενα παιχνίδια. Μελετήστε τα παρακάτω σενάρια και διαπιστώστε ποια είναι η διαφορά από τους τρόπους κίνησης που έχουμε δει μέχρι τώρα.

Σε αντίθεση με τα προηγούμενα παραδείγματα, στα οποία προσδιορίζαμε ένα συγκεκριμένο αριθμό βημάτων για την αλλαγή της θέσης του αντικειμένου, εδώ προσδιορίζουμε την αλλαγή μέσω μιας μεταβλητής, της μεταβλητής «βήμα». Μέσω αυτής της μεταβλητής μπορούμε να αλλάζουμε το βήμα με το οποίο κινείται το αντικείμενο προς όλες τις κατευθύνσεις αλλάζοντας μόνο την τιμή της μεταβλητής. Ανάλογα με την ταχύτητα που θέλουμε να κινείται ο ήρωάς μας, μπορούμε απλά να αλλάζουμε το βήμα στην αρχή του παιχνιδιού. Παρατηρήστε ότι οι αρνητικές που θέλαμε να δοθούν όταν αλλάζουν οι θέσεις x και y προς τα αριστερά και προς τα κάτω αντίστοιχα, δίνονται με τη βοήθεια του τελεστή πολλαπλασιασμού …*… από την παλέτα **Τελεστές**, πολλαπλασιάζοντας τη μεταβλητή «βήμα» με το -1.

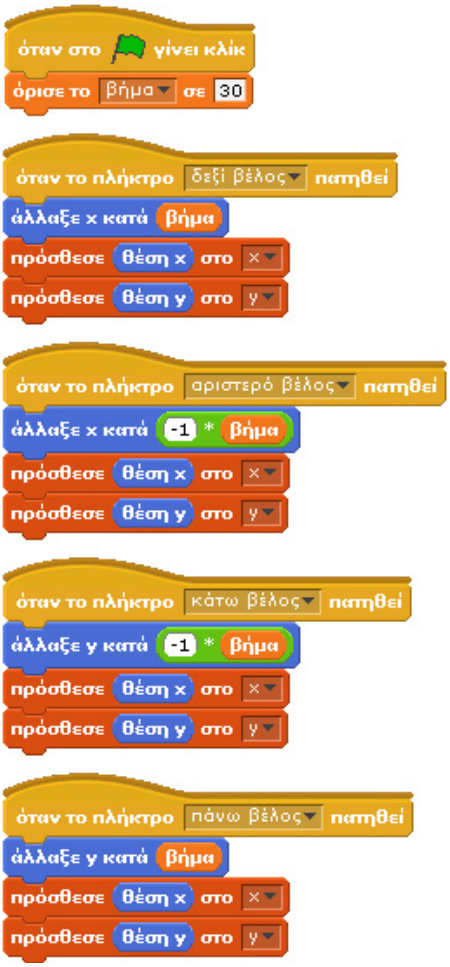
Καταγραφή της κίνησης του Mr.Blue

Που θα καταγράφονται οι θέσεις του Mr.Blue; Από τη στιγμή που θέλουμε να έχουμε στη διάθεση μας όλες τις θέσεις από τις οποίες περνάει ο Mr.Blue, χρειαζόμαστε μια λίστα που να αποθηκεύει τις συντεταγμένες της μετακίνησής του. Μόνο μία λίστα όμως είναι αρκετή; Αν και οι δύο συντεταγμένες του Mr.Blue αποθηκεύονται σε μία θέση της λίστας θα μπορέσουμε να τις επεξεργαστούμε; Η απάντηση είναι πώς δεν μπορούμε να ξεχωρίσουμε αυτές τις τιμές όταν αποθηκεύονται στην ίδια θέση και για το λόγο αυτό θα πρέπει να δημιουργήσουμε δύο λίστες: στη μία θα αποθηκεύονται οι θέσεις x του Mr.Blue και στην άλλη οι θέσεις y, προσέχοντας όμως η θέση αποθήκευσης κάθε φορά στις δυο λίστες να είναι αντίστοιχη.

Ας ονομάσουμε τις λίστες αυτές «x» και «y» αντίστοιχα. Η Mrs.Pink μπορεί και ακολουθεί κατά βήμα τον Mr.Blue γιατί έχει πρόσβαση στις τιμές αυτές. Άρα, οι λίστες «x» και «y» πρέπει να είναι «Για όλες τις μορφές», δηλαδή κοινές.

Τώρα πρέπει να αποφασίσουμε πότε θα καταγράφονται οι τιμές των θέσεων στις οποίες βρίσκεται ο Mr.Blue. Οι θέσεις αυτές μπορούν να καταγράφονται κάθε φορά που ο παίκτης πατήσει κάποιο βελάκι. Άλλωστε, μόνο τότε αλλάζει η θέση του Mr.Blue και τότε πρέπει να εισάγουμε στις λίστες τις νέες συντεταγμένες του.

Ο στόχος μας θα πραγματοποιηθεί με τις εντολές **πρόσθεσε [θέση x] στο [x]** και **πρόσθεσε [θέση y] στο [y]** από την παλέτα **Μεταβλητές**. Οι προσθήκες πρέπει να γίνουν ακριβώς μετά τις εντολές μετακίνησης του αντικειμένου. Το προηγούμενο σενάριο έχει γίνει πλέον:



Όπως είπαμε και στην αρχική περιγραφή του Mr.Blue, ο χαρακτήρας θα πρέπει όταν μετακινείται να αφήνει πίσω του ίχνη. Για το λόγο αυτό θα χρησιμοποιήσουμε την πένα. Αρχικά, πρέπει να ορίσουμε το χρώμα και το μέγεθος της πένας. Οι ιδιότητες αυτές καλό είναι να καθοριστούν κατά την έναρξη του προγράμματος. Έτσι, προσθέτουμε τις εντολές **όρισε το χρώμα πένας σε…** και **όρισε το μέγεθος πένας σε [2]** από την παλέτα **Πένα**. Επίσης, για να μπορέσει να αφήνει τα ίχνη του πρέπει το αντικείμενο να κατεβάσει την πένα με τη βοήθεια της εντολής **κατέβασε πένα**. Ανανεώνουμε για άλλη μια φορά το αρχικό μας σενάριο.

Τέλος, είναι αναγκαίο όταν ξεκινάει το παιχνίδι, οι λίστες με τις θέσεις από τις οποίες έχει περάσει ο Mr.Blue να είναι άδειες και να μην περιέχουν τιμές από προηγούμενο παιχνίδι. Για το λόγο αυτό, κατά την έναρξη, πρέπει να διαγράφονται όλα τα στοιχεία που πιθανόν να περιέχουν. Αυτό θα πραγματοποιηθεί με τις εντολές **διέγραψε [όλα] από το [x]** και **διέγραψε [όλα] από το [y]**. Επιπλέον, πρέπει να καθαρίζεται και η σκηνή από τυχόν υπολειπόμενα ίχνη του Mr.Blue με την εντολή **καθάρισε**. Το ολοκληρωμένο πλέον σενάριο του Mr.Blue όταν πατηθεί η πράσινη σημαία είναι το παρακάτω:

Κίνηση της Mrs.Pink

Τι θα λέγατε να βοηθήσουμε την Mrs.Pink να έρθει πιο κοντά στον Mr.Blue; Όπως έχουμε αναφέρει μέχρι τώρα, το αντικείμενο αυτό ακολουθεί κατά βήμα την κίνηση του Mr.Blue. Για να το πετύχει αυτό πρέπει να εκμεταλλευτεί τις λίστες «x» και «y» που περιέχουν τις πληροφορίες μετακίνησης του άλλου αντικειμένου. Σκοπός της Mrs.Pink είναι να περάσει από όλες τις θέσεις από τις οποίες πέρασε και ο Mr.Blue. Επομένως, όσο βρίσκει στοιχεία στις λίστες αυτές, ίχνη δηλαδή του Mr. Blue, η κίνηση της δε σταματάει. Πως όμως θα διαβάσει τις διαφορετικές συντεταγμένες που είναι αποθηκευμένες στις λίστες; Με μια επανάληψη και μια μεταβλητή-μετρητή. Η επανάληψη θα διαρκεί όσο ο μετρητής είναι μικρότερος από το μήκος της λίστας, δηλαδή όσο υπάρχουν στοιχεία μέσα στη λίστα που δεν έχει επισκεφτεί. Όσο δηλαδή δεν έχει φτάσει στον αγαπημένο της!

Αρχικά, η μεταβλητή μετρητής θα πρέπει να παίρνει την τιμή 0 και στη συνέχεια θα πρέπει να αλλάζει κατά ένα σε κάθε επανάληψη έτσι ώστε κάθε φορά να δείχνει στα επόμενα στοιχεία των λιστών x και y. Η κίνηση προς τα στοιχεία αυτά για την Mrs. Pink θα πραγματοποιείται με την εντολή **πήγαινε στο x… y…** από την παλέτα **Κίνηση**. Οι συντεταγμένες του σημείου μετακίνησης θα προκύψουν από τη χρήση των εντολών **στοιχείο [ ] του [x]** και **στοιχείο [ ] του [y]** σε συνδυασμό με τη μεταβλητή-μετρητή «θέση». Μελετήστε το παρακάτω σενάριο:

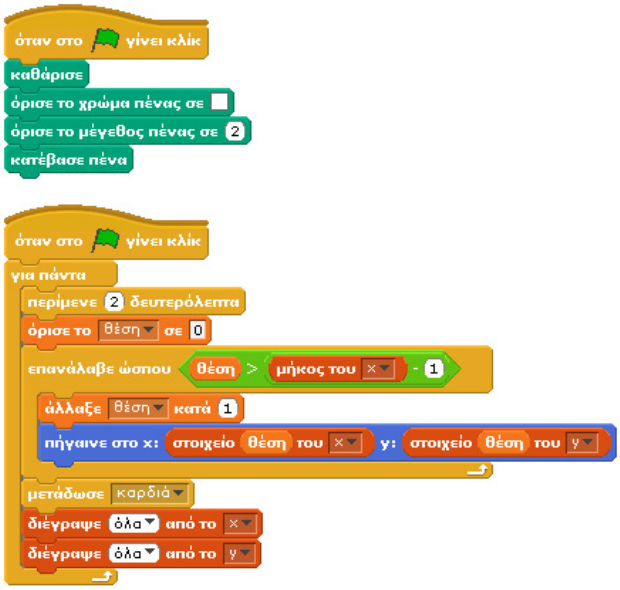
Γιατί έχουμε τη συγκεκριμένη συνθήκη στην επανάληψη και όχι τη συνθήκη **(θέση > μήκος του x)**; Πως θα ξαναγράφατε τη συνθήκη χρησιμοποιώντας τον τελεστή της ισότητας;

Ας γυρίζουμε όμως στο παιχνίδι. Ακόμη κι αν η Mrs.Pink φτάσει τον αγαπημένο της, το παιχνίδι δε σταματά αφού ο Mr.Blue μπορεί να ξεφύγει και να ξεκινήσει νέο κυνηγητό. Για το λόγο αυτό η διαδικασία της κίνησης της Mrs.Pink θα πρέπει να εκτελείται συνεχώς. Αυτό το εξασφαλίζει η εντολή «για πάντα…».

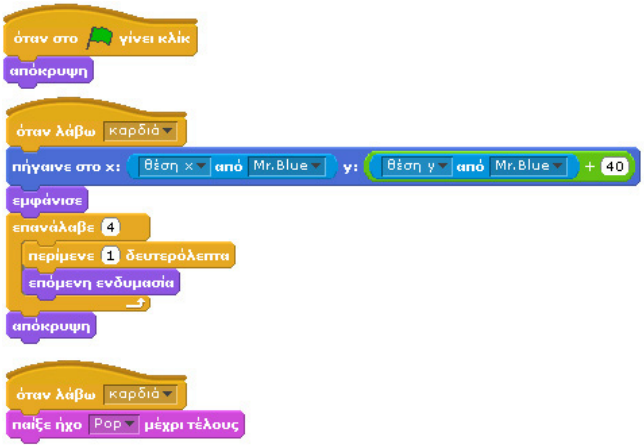
Ακόμη, η Mrs.Pink επιθυμεί να σβήνει τα ίχνη του αγαπημένου της. Για το λόγο αυτό θα χρησιμοποιήσει πένα με χρώμα λευκό και πάχος ίσο με αυτό της πένας του Mr.Blue. Οι ιδιότητες της πένας θα ορίζονται κατά την έναρξη του προγράμματος, όπως και το κατέβασμά της. Ίχνη βέβαια δεν αποτελούν μόνο τα αποτυπώματα που αφήνει η πένα του Mr.Blue αλλά και οι συντεταγμένες των θέσεων από τις οποίες πέρασε. Για το λόγο

αυτό η Mrs.Pink κάθε φορά που συναντά τον Mr.Blue διαγράφει τα περιεχόμενα των λιστών «x» και «y» με τη χρήση της εντολής **διέγραψε [όλα] από το [x/y]**.

Τέλος, επειδή η Mrs.Pink πρέπει να αφήνει τον Mr.Blue για λίγο ελεύθερο για να μετακινηθεί, θα χρειαστούμε και την εντολή **περίμενε…δευτερόλεπτα**. Τα ολοκληρωμένα σενάρια για την Mrs.Pink φαίνεται στην παρακάτω εικόνα.



Τι θα λέγατε να προσθέταμε ένα ακόμη αντικείμενο το οποίο να έχει σχήμα καρδιάς και κάθε φορά που η Mrs.Pink φτάνει τον Mr.Blue να εμφανίζεται στη σκηνή; Για να γνωρίζει βέβαια αυτό το αντικείμενο πότε να εμφανιστεί , θα πρέπει η Mrs.Pink να μεταδίδει ένα μήνυμα τη στιγμή που φτάνει τον αγαπημένο της. Ας ονομάσουμε το αντικείμενο αλλά και το συγκεκριμένο μήνυμα ως «καρδιά». Όταν λοιπόν το μήνυμα αυτό ληφθεί από το αντικείμενο Καρδιά, τότε ο έρωτας τις Mrs.Pink θα γίνεται ακόμη πιο φανερός, ενώ το αντικείμενο μπορεί να αλλάζει ενδυμασίες έτσι ώστε να φαίνεται ότι μεγαλώνει και να αναπαράγει και κάποιον ήχο. Μία ενδεικτική υλοποίηση του αντικειμένου αυτού παρουσιάζεται στην επόμενη εικόνα.



[http://www.shallwelearn.com/scratchprogrammingforkidscategory/project-tutorial/176-love-chase-more-on-lists-]

Παράδειγμα «ο δεινόσαυρος-μαθηματικός»: Ας δούμε ένα πρόγραμμα το οποίο υπολογίζει απλές μαθηματικές εκφράσεις χρησιμοποιώντας, τις μεταβλητές, τους τελεστές της πρόσθεσης (+) και του πολλαπλασιασμού (*). Σκοπός μας είναι να δημιουργήσουμε ένα πρόγραμμα το οποίο να μπορεί να υλοποιεί και να υπολογίζει το αποτέλεσμα των παρακάτω μαθηματικών εκφράσεων ανάλογα με την επιλογή του χρήστη.

- ✓ Εμβαδό ορθογωνίου = α * β, όπου α και β οι πλευρές του ορθογωνίου.
- ✓ Περίμετρος ορθογωνίου = 2*α+2*β, όπου α και β οι πλευρές του ορθογωνίου.
- ✓ Εμβαδό κύκλου = 3.14*α*α , όπου α η ακτίνα του κύκλου.
- ✓ Περίμετρος κύκλου = 2*(3.14)*α, όπου α η ακτίνα του κύκλου.

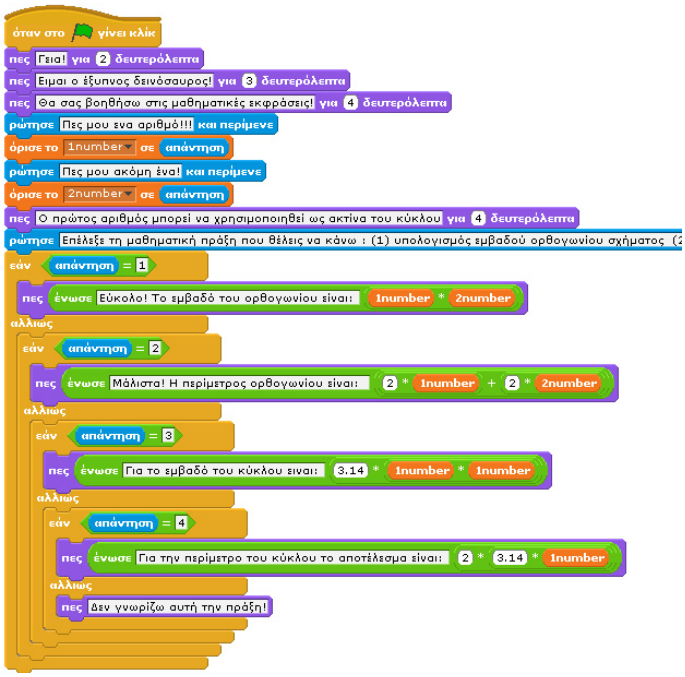
Τις παραμέτρους α και β θα πρέπει να τις δίνει ο χρήστης ως είσοδο στο πρόγραμμα και το έργο μας θα κάνει τους υπολογισμούς. Δηλαδή θα δημιουργήσουμε ένα γεωμετρικό υπολογιστή!

Για τον υπολογισμό των παραπάνω μαθηματικών εκφράσεων θα μας βοηθήσει ένας μικρός δεινόσαυρος. Το σκηνικό και ο δεινόσαυρος φαίνονται στην επόμενη εικόνα.

Όπως αναφέραμε, ο χρήστης θα πρέπει να δίνει δύο αριθμούς βάσει των οποίων θα γίνονται οι υπολογισμοί με τη χρήση της εντολής **ρώτησε…και περίμενε** και της μεταβλητής **απάντηση**. Για να μπορέσουμε να χρησιμοποιήσουμε τους αριθμούς αυτούς πολλές φορές και σε διαφορετικές εντολές, θα πρέπει να τους αποθηκεύσουμε. Για το λόγο αυτό θα χρειαστούμε τη δημιουργία δυο αντίστοιχων μεταβλητών με όνομα 1number και 2number. Η πρώτη τιμή που εισάγει ο χρήστης θα αποθηκευτεί στη μεταβλητή με όνομα 1number, ενώ η δεύτερη στη 2number.

Έπειτα, ο χρήστης καλείται να επιλέξει ποια πράξη επιθυμεί να εκτελεστεί. Αν επιλέξει τον αριθμό 1, τότε πραγματοποιείται η πράξη για την εύρεση του εμβαδού ενός ορθογωνίου. Αν επιλέξει τον αριθμό 2, πραγματοποιείται ένας συνδυασμός πράξεων για την εύρεση της περιμέτρου του ορθογωνίου. Αν επιλέξει 3, ο δεινόσαυρος εκτελεί τον υπολογισμό του εμβαδού του κύκλου. Αν επιλέξει 4, εκτελείται η πράξη για την εύρεση της περιμέτρου του κύκλου. Όλοι υπολογισμοί μπορούν να γίνουν με τη χρήση των τελεστών πολλαπλασιασμού [ ]*[ ] και πρόσθεσης [ ]+[ ]. Για να γίνει όμως πιο πειστική η ανακοίνωση του αποτελέσματος από το δεινόσαυρο, χρησιμοποιούμε ακόμη μία εντολή της παλέτας των Τελεστών, την **ένωσε [ ] [ ]** η οποία μας επιτρέπει να ενώσουμε ένα σύντομο κείμενο με μια μεταβλητή και να δημιουργήσουμε ένα ενιαίο κείμενο που μπορεί να «ειπωθεί» από τον ήρωα μας.

Μελετήστε το παρακάτω σενάριο:



[14_π07.sb]

[http://scratch.mit.edu/projects/mrdinosaur/949988]

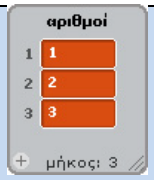
Περίληψη

Στο κεφάλαιο αυτό ασχοληθήκαμε με δύο ιδιαίτερα σημαντικές έννοιες τόσο του Scratch, όσο και του προγραμματισμού γενικότερα. Δεν είναι οι πιο εύκολες έννοιες που συναντήσαμε! Αλλά είναι αυτές που πραγματικά θα μας επιτρέψουν στη συνέχεια να αναπτύξουμε εκπληκτικά παιχνίδια. Ασχοληθήκαμε με τις μεταβλητές, που είναι δομές αποθήκευσης τιμών που χρειάζεται ένα έργο κατά τη διάρκεια εκτέλεσής του. Οι τιμές αυτές μπορεί να αφορούν ένα και μόνο αντικείμενο ή να διαβάζονται και να αλλάζουν από όλα τα αντικείμενα. Συνήθως, δίνουμε μια αρχική τιμή σε κάθε μεταβλητή στην αρχή του προγράμματος και αλλάζουμε την τιμή αυτή κατά τη διάρκεια εκτέλεσης του έργου. Σκορ, ζωές, χρόνος κτλ θα αποθηκεύονται τις περισσότερες φορές σε μεταβλητές. Στις λίστες μπορούν να αποθηκευτούν περισσότερες από μία τιμές χωρίς να χρειάζεται να γνωρίζουμε από την έναρξη του προγράμματος πόσες τιμές θέλουμε να αποθηκεύσουμε. Σε μία λίστα, μπορούμε να προσθέσουμε, να διαγράψουμε αλλά και να αντικαταστήσουμε τα στοιχεία της. Επιπλέον, το Scratch μας δίνει τη δυνατότητα να εξετάσουμε εύκολα το περιεχόμενο μιας λίστας και να γνωρίζουμε το μήκος της. Έχουμε λοιπόν δυο αποθηκευτικά εργαλεία που είναι απολύτως απαραίτητα όπως θα καταλάβετε και στο τελευταίο κεφάλαιο του βιβλίου μας.

Ερωτήσεις

1) Έστω ότι σε μία εφαρμογή έχουμε τη μεταβλητή «άθροισμα» την οποία αρχικοποιούμε με την τιμή 0. Ποια τιμή θα έχει η μεταβλητή μετά την εκτέλεση της εντολής **άλλαξε [άθροισμα] κατά [-2]**;

2) Στην παρακάτω εικόνα εμφανίζεται η λίστα «αριθμοί» με τα περιεχόμενά της.

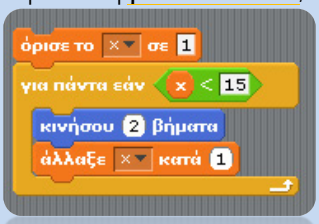


Ποια θα είναι η μορφή της λίστας μετά την εκτέλεση της εντολής **παρέμβαλλε το [4] στη θέση [2] του [αριθμοί]**;

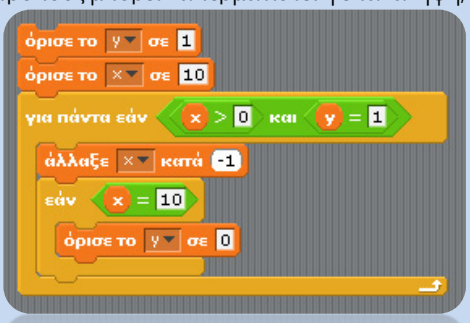
3) Αν μία λίστα περιέχει 3 στοιχεία και εμείς εκτελέσουμε την εντολή **παρέμβαλλε το [αντικείμενο] στη θέση [οποιοδήποτε] του [λίστα]**, σε πόσα σημεία είναι δυνατό να μπει το αντικείμενο;

4) Αν θέλετε κάθε στοιχείο που προσθέτετε σε μία λίστα να μπαίνει στην πρώτη και όχι στην τελευταία της θέση, ποια εντολή πρέπει να χρησιμοποιήσετε;

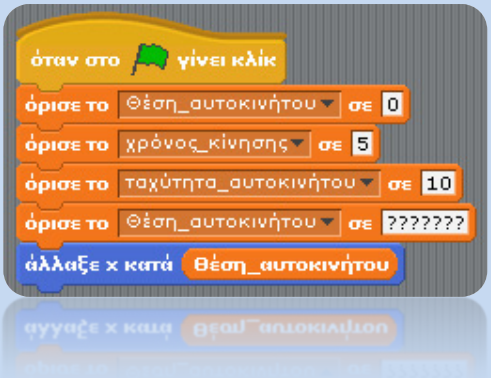
5) Στην παρακάτω εικόνα δίνεται ένα σενάριο επανάληψης. Πόσες φορές θα εκτελεστεί ο κώδικας που περιέχεται μέσα στην εντολή **για πάντα εάν ...**;



6) Δίνεται το παρακάτω σενάριο. Με πόσους διαφορετικούς τρόπους μπορεί να τερματιστεί η επανάληψη;



7) Δίνεται το παρακάτω σενάριο στο οποίο θέλουμε να υπολογίζουμε τη θέση ενός αυτοκινήτου που κινείται με ταχύτητα «ταχύτητα_αυτοκινήτου» μετά από χρόνο «χρόνος_κίνησης». Στον κώδικα αυτό λείπει ο τύπος για τον υπολογισμό της θέσης αυτής. Μπορείτε να τον συμπληρώσετε με τη βοήθεια των αριθμητικών τελεστών; Θυμίζουμε ότι ο τύπος για τη μετατόπιση ενός αντικειμένου που κινείται ομαλά είναι: x_τελικό = x_αρχικό + u * Δt .



8) Δημιουργείστε ένα έργο το οποίο θα ζητάει από το χρήστη την είσοδο δύο αριθμών που θα αντιστοιχούν στο μήκος των δύο κάθετων πλευρών ενός ορθογωνίου τριγώνου. Το έργο σας θα υπολογίζει το μήκος της υποτείνουσας με τη βοήθεια του Πυθαγορείου Θεωρήματος. Το Πυθαγόρειο Θεώρημα είναι το ακόλουθο: $\alpha^2 = \beta^2 + \gamma^2$.

9) Δημιουργείστε ένα έργο στο οποίο ο χρήστης θα δίνει τυχαία αριθμούς τους οποίους το έργο σας θα αποθηκεύει σε μία λίστα αλλά πάντα με αριθμητική σειρά από το μικρότερο προς το μεγαλύτερο αριθμό.

10) Δημιουργήστε ένα πρόγραμμα που θα υλοποιεί την αντίστροφη μέτρηση. Στο πρόγραμμα αυτό, ο χρήστης θα δίνει τον αριθμό των δευτερολέπτων και έπειτα ο χρόνος θα μετράει αντίστροφα.

Δραστηριότητες

1) Δημιουργήστε μία εφαρμογή στην οποία θα υπάρχει μία φιγούρα της επιλογής σας, όπως ένα άτομο ή ένα ζώο, το οποίο θα προσπαθεί να συλλέξει τη μεγαλύτερη αξία σε κέρματα που μπορεί μέσα σε 15 δευτερόλεπτα. Τα κέρματα που θα εμφανίζονται στη σκηνή θα πρέπει να είναι τόσα ώστε να μην είναι εύκολο να τα μαζέψει όλα μέσα σε αυτό το χρονικό διάστημα. Ένας ικανοποιητικός αριθμός κερμάτων θα ήταν 30. Επιπλέον, τα κέρματα θα πρέπει να έχουν διαφορετική αξία, π.χ. να υπάρχουν χρυσά και ασημένια τα οποία θα προσφέρουν διαφορετικούς πόντους στον παίκτη. Μία ενδεικτική τιμή θα ήταν οι 3 πόντοι για τα χρυσά και 1 πόντος για τα ασημένια. Επίσης, θα είχε ενδιαφέρον αν στη σκηνή βάζατε και κάποια εμπόδια που θα κάνουν δύσκολη την πρόσβασή του χαρακτήρα στα κέρματα. Το πλήθος, τη μορφή και τις θέσεις των εμποδίων μπορείτε να τις αποφασίσετε μόνοι σας. Τα εμπόδια θα μπορούσαν να έχουν για παράδειγμα τη μορφή βράχων. Σε περίπτωση που τα ακουμπήσει θα χάνει. Νικητής είναι αυτός που θα συγκεντρώσει τους περισσότερους πόντους μέσα στα 15 αυτά δευτερόλεπτα χωρίς να πέσει πάνω στα εμπόδια.

2) Δημιουργήστε ένα τυχερό παιχνίδι! Στο παιχνίδι αυτό θα υπάρχει μία λίστα όλων των ακεραίων αριθμών από το 1 έως το 40. Ο χρήστης αρχικά θα πληκτρολογεί έξι αριθμούς της επιλογής του. Στη συνέχεια, με τυχαίο τρόπο 34 από τους 40 αριθμούς της λίστας θα διαγράφονται. Ανάλογα με το πόσοι από τους έξι αριθμούς που έδωσε ο χρήστης βρίσκονται στη νέα λίστα θα προσδιορίζεται και το ποσοστό της επιτυχίας του.

3)Δημιουργήστε μία εφαρμογή στην οποία θα υπάρχει ένα κανόνι που θα κινείται από το χρήστη. Η θέση του κανονιού θα παραμένει σταθερή στο κάτω μέρος της σκηνής, ενώ η κατεύθυνση του θα μπορεί να αλλάζει με το δεξί και το αριστερό βέλος του πληκτρολογίου. Μπορείτε να επιλέξετε εσείς ποιο θα είναι το βήμα με το οποίο θα αλλάζει η κατεύθυνση του κανονιού, μία ενδεικτική τιμή είναι οι 5 μοίρες. Το κανόνι θα ρίχνει σφαίρες στην κατεύθυνση στην οποία δείχνει, οι οποίες θα κινούνται μέχρι να συναντήσουν τα όρια της σκηνής. Με τις σφαίρες ο παίχτης θα προσπαθεί να πετύχει τις τέσσερις πάπιες που θα υπάρχουν στη σκηνή. Οι πάπιες αυτές θα κινούνται γύρω από τη θέση τους δεξιά και αριστερά. Τις αρχικές τους θέσεις μπορείτε να τις αποφασίσετε μόνοι σας. Άλλες μπορούν να είναι πιο κοντά στο κανόνι και άλλες πιο μακριά από αυτό. Κάθε πάπια έχει τρεις ζωές. Κάθε φορά που μία σφαίρα πετύχει κάποια πάπια, η ζωή της μειώνεται κατά μία. Όταν οι ζωές κάποιας πάπιας τελειώσουν, τότε αυτή θα πρέπει να βγαίνει από το παιχνίδι. Για να τερματιστεί το παιχνίδι θα πρέπει όλες οι πάπιες να έχουν χάσει τις ζωές τους. Επιπλέον, θα μπορούσατε να χρησιμοποιήσετε ένα χρονόμετρο για να περιορίσετε το χρόνο που έχει ο παίκτης στη διάθεση του.


Λοιπόν, πώς σου φάνηκε το μάθημα για τις μεταβλητές;
Μπορώ να πω ότι κράτησε το ενδιαφέρον μου αμετάβλητο...
Τόσο πολύ σου άρεσε δηλαδή;
Μπα, απλά με είχε πάρει ο ύπνος από την αρχή του μαθήματος...
...
Λέω να κοιμηθώ λίγο ακόμα την επόμενη ώρα. Ξύπνα με όταν πει τα SOS!