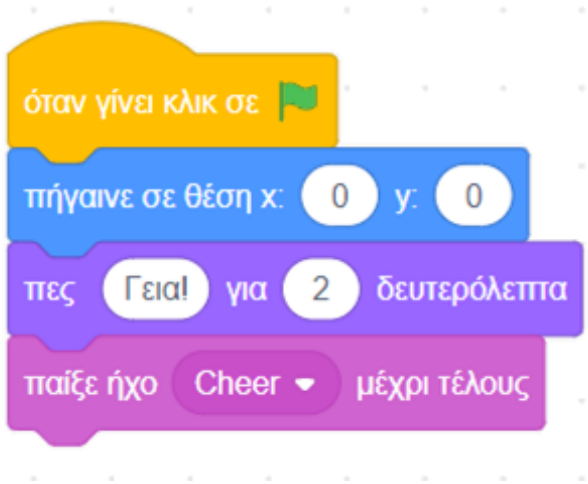

SCRATCH
forever
imagine
program
share

ΔΟΜΗ ΑΚΟΛΟΥΘΙΑΣ

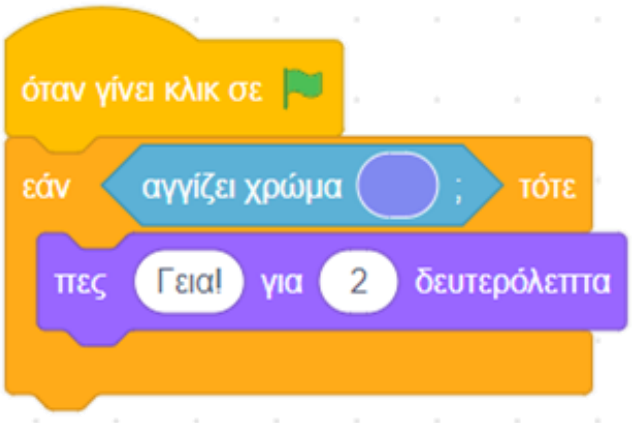
Στο πλαίσιο του **Προγραμματισμού** και κυρίως του Δομημένου Προγραμματισμού, ο τρόπος που συντάσσουμε τις εντολές μας παίζει σημαντικό ρόλο στην εκτέλεση συνολικά του κώδικά μας. Διακρίνουμε τρις κατηγορίες δομών προγραμματισμού και η δομή της ακολουθίας είναι η πιο απλή από τις τρείς δομές . Στη δομή αυτή οι εντολές που περιγράφουμε εκτελούνται όλες η μια μετά απο την άλλη , με τη σειρά (ή αλλιώς ακολουθιακά!).

Στο Προγραμματιστικό περιβάλλον του **Scratch** χρησιμοποιούμε τις πιο κάτω δομές (Παρουσίαση παραδειγμάτων):

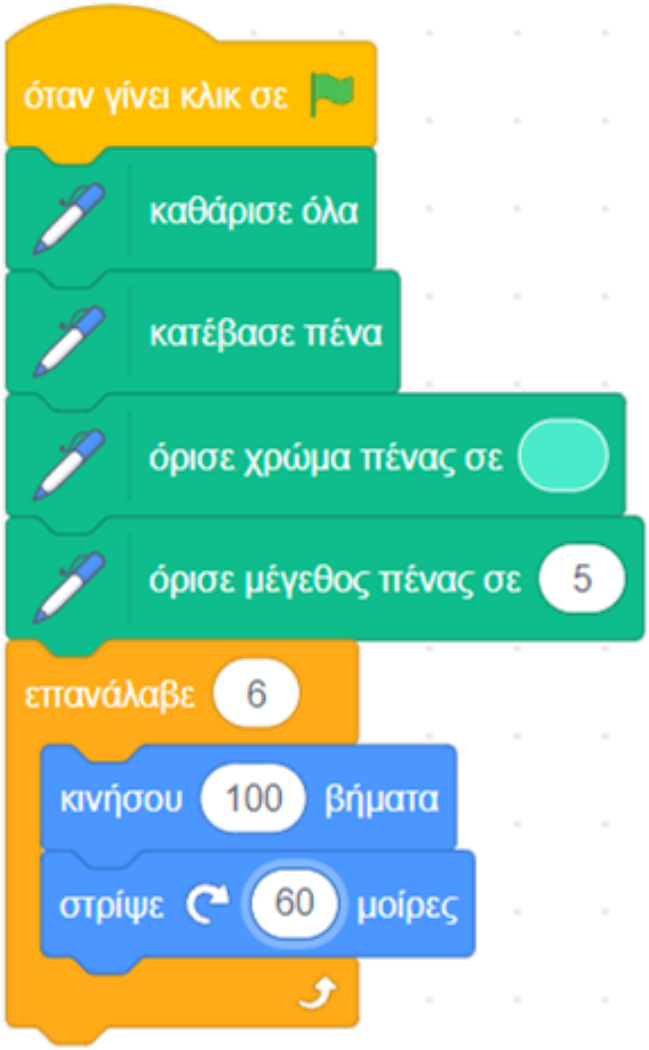
1. **ΔΟΜΗ ΑΚΟΛΟΥΘΙΑΣ** (Οι εντολές εκτελούνται η μία μετά την άλλη)



2. **ΔΟΜΗ ΕΠΙΛΟΓΗΣ** (Με βάση τη συνθήκη (Αληθής ή Ψευδής) εκτελείται μία, πολλές ή καμία εντολή)



3. ΔΟΜΗ ΕΠΑΝΑΛΗΨΗΣ (Με βάση τη συνθήκη (Αληθής ή Ψευδής) οι εντολές επαναλαμβάνονται)



ΔΟΜΗ ΕΠΙΛΟΓΗΣ (Εάν … τότε … αλλιώς)

Όπως πολύ συχνά συμβαίνει και στην καθημερινή μας ζωή, έτσι και στον προγραμματισμό, είναι ανάγκη να λαμβάνονται κάποιες αποφάσεις ανάλογα με το αν ισχύουν ή όχι κάποια ή κάποιες συνθήκες.

Για να μπορέσουμε να επιτύχουμε τη λήψη αποφάσεων και τη σωστή δρομολόγηση του προγράμματος ώστε να εκτελεί τις κατάλληλες εντολές για κάθε περίπτωση, χρησιμοποιούμε τη **δομή επιλογής** .

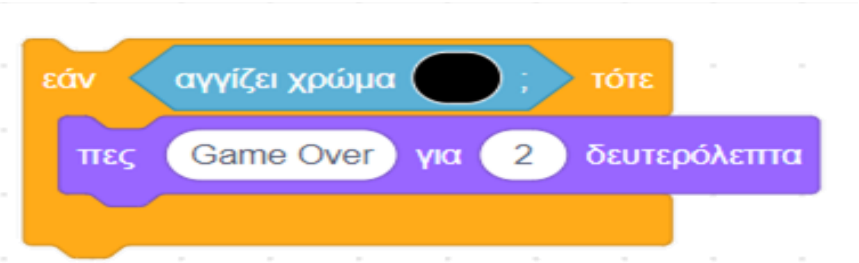
Στο Προγραμματιστικό περιβάλλον του **Scratch** χρησιμοποιούμε τις πιο κάτω δομές επιλογής:

1. Απλή Δομή Επιλογής **2. Σύνθετη Δομή Επιλογής** **3. Πολλαπλή Δομή Επιλογής**

1. Απλή Δομή Επιλογής

Τη δομή της απλής επιλογής τη χρησιμοποιούμε όταν θέλουμε να εκτελεστεί μια ομάδα (σειρά) εντολών, όταν ισχύει μια συνθήκη. Αν ισχύει η συνθήκη τότε εκτελείται η ομάδα των εντολών που περικλείεται μέσα στη δομή επιλογής (Εάν…τότε), αλλιώς αν δεν ισχύει η συνθήκη τότε δεν εκτελούνται οι συγκεκριμένες εντολές και το πρόγραμμα συνεχίζεται με τις επόμενες εντολές εκτός του (Εάν…τότε).

Παράδειγμα:

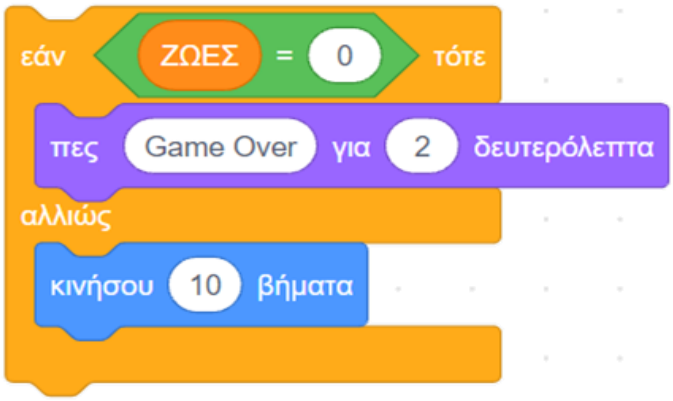


2. Σύνθετη Δομή Επιλογής

Σε πολλές όμως περιπτώσεις θα χρειάζεται να εκτελεστεί μια ομάδα εντολών ακόμη και στη περίπτωση όπου δεν ισχύει η λογική συνθήκη της επιλογής.

Η δομή της σύνθετης επιλογής μοιάζει πολύ με την δομή της απλής επιλογής με τη διαφορά ότι σε αυτή εκτελούνται εντολές ακόμη και αν δεν ισχύει η λογική συνθήκη. Οι εντολές που εκτελούνται σε αυτή την περίπτωση περιγράφονται μετά τη **δεσμευμένη λέξη** ΑΛΛΙΩΣ. Έτσι λοιπόν καταλαβαίνουμε ότι το ΑΛΛΙΩΣ καλύπτει τις περιπτώσεις εκείνες οι οποίες δεν περιλαμβάνονται στη λογική συνθήκη.

Παράδειγμα:



3. Πολλαπλή Δομή Επιλογής

Χρησιμοποιούμε την δομή της πολλαπλής επιλογής όταν υπάρχουν περισσότεροι από 2 εναλλακτικοί «δρόμοι» τους οποίους μπορεί να επιλέξει το πρόγραμμα για την εκτέλεσή του, ανάλογα με το ποια λογική συνθήκη ικανοποιείται κάθε φορά.

Στη δομή της πολλαπλής επιλογής μπορούμε να έχουμε στην ίδια επιλογή περισσότερες από μια λογικές συνθήκες. Πρέπει να επισημάνουμε ότι στην δομή αυτή κάθε φορά ικανοποιείται μόνο μια λογική συνθήκη (ανεξάρτητα από το πόσες έχουμε). Έτσι μόλις βρεθεί η λογική συνθήκη η οποία ικανοποιείται αμέσως εκτελείται η ομάδα εντολών που ανήκει στη συνθήκη αυτή και τερματίζεται η επιλογή, αυτό άμεσα σημαίνει ότι όλες οι άλλες συνθήκες είναι ψευδείς.

ΔΟΜΗ ΕΠΑΝΑΛΗΨΗΣ (Επανάλαβε – Επανάλαβε ώσπου – Περίμενε ώσπου)

Η διαδικασία της επανάληψης είναι ιδιαίτερα συχνή, σε πλήθος προβλημάτων στα οποία θα πρέπει να επαναληφθούν οι ίδιες ακριβώς εντολές πολλές φορές προκειμένου να γίνει εισαγωγή των δεδομένων, η επεξεργασία τους ή ακόμη και η εμφάνιση των αποτελεσμάτων.

Στο Προγραμματιστικό περιβάλλον του **Scratch** χρησιμοποιούμε τις πιο κάτω δομές επανάληψης:

1. ΓΙΑ ΠΑΝΤΑ

2. ΕΠΑΝΑΛΑΒΕ (Χ φορές)

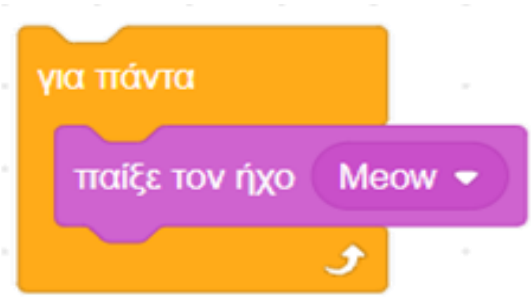
3. ΕΠΑΝΑΛΑΒΕ ΩΣΠΟΥ (Συνθήκη)

---

1. ΕΝΤΟΛΗ ΕΠΑΝΑΛΗΨΗΣ – **ΓΙΑ ΠΑΝΤΑ**

Η εντολή επανάληψης (ΓΙΑ ΠΑΝΤΑ) είναι η πιο χρησιμοποιούμενη εντολή στο προγραμματιστικό περιβάλλον του Scratch και χρησιμοποιείται όταν θέλουμε κάποιες εντολές να εκτελούνται συνέχεια (για πάντα). Οι εντολές αυτές περικλείονται μέσα στο block (για πάντα) και θα εκτελούνται συνέχεια μέχρι να τερματιστεί το πρόγραμμα. Τις περισσότερες φορές η εντολή (Για πάντα) χρησιμοποιείται συνδυαστικά με τις άλλες εντολές επανάληψης και επιλογής και σχηματικά φαίνεται να τις περικλείει μέσα της.

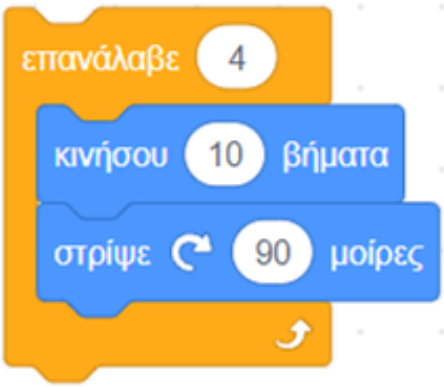
Παράδειγμα:



2. ΕΝΤΟΛΗ ΕΠΑΝΑΛΗΨΗΣ – **ΕΠΑΝΑΛΑΒΕ (Χ φορές)**

Σε πολλές περιπτώσεις προβλημάτων θα πρέπει να καθορίσουμε τον αριθμό των επαναλήψεων που θα εκτελεστούν. Αυτό επιτυγχάνεται με την εντολή **ΕΠΑΝΑΛΑΒΕ (Χ φορές)**, όπου φορές είναι ο αριθμός ων επαναλήψεων.

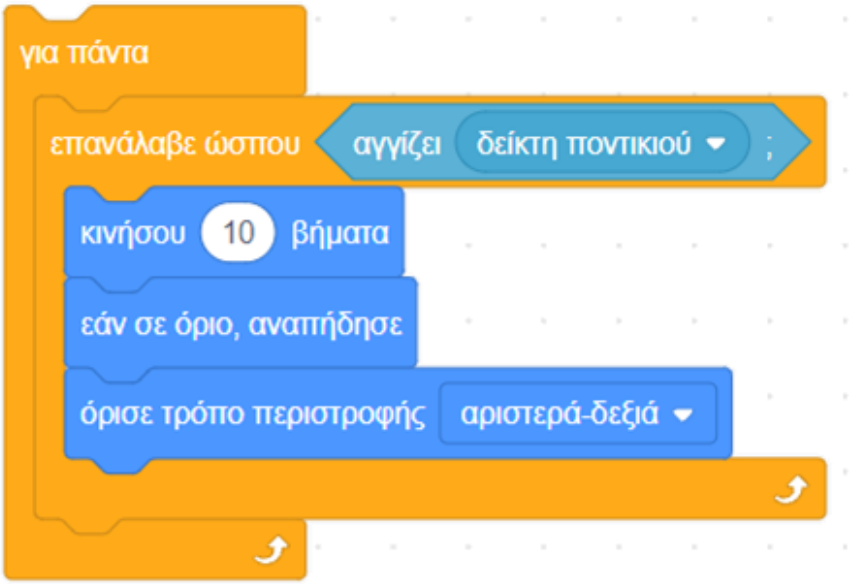
Παράδειγμα:



3. ΕΝΤΟΛΗ ΕΠΑΝΑΛΗΨΗΣ – **ΕΠΑΝΑΛΑΒΕ ΩΣΠΟΥ (Συνθήκη)**

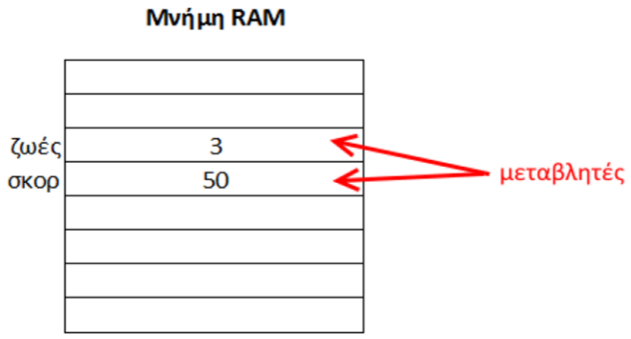
Σε αυτήν τη δομή επανάληψης οι εντολές που περικλείονται μέσα στο block (Επανάλαβε ώσπου) εκτελούνται επαναληπτικά μέχρι η συνθήκη να γίνει αληθής. Η λογική συνθήκη σε αυτή την περίπτωση είναι συνθήκη τερματισμού και όχι συνέχισης της επανάληψης. Παρατηρούμε ότι όσο η λογική συνθήκη είναι ψευδής η επανάληψη συνεχίζεται, ενώ όταν η λογική συνθήκη γίνει αληθής η επανάληψη τερματίζεται.

Παράδειγμα:



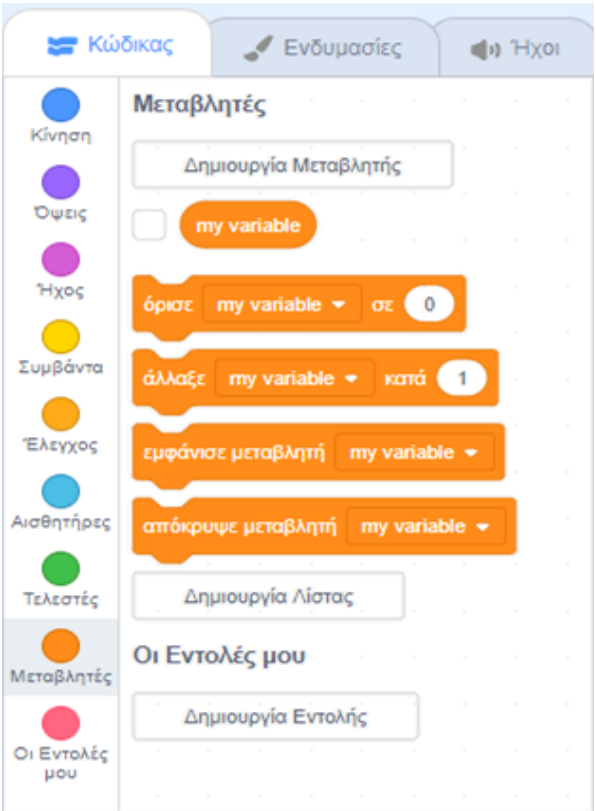
ΜΕΤΑΒΛΗΤΕΣ - Scratch

❖ ***μεταβλητές*** είναι συμβολικά ονόματα που αντιστοιχούν σε θέσεις μνήμης του υπολογιστή. Οι μεταβλητές χρησιμοποιούνται για την αποθήκευση και διαχείριση δεδομένων σε ένα πρόγραμμα (π.χ. σε μια μεταβλητή μπορείς να κρατάς το σκορ ενός παιχνιδιού).

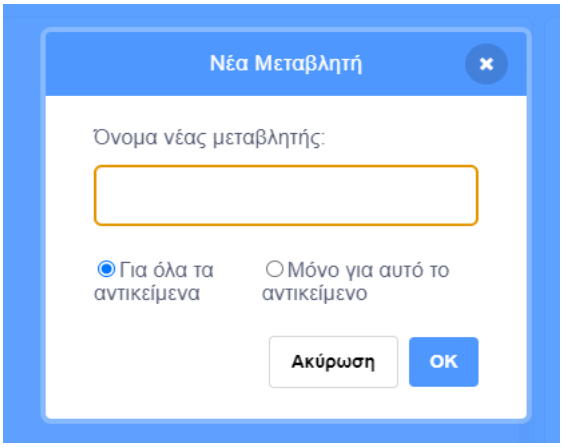


Μια μεταβλητή (δηλαδή μια θέση μνήμης) μπορεί να έχει μόνο μια τιμή κάθε φορά. Η τιμή μιας μεταβλητής είναι η τελευταία που έχεις εκχωρήσει σε αυτή.

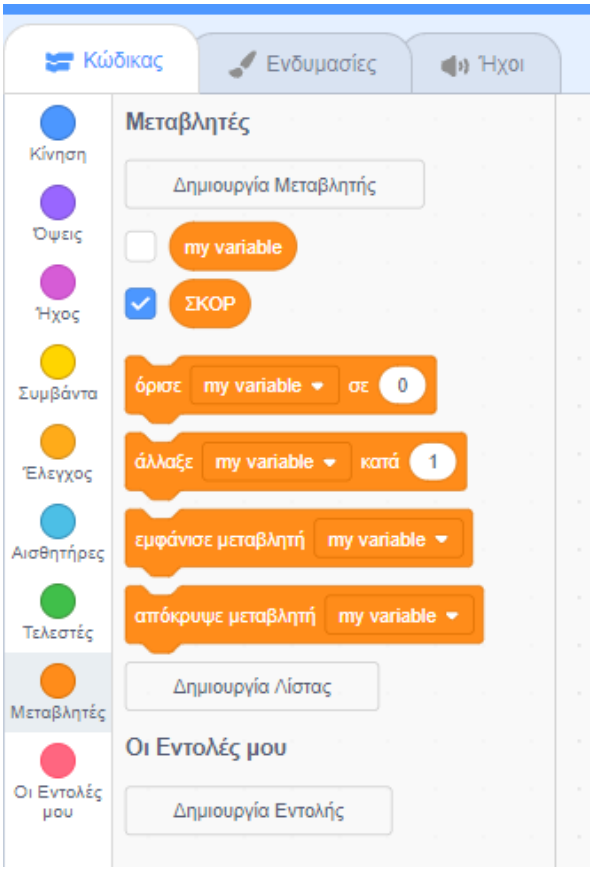
❖ Στο Προγραμματιστικό περιβάλλον του **Scratch** για να δημιουργήσουμε μια μεταβλητή πρέπει να μεταβούμε στην πιο κάτω καρτέλα εντολών:



❖ Επιλέγουμε το κουμπί **Δημιουργία Μεταβλητής** και στη συνέχεια δίνουμε ένα **<u>όνομα</u>** σε αυτήν. Π.χ. ΣΚΟΡ, ΖΩΕΣ, ΧΡΟΝΟΜΕΤΡΟ κλπ.

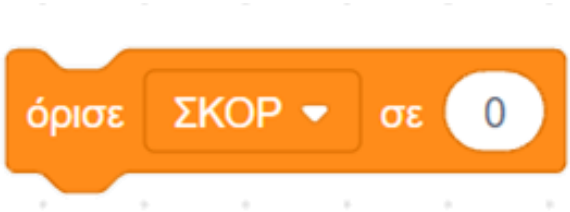


❖ Στη συνέχεια η μεταβλητή που δημιουργήσαμε εμφανίζεται στην καρτέλα εντολών μας ως εξής:



- **<u>1η ΒΑΣΙΚΗ ΕΝΤΟΛΗ ΓΙΑ ΤΙΣ ΜΕΤΑΒΛΗΤΕΣ</u>**: Είναι η εντολή με την οποία ορίζουμε ή αλλιώς δίνουμε μια αρχική τιμή σε αυτήν.

  Παράδειγμα:



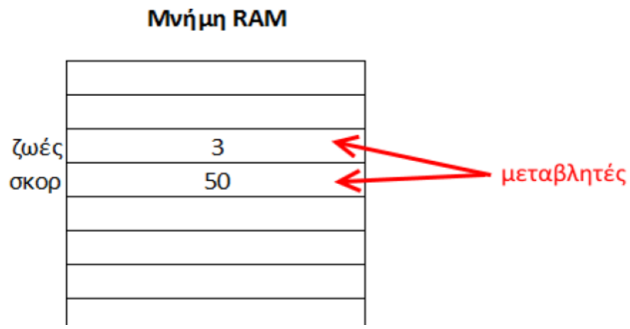
- **<u>2η ΒΑΣΙΚΗ ΕΝΤΟΛΗ ΓΙΑ ΤΙΣ ΜΕΤΑΒΛΗΤΕΣ</u>**: Είναι η εντολή με την οποία αλλάζουμε την τιμή μιας μεταβλητής, διαγράφοντας την προηγούμενή της τιμή.

  Παράδειγμα:

- Επιπλέον **μεταβλητές** (έτοιμες) στο Προγραμματιστικό εργαλείο του Scratch είναι οι ακόλουθες: **ΑΠΑΝΤΗΣΗ, ΧΡΟΝΟΜΕΤΡΟ, ΕΝΤΑΣΗ, ΟΝΟΜΑ ΧΡΗΣΤΗ, ΤΡΕΧΩΝ (ΕΤΟΣ, ΜΗΝΑΣ, ΜΕΡΑ, ΩΡΑ κλπ.).**

Λίστες Scratch

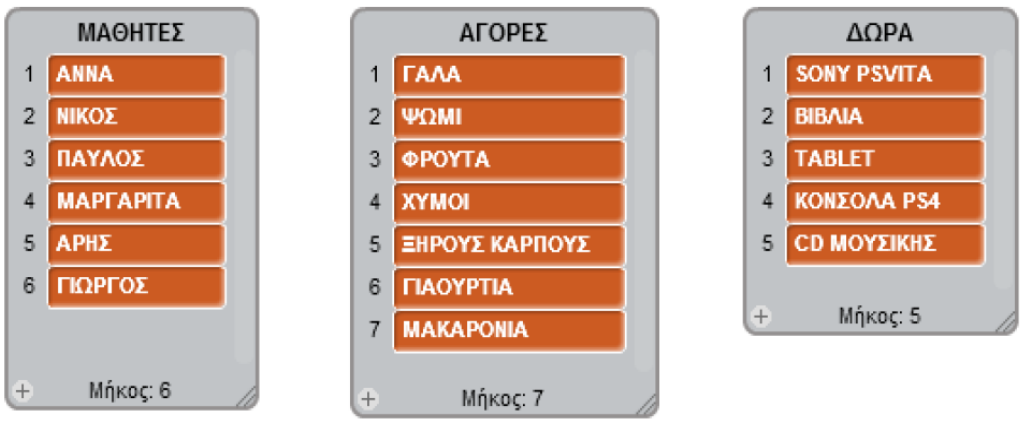
Σε αρκετές δραστηριότητες μέχρι τώρα χρησιμοποιήσαμε μεταβλητές. Οι ***μεταβλητές*** είναι συμβολικά ονόματα που αντιστοιχούν σε θέσεις μνήμης του υπολογιστή. Οι μεταβλητές χρησιμοποιούνται για την αποθήκευση και διαχείριση δεδομένων σε ένα πρόγραμμα (π.χ. σε μια μεταβλητή μπορείς να κρατάς το σκορ ενός παιχνιδιού).



Μια μεταβλητή (δηλαδή μια θέση μνήμης) μπορεί να έχει μόνο μια τιμή κάθε φορά. Η τιμή μιας μεταβλητής είναι η τελευταία που έχεις εκχωρήσει σε αυτή.

Εκτός από τις μεταβλητές, το Scratch σου δίνει τη δυνατότητα να χρησιμοποιείς ***λίστες***. Σε μια ***λίστα*** μπορείς να αποθηκεύσεις πολλές τιμές και να αναφέρεσαι σε αυτές με ένα κοινό όνομα. Συνήθως οι τιμές μιας λίστας έχουν τα ίδια χαρακτηριστικά. Μια λίστα θα μπορούσε να περιέχει:

- τα ονόματα των μαθητών που θα πάνε πενθήμερη εκδρομή
- τι να αγοράσω από το σούπερ μάρκετ
- τα δώρα που θέλω να αγοράσω για τα Χριστούγεννα



Χαρακτηριστικά μιας λίστας:

- Οι λίστες είναι ένας ευέλικτος τύπος δεδομένων, που μπορεί να γραφτεί ως μια σειρά τιμών με ένα κοινό όνομα.
- Οι λίστες στις γλώσσες προγραμματισμού πρέπει να περιέχουν στοιχεία ίδιου τύπου. Μπορεί δηλαδή μια λίστα να περιέχει μόνο αριθμούς ή μόνο χαρακτήρες – συμβολοσειρές ή μόνο λογικές τιμές.
- Κάθε στοιχείο αποθηκεύεται σε μια θέση μέσα στη λίστα. Η εύρεση της θέσης γίνεται με τη βοήθεια ενός «δείκτη».
- Οι λίστες είναι μεταλλάξιμες ή τροποποιήσιμες. Μπορούμε να αλλάξουμε την τιμή των στοιχείων και μετά τη δημιουργία τους.

Βασικές ενέργειες σε μια λίστα:

- Δημιουργία λίστας
- Εισαγωγή στοιχείων
- Επεξεργασία στοιχείων
- Διαγραφή στοιχείων λίστας & ολόκληρης της λίστας

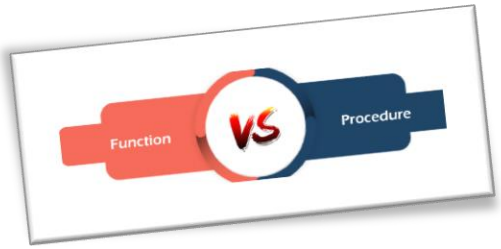
Υποπρογράμματα (My Blocks) Scratch

- **Τι ονομάζεται υποπρόγραμμα ;**

❑ Το υποπρόγραμμα είναι ένα τμήμα προγράμματος γραμμένο ξεχωριστά από το κυρίως πρόγραμμα, που είναι υπεύθυνο για την εκτέλεση ενός κώδικα που επιτελεί έναν σκοπό.

❑ Οι βασικές κατηγορίες υποπρογραμμάτων είναι:

1. Οι **διαδικασίες (procedures).**
2. Οι **συναρτήσεις (functions).**



- **Πιο συγκεκριμένα:**

❑ Η κλήση ενός υποπρογράμματος, μέσα στο κυρίως πρόγραμμα, γίνεται μέσω του ονόματός του.

❑ Τα υποπρογράμματα μπορεί να επιστρέφουν στο κυρίως πρόγραμμα τιμή ή τιμές, χρησιμοποιώντας το όνομά τους. Ειδικότερα, η συνάρτηση (function) επιστρέφει μία μόνο τιμή, ενώ η διαδικασία (procedure) παραπάνω από μία και ακόμη σε μια συνάρτηση (function) πρέπει να δηλώνουμε τον τύπο δεδομένων που επιστρέφει κάθε φορά ενώ σε μια διαδικασία (procedure) αυτό δεν ισχύει.

❑ Κάθε υποπρόγραμμα μπορεί να καλεί ένα άλλο υποπρόγραμμα.

❑ Τα υποπρογράμματα καλό είναι να μην έχουν μεγάλο μέγεθος, για να μπορούν να περιοριστούν τα λάθη.

❑ Κατά την εκτέλεση ενός υποπρογράμματος, το αρχικό πρόγραμμα σταματάει να εκτελείται προσωρινά μέχρι την ολοκλήρωσή του.

❑ Τα υποπρογράμματα είναι πολύ χρήσιμα, γιατί επιτρέπουν στον προγραμματιστή να χρησιμοποιεί τον ίδιο κώδικα πολλές φορές μέσα στο πρόγραμμά του, χωρίς να πρέπει να τον δημιουργήσει ξανά.