



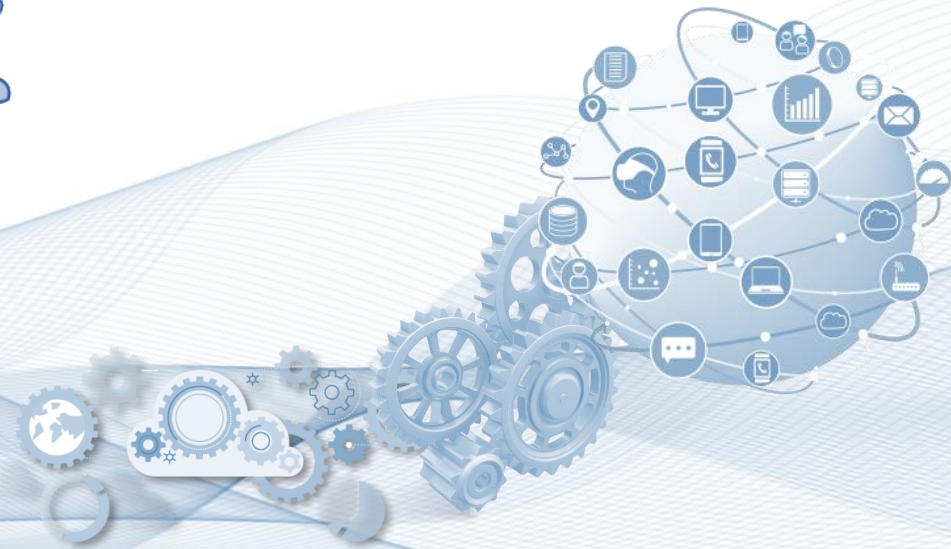
ΤΕΧΝΙΚΕΣ ΠΡΟΓΡΑΜΜΑΤΙΜΟΥ ΥΠΟΛΟΓΙΣΤΩΝ

Διάλεξη II

Το Τελικό Βήμα με Scratch – Μεταβλητές & Πρότυπα



Γιάννης Τζήμας, Καθηγητής



Τι θα δούμε σήμερα...

- Προχωρώντας θα δούμε πιο αναλυτικά τις έννοιες των:
 - μεταβλητών
 - και των προτύπων.



Τι είδαμε στα προηγούμενα μαθήματα και τι μας λείπει;

- Οι **τρεις** προγραμματιστικές δεξιότητες που θα πρέπει να καλλιεργήσετε όπως έχουμε πει είναι:
 - Η **αλληλουχία** των εντολών
 - Η **επιλογή** ανάμεσα σε διάφορες εναλλακτικές και
 - Η **επανάληψη**

Ποιο είναι το **επόμενο βήμα**;
Στην πράξη χρειαζόμαστε επιπλέον:

- Ένα τρόπο για να θυμόμαστε πράγματα – “**variables – μεταβλητές**”
- Δεξιότητες για επίλυση προβλημάτων – “**patterns - πρότυπα**”



Τύποι Μνήμης

- Μόνιμη

- Τα δεδομένα υπάρχουν ακόμα και αν η συσκευή κλείσει.
- π.χ. σκληροί δίσκοι, memory cards κλπ.

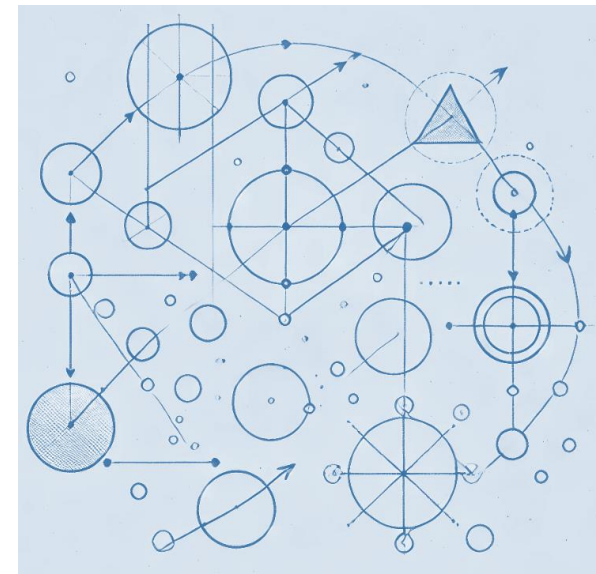
- Προσωρινή

- Τα δεδομένα αποθηκεύονται προσωρινά ώστε να γίνει η απαραίτητη επεξεργασία.
- π.χ. RAM – Random Access Memory



Μεταβλητές (Variables) = Μνήμη (Memory)

- Οι μεταβλητές είναι η **μνήμη** του προγράμματός σας.
 - Τις χρησιμοποιούμε για να αποθηκεύσουμε πληροφορίες όσο το πρόγραμμα «τρέχει».
- **Γιατί** χρειαζόμαστε μεταβλητές;
 - Μερικά παραδείγματα...
 - Για να κάνουμε **μαθηματικούς υπολογισμούς** – «η απόσταση έως την άκρη της σκηνής είναι...»
 - Για να **παρακολουθούμε** διάφορα πράγματα – «το σκορ του παιχνιδιού είναι...»
 - Για να **αποθηκεύουμε τιμές εισόδου** από το χρήστη – «Παρακαλώ πληκτρολογήστε το όνομά σας...»
 - Για να **κρατάμε τιμές εξόδου** που μπορεί να δείξουμε στην οθόνη – «Η σωστή απάντηση στην ερώτηση είναι...»
- **Ενδιαφέρουσα σημείωση:**
 - Στις περισσότερες γλώσσες προγραμματισμού (π.χ. Java, C++, C# κλπ.), οι μεταβλητές χρησιμοποιούν την **προσωρινή μνήμη** – οι πληροφορίες χάνονται με τον τερματισμό του προγράμματος.
 - Στο **Scratch** οι μεταβλητές χρησιμοποιούν τη **μόνιμη**.

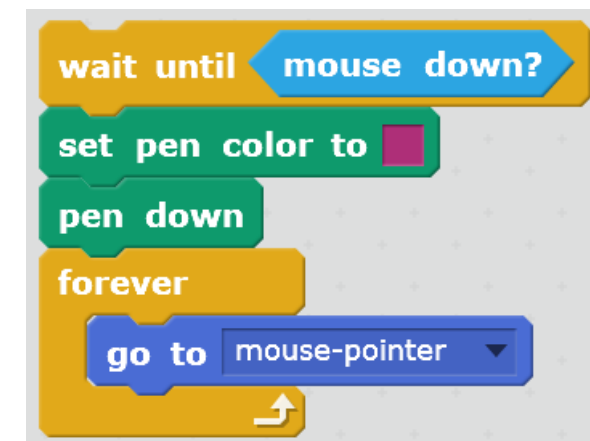
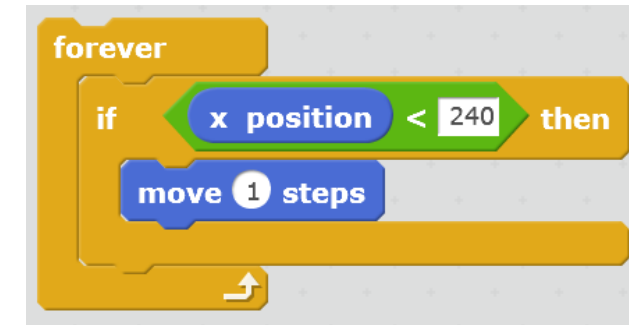


Παραδείγματα στο Scratch

- Έχουμε ήδη χρησιμοποιήσει **προκαθορισμένες μεταβλητές** στα προγράμματα που φτιάξαμε...
 - x position
 - y position
 - direction

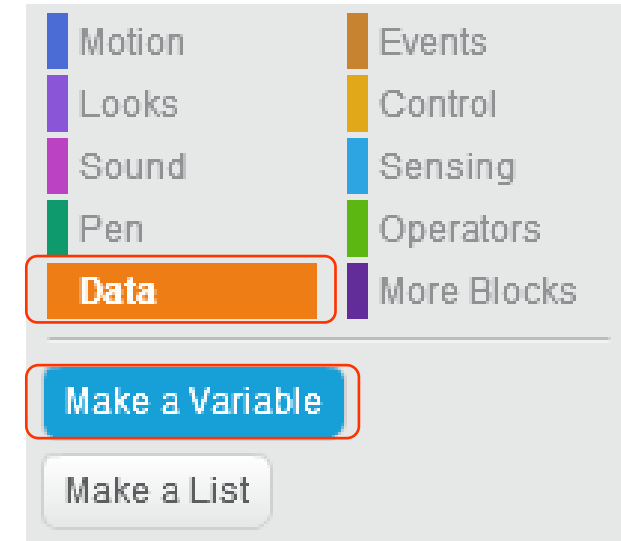
- pen color
- pen state (up? down?)
- mouse position
- mouse state

- volume
-



Δημιουργώντας τις Μεταβλητές σας...

- Δημιουργείστε μία μεταβλητή όταν θέλετε να παρακολουθήσετε κάτι...
- Οι μεταβλητές έχουν διάφορα **χαρακτηριστικά**:
 - Κάθε μεταβλητή έχει **όνομα**.
 - Μία μεταβλητή περιέχει **μία τιμή**.
 - Το **περιεχόμενο** μίας μεταβλητής μπορεί να **αλλάξει**.
 - Μία μεταβλητή **συσχετίζεται** με **ένα** (τοπική/**local variable**), ή με **όλα** τα **sprites** (καθολική/**global variable**).
 - Μία μεταβλητή μπορεί να **φαίνεται** (π.χ. το σκορ), ή να **μη φαίνεται** στη σκηνή.



Ένα μικρό demo...

- Δημιουργώντας μεταβλητές:
 - Πόσα **βήματα** πρέπει να γίνουν έως η Scratch να αγγίξει τη γωνία;
 - Ποια είναι η **απόσταση** από τη δεξιά πλευρά;
 - Ποιο είναι το **όνομα** του χρήστη (επώνυμο, όνομα);
 - Ποιο είναι το **όνομα** και η **χρονιά** που γεννήθηκε ο χρήστης;
 - Υπολογισμός της ηλικίας

The image shows the Scratch Variables palette. At the top, there are buttons for Motion, Control, Looks, Sensing, Sound, Operators, Pen, and Variables (highlighted with a dashed orange border). Below these are buttons for 'Make a variable' and 'Delete a variable'. A list of variables is shown with checkboxes: Age, Name, Distance, and Steps. Below the list, a script is shown with four blocks: 'set Age to 0', 'change Age by 1', 'show variable Age', and 'hide variable Age'. The entire script area is enclosed in a dashed red border.

Στην πράξη...

```
set Steps to 0
repeat until touching edge?
  move 1 steps
  change Steps by 1
say Steps for 4 secs
```


```
set Distance to 240 - x position
say Distance for 4 secs
```

```
ask What's your name? and wait
set Name to answer
say join Hi Name for 2 secs
```


```
ask What's your name? and wait
set Name to answer
say join Hi Name for 2 secs
ask What year were you born? and wait
set Age to 2013 - answer
say join I'm guessing your age is Age for 4 secs
```



Name **Joe**
Age **11**
Sprite 1 Steps **212**
Sprite 1 Distance **256.9**



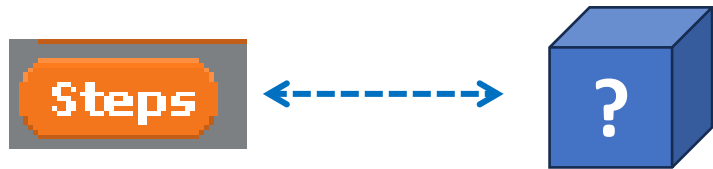
Name **Γιάννης**
Age **11**
Sprite 1 Steps **212**
Sprite 1 Distance **29.9**



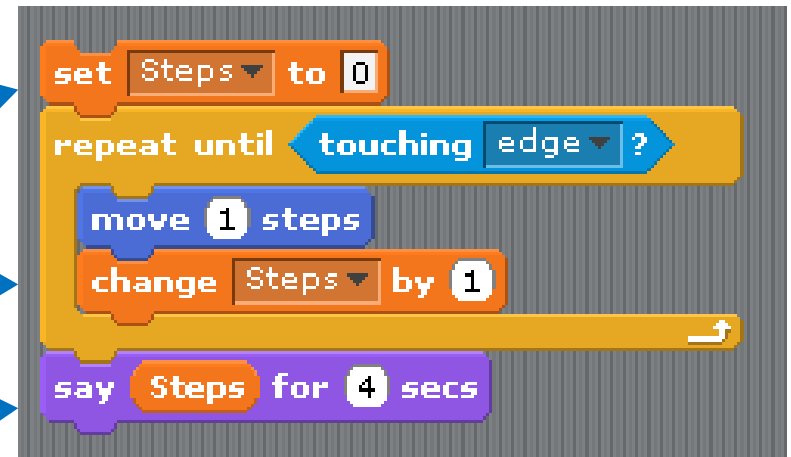
lecture2_01.sb3

Όταν χρησιμοποιείτε λοιπόν μεταβλητές...

- Να σκέφτεστε τη μεταβλητή σαν ένα **κουτί χωρίς αρχική τιμή**:



- Θα πρέπει να **δώσετε** μία **αρχική τιμή**,
- καθώς το πρόγραμμα τρέχει να **ενημερώνετε** την τιμή αυτή
- και στο τέλος να τη **χρησιμοποιείτε**, ή να την **παρουσιάζετε**.



Χρυσός Κανόνας: Χρησιμοποιείτε μία μεταβλητή για κάθε τμήμα δεδομένων που θα πρέπει να παρακολουθείτε!

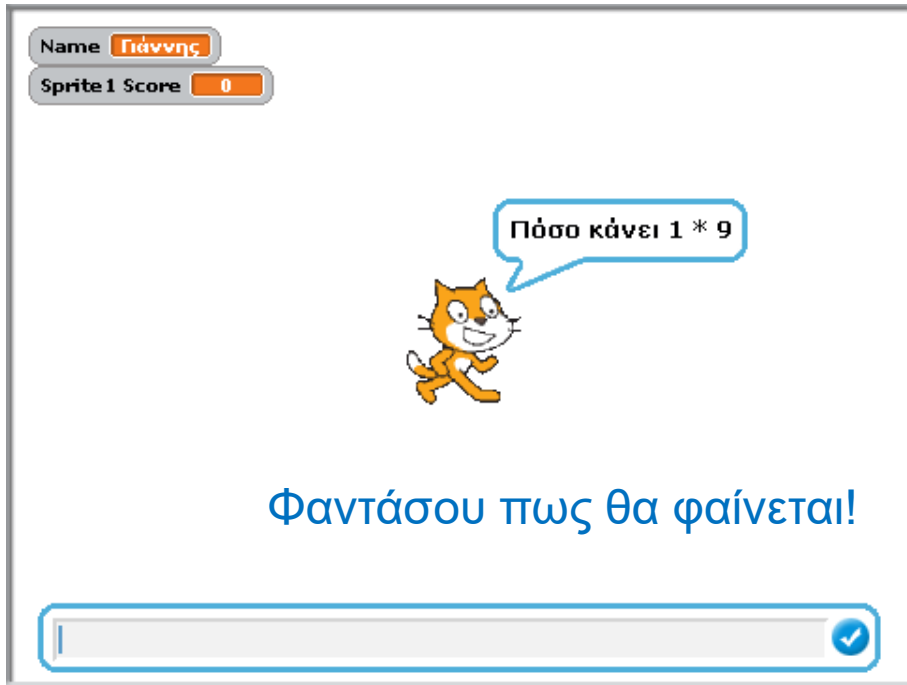
Ένα ολοκληρωμένο παράδειγμα...

- Ας χρησιμοποιήσουμε όλα όσα μάθαμε μέχρι στιγμής για να φτιάξουμε ένα απλό **παιχνίδι πολλαπλασιασμού**:
- Το πρόγραμμα θα κάνει τα ακόλουθα:
 - Επιλέγει δύο τυχαίους αριθμούς X και Y
 - Υπολογίζει το σωστό αποτέλεσμα " $X * Y$ "
 - Στη συνέχεια ρωτάει τον παίκτη: "Πόσο κάνει $X * Y$;"
 - Συγκρίνει την απάντηση με το σωστό αποτέλεσμα
 - Αν είναι σωστή η απάντηση, τότε ο παίκτης παίρνει ένα βαθμό
 - Επανάληψη...
 - Όταν το παιχνίδι τελειώνει εμφανίζεται το σκορ

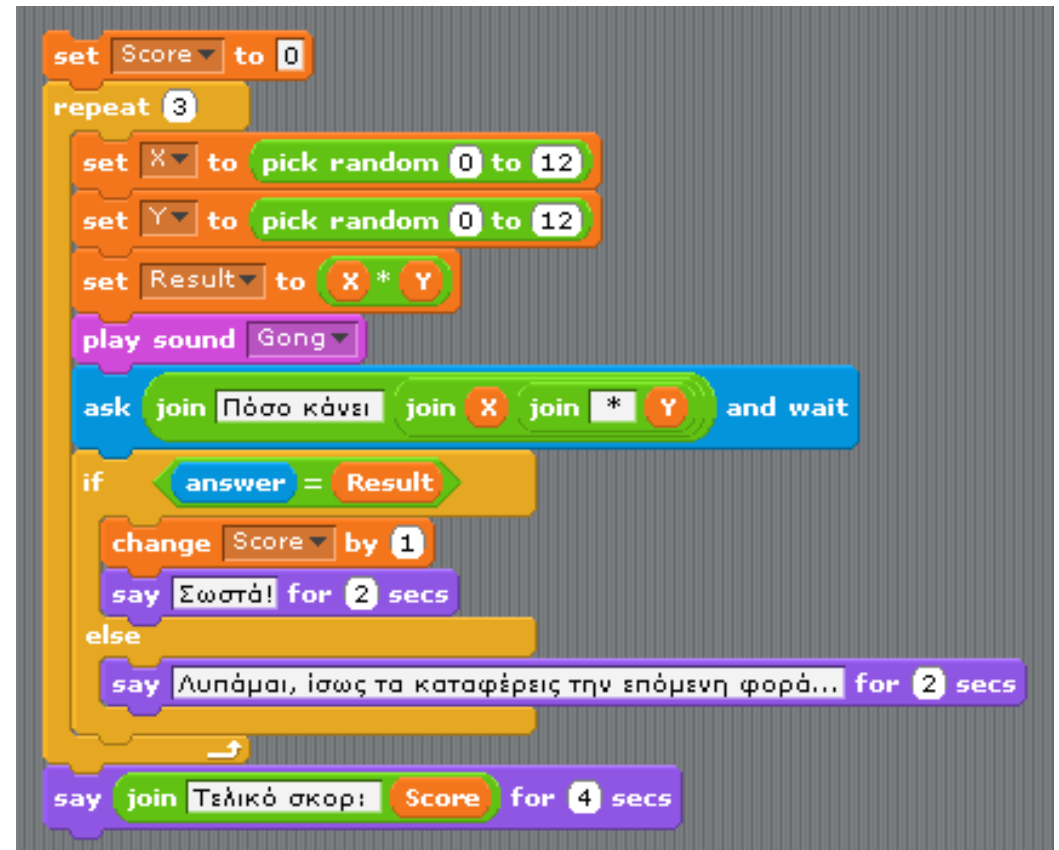
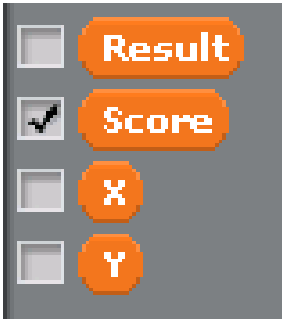


Σχεδιάζοντας τη λύση...

- Επανέλαβε 3 φορές:
 - Επέλεξε δύο τυχαίους αριθμούς, και ονόμασέ τους X και Y
 - Υπολόγισε το σωστό αποτέλεσμα " $= X * Y$ "
 - Ρώτα τον παίκτη: "Πόσο κάνει $X * Y$;"
 - Σύγκρινε την απάντηση με το σωστό αποτέλεσμα
 - Αν είναι σωστή η απάντηση, τότε ο παίκτης παίρνει ένα βαθμό
- Όταν το παιχνίδι τελιώνει εμφανίζεται το σκορ



- Πόσες μεταβλητές χρειαζόμαστε;
 - Τουλάχιστον τέσσερις
 - X, Y, Σωστό Αποτέλεσμα, Σκορ
- Η Αλληλουχία:



Συνολικά

- Name
- Result
- Score
- X
- Y

```
when clicked
ask Ποιο είναι το όνομά σου; and wait
set Name to answer
say join Ετοιμάσου Name for 2 secs
set Score to 0
repeat 3
set X to pick random 0 to 12
set Y to pick random 0 to 12
set Result to X * Y
play sound Gong
ask join Πόσο κάνει join X join * Y and wait
if answer = Result
change Score by 1
say Σωστά! for 2 secs
else
say Λυπάμαι, ίσως τα καταφέρεις την επόμενη φορά... for 2 secs
say join Τελικό σκορ: Score for 4 secs
```

Name Γιάννης

Sprite 1 Score 0

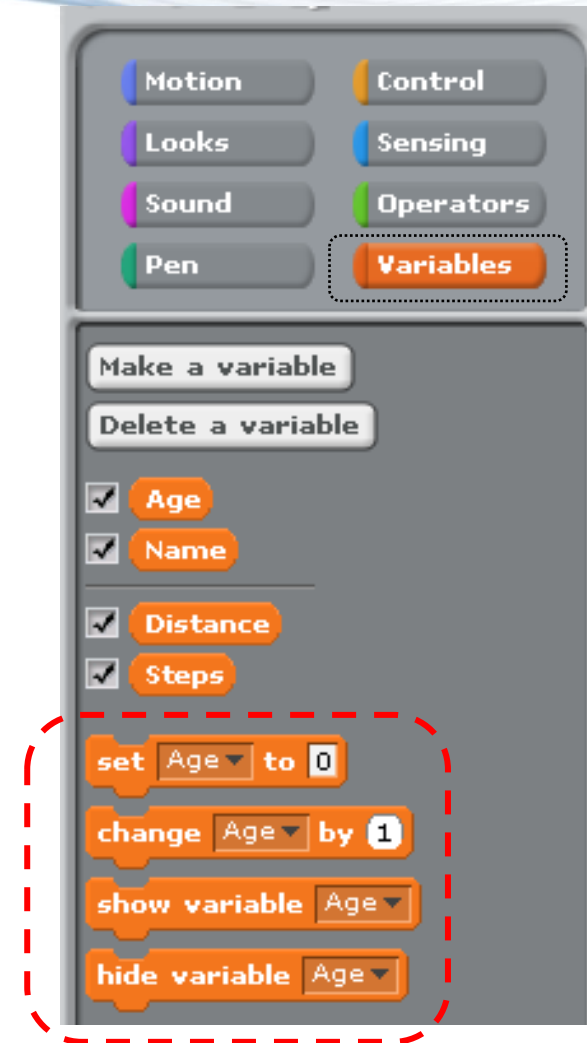
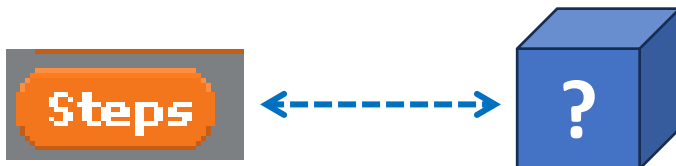


Πόσο κάνει 1 * 8

lecture3_demo2.sb3

Τι είδαμε μέχρι τώρα;

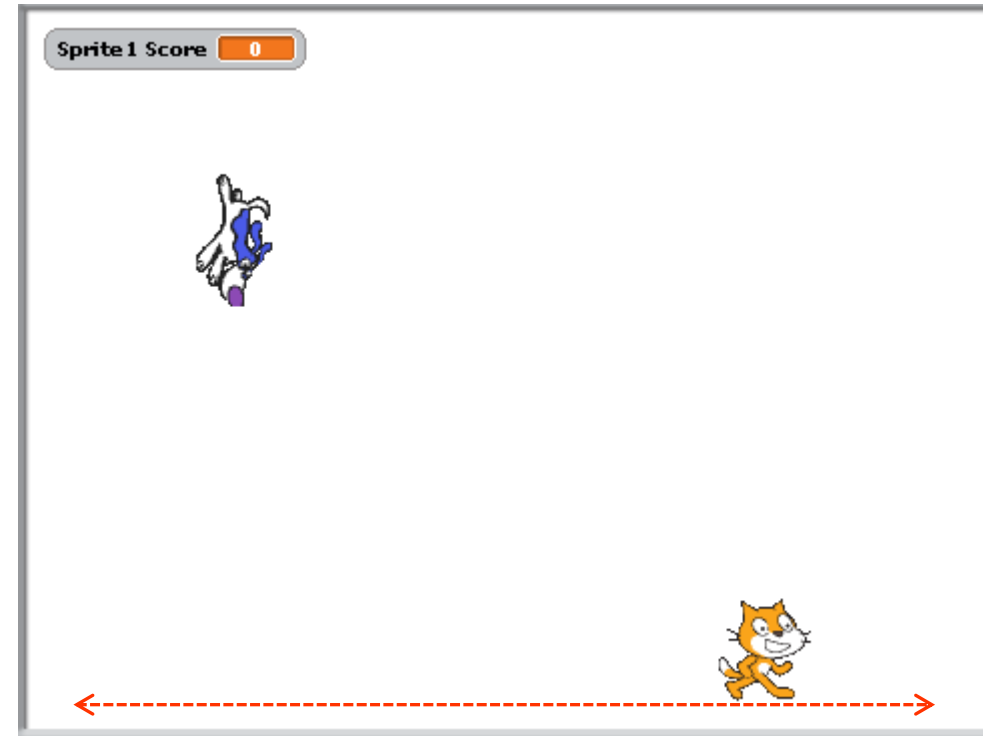
- Οι **μεταβλητές** είναι η μνήμη του προγράμματος:
 - Αποθηκεύουν πληροφορίες ενώ το πρόγραμμα «τρέχει».
 - Στο **Scratch**, οι μεταβλητές δε χρησιμοποιούν την προσωρινή μνήμη, σε αντίθεση με άλλες γλώσσες.
- **Σημαντικός κανόνας:**
 - Χρησιμοποιείτε **μία μεταβλητή για κάθε τμήμα δεδομένων** που θα πρέπει να παρακολουθείτε!
- **Μην ξεχνάτε:**
 - **Αρχικοποιείτε** τις μεταβλητές πριν την πρώτη χρήση τους...



Και λίγη δουλειά για το σπίτι... (1/2)

- Άσκηση 1:

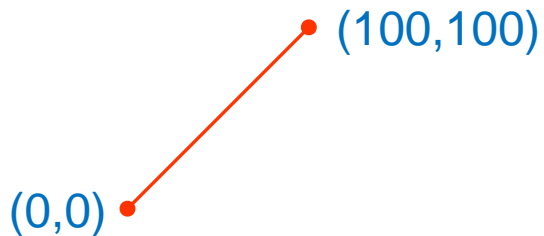
- Η πρώτη βασίζεται στην εργασία του προηγούμενου μαθήματος που φτιάξαμε ένα πρόγραμμα που έβρεχε γάτες και σκύλους.
- Αυτή τη φορά θέλουμε η Scratch να μπορεί να μετακινηθεί αριστερά και δεξιά στο κάτω μέρος της σκηνής, προσπαθώντας να πιάσει ένα σκυλάκι το οποίο πέφτει από πάνω προς τα κάτω. Για κάθε σκυλάκι που πιάνει η Scratch θα κερδίζει ένα πόντο.



Και λίγη δουλειά για το σπίτι... (2/2)

- Άσκηση 2:

- Υπολογίστε την κλίση μίας γραμμής. Η κλίση συντίθεται από δύο πράγματα:
 - Πόσο απότομη είναι και
 - Από το αν ανεβαίνει ή κατεβαίνει
- Για παράδειγμα η γραμμή αριστερά έχει κλίση 1, ενώ η γραμμή δεξιά 0.



<http://en.wikipedia.org/wiki/Slope>

- Δεδομένων δύο σημείων (x_1, y_1) και (x_2, y_2) η κλίση υπολογίζεται από τον ακόλουθο τύπο:
$$\text{slope} = (y_2 - y_1) / (x_2 - x_1)$$
- Γράψτε ένα πρόγραμμα που δέχεται σαν είσοδο τα x_1, y_1, x_2 και y_2 ζωγραφίζει μία γραμμή και υπολογίζει την κλίση της.
Προσοχή στις κάθετες γραμμές!

Η κλίση είναι -0.4



Προγραμματιστικές Δεξιότητες

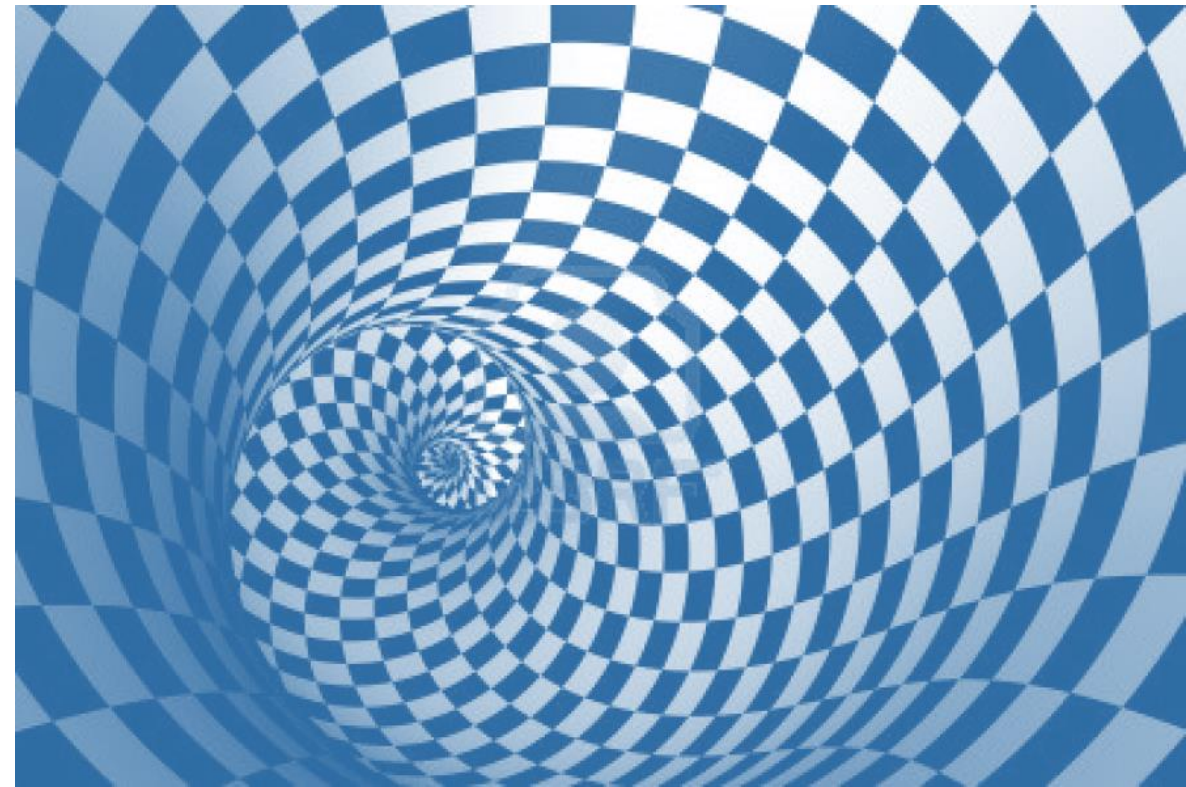
- Σε **λογικό** επίπεδο:
 - Αλληλουχία των εντολών
 - Επιλογή ανάμεσα σε διάφορες εναλλακτικές
 - Επανάληψη
- Σε **πρακτικό** επίπεδο:
 - Μεταβλητές
 - Εμπειρία
 - Αποφασιστικότητα
 - Οι Η/Υ μπορεί να σας δημιουργήσουν απογοήτευση, αλλά...

μην τα παρατήσετε!



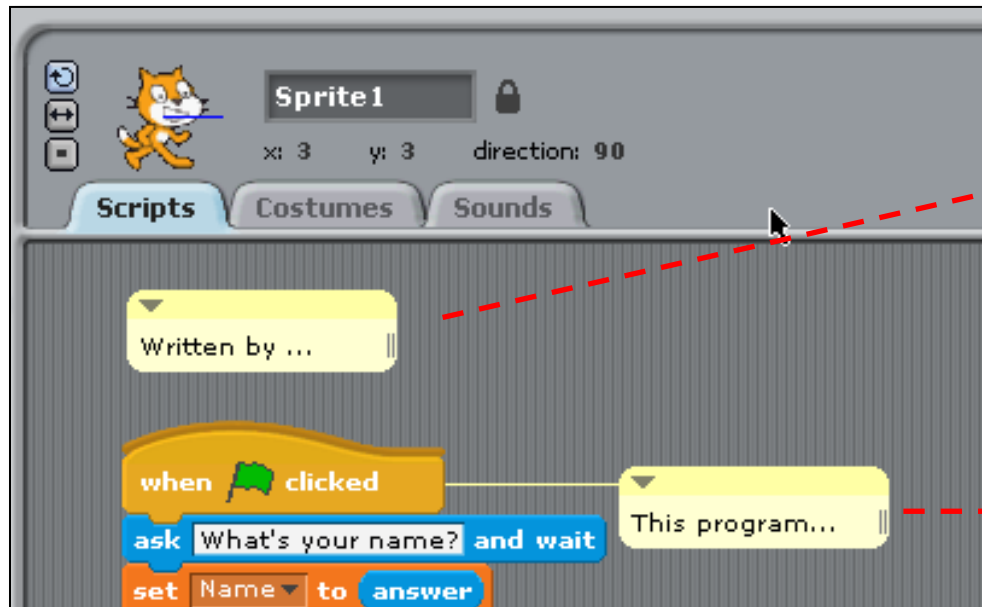
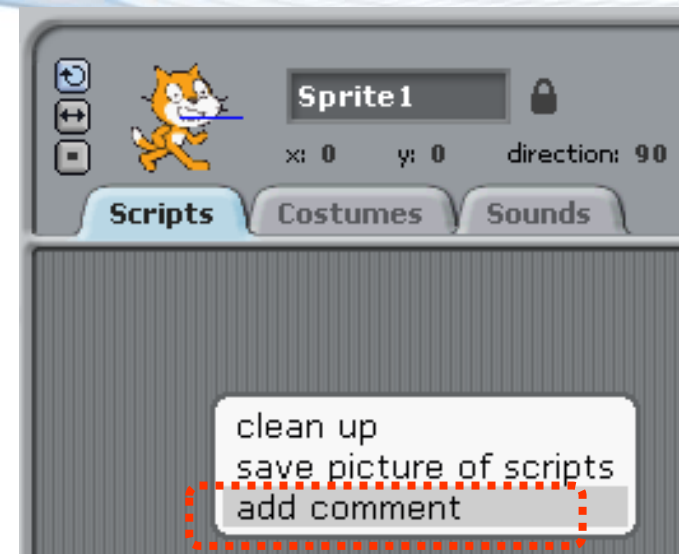
Πρότυπα (patterns) = Εμπειρία (experience)

- Τα πρότυπα είναι ένας τρόπος σύλληψης της συλλογικής εμπειρίας άλλων.
 - “standing on the shoulders of giants”
- **Απλά** patterns:
 - commenting on your programs
 - centralized code
 - regression testing
- Μερικά πιο **περίπλοκα**:
 - game play
 - count, sum, and compute



#1 Σχολιασμός

- Καθώς τα προγράμματα γίνονται όλο και πιο περίπλοκα είναι πολύ δύσκολο να θυμάστε όλες τις λεπτομέρειες!
 - Μην το προσπαθήσετε καν!
- Αντί για αυτό, προσθέστε **σχόλια**
 - Τα σχόλια είναι ένας τρόπος **μόνιμης καταγραφής των σκέψεών σας**, ως μέρος του προγράμματος
 - Τα σχόλια αγνοούνται από το Scratch και από όλες τις γλώσσες προγραμματισμούς – είναι φτιαγμένα **για να τα διαβάζουμε εμείς**.



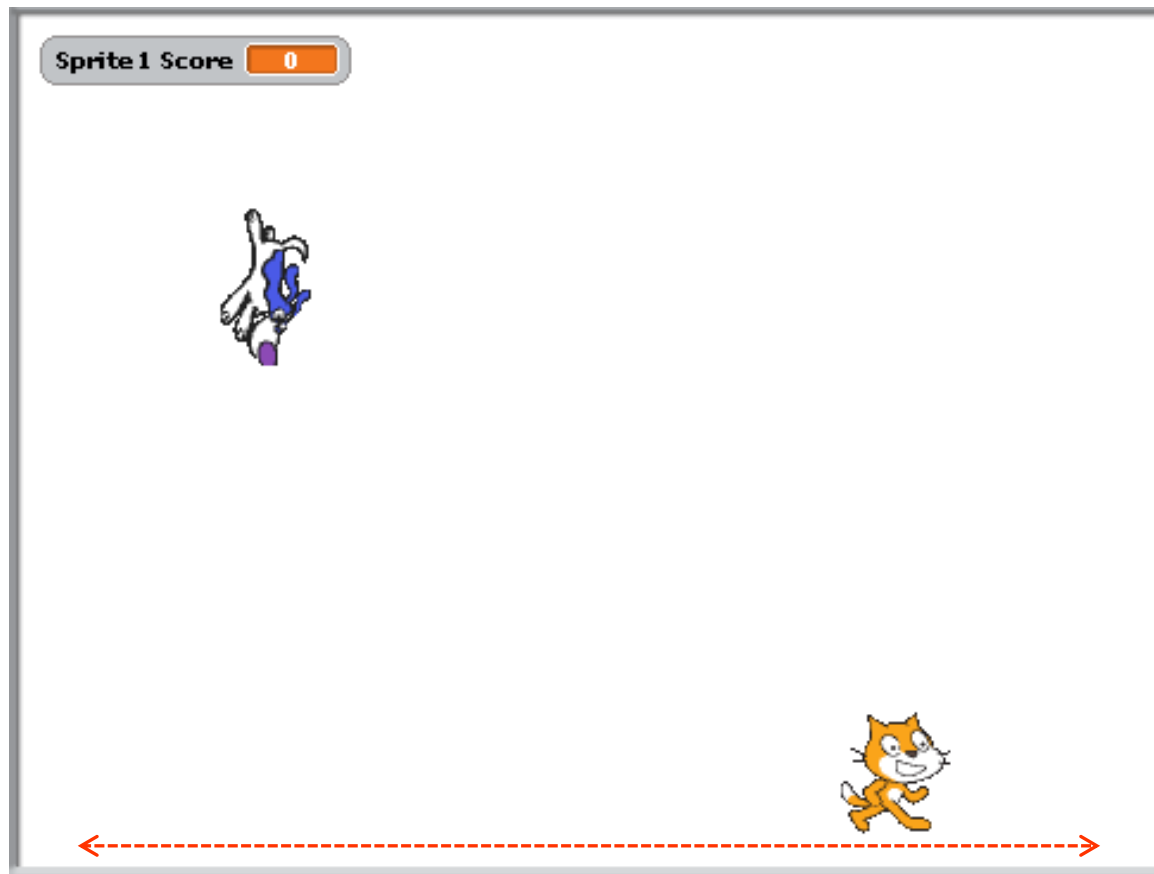
standalone

associated with a block



Ένα μικρό demo...

- Από την πρώτη άσκηση...



A screenshot of a Scratch script area. The script consists of the following blocks:

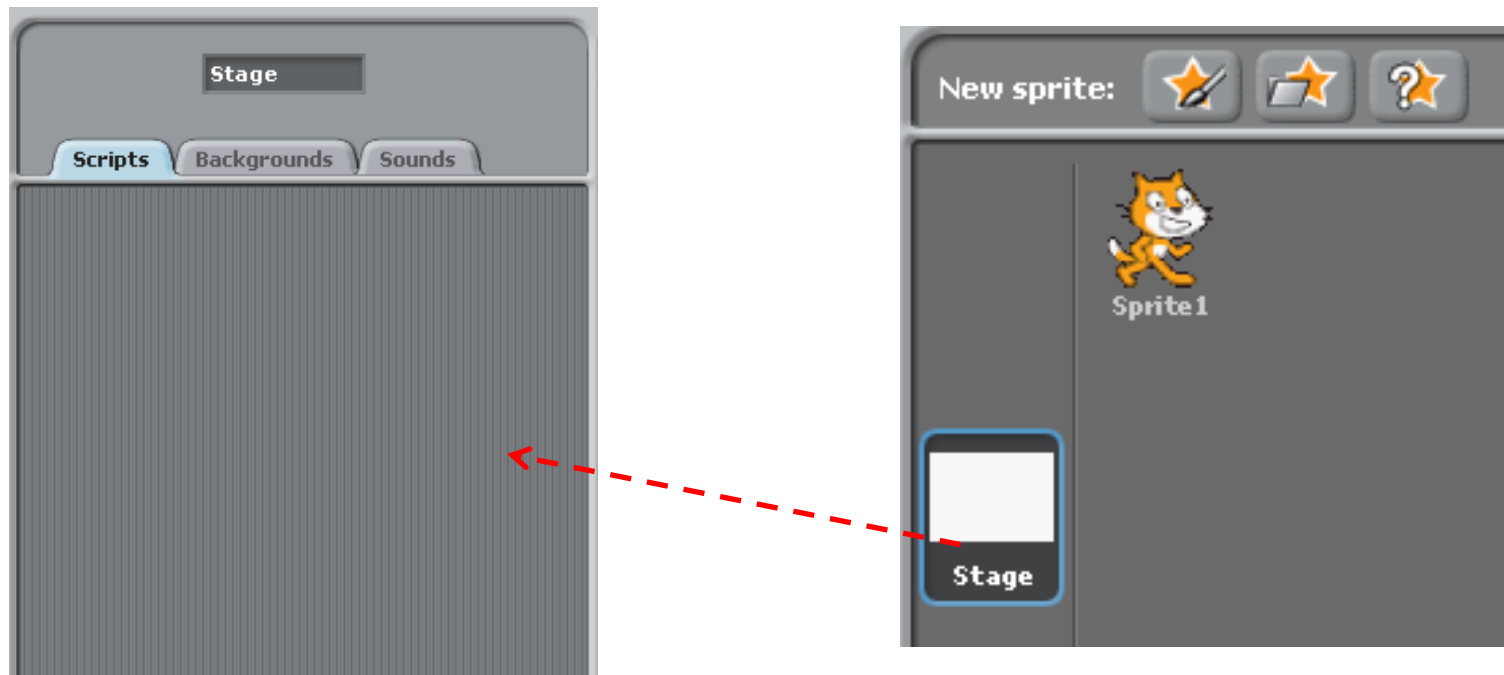
- when green flag clicked
- set volume to 20 %
- forever loop containing:
 - play sound WaterRunning
 - rest for 1.2 beats

There are two yellow sticky notes with Greek text:

- The top note says: "Με αυτό το script η Scratch θα προσπαθήσει να πιάσει το σκυλί καθώς αυτό πέφτει" (With this script, Scratch will try to catch the dog as it falls).
- The bottom note says: "Θα πρέπει να ακούγεται σαν βροχή!" (It should sound like rain!).

#2 Centralized Code

- Κάποιος κώδικάς μας είναι **γενικού σκοπού** – π.χ. δεν αναφέρεται σε κάποιο sprite
 - “παίξε ένα τραγούδι όσο το παιχνίδι τρέχει”
 - “παρακολουθήσε αν θα πατηθεί κάποιο κουμπί για να αλλάξει η κατάσταση του παιχνιδιού”
 - “σταμάτα το παιχνίδι μετά από Χ δευτερόλεπτα...”
- Στο Scratch, το "stage" (σκηνή) μπορεί επίσης να περιέχει scripts
 - Αυτό είναι ένα πολύ καλό μέρος για να μπει ο **κώδικας γενικού σκοπού**...



Ένα μικρό demo...

- Από την πρώτη άσκηση...
 - Ας κεντριοποιήσουμε τον κώδικα και
 - ας βάλουμε ένα χρονόμετρο.



lecture3_03.sb3



#3 Regression Testing

- Ο έλεγχος των προγραμμάτων είναι δύσκολος
 - Θα πρέπει να **σκεφτείτε πολύ** για όλα όσα το πρόγραμμά σας πρέπει να κάνει και να **επινοήσετε ελέγχους**, ώστε να σιγουρευτείτε ότι τα κάνει σωστά.
 - Θα πρέπει να γίνετε το ίδιο καλός/ή στο να «**σπάτε**» προγράμματα (να βρίσκετε τα λάθη τους), όσο είστε στο να τα **δημιουργείτε**.
- Έστω ότι επεξεργάζεστε και αλλάζετε το πρόγραμμά σας – τότε τι θα πρέπει να κάνετε;
 - Θα πρέπει να κάνετε **όλους τους ελέγχους ξανά**.
 - Αυτή η διαδικασία λέγεται **regression testing**.



Κανόνας: Θα πρέπει να διαθέσετε τόσο χρόνο στον έλεγχο του προγράμματος, όσο έχετε διαθέσει και στη δημιουργία του.

Ένα μικρό demo...

- Ας δούμε την άσκηση 2 με την κλήση....



```
set Slope to Rise / Run
say join The slope is Slope for 4 secs
```

Θα αποτύχει στη διαίρεση με το 0...

```
if Run = 0
  say The slope of a vertical line is undefined! for 4 secs
else
  set Slope to Rise / Run
  say join The slope is Slope for 4 secs
```

“Φρουρήστε” για να μη γίνει αυτή η διαίρεση...

Αν δεν κάνουμε τον έλεγχο...

```
when clicked
clear
go to x: -200 y: 120
hide
ask X1? and wait
set X1 to answer
ask Y1? and wait
set Y1 to answer
ask X2? and wait
set X2 to answer
ask Y2? and wait
set Y2 to answer
set pen color to blue
set pen size to 5
go to x: X1 y: Y1
pen down
go to x: X2 y: Y2
pen up
go to x: -200 y: 120
show
set Rise to Y2 - Y1
set Run to X2 - X1
set Slope to Rise / Run
say join The slope is Slope for 4 secs
```



lecture2_04.sb3



#4 Game Play

- Μία από τις προκλήσεις των παιχνιδιών είναι ο **συντονισμός** όλων των sprites.
 - Όταν *αλλάζουμε επίπεδα*, πως το ξέρουν όλα τα sprites;
 - Όταν *τελειώνει το παιχνίδι*, πώς σταματάνε όλα τα sprites;
 - Πρέπει κάθε sprite να ξέρει *πως τελειώνει το παιχνίδι*, ή μπορούμε αυτό να το κεντροποιήσουμε;
 - Ο κώδικας που υπάρχει κεντρικά μπορεί να αλλάξει ευκολότερα στο μέλλον...



Λύση (Message Passing):

- ❖ χρησιμοποιείτε ένα **broadcast block** (στην κατηγορία *Events*) για να στείλετε ένα μήνυμα όταν κάτι συμβαίνει
- ❖ τα sprites χρησιμοποιούν ένα **when I receive block** (επίσης στην κατηγορία *Events*) για να αντιδράσουν σε ένα μήνυμα

Ένα μικρό demo...

- Θα χρησιμοποιήσουμε το παράδειγμα με την καταιγίδα ("Raining Cats and Dogs") και θα:
 - Βάλουμε ένα "Game Over!" μήνυμα στο τέλος
 - Βάλουμε ένα "Spin" μήνυμα για να περιστρέφουμε τις γάτες και τους σκύλους
 - Βάλουμε κουμπιά **αλλαγής ταχύτητας**...





Στην πράξη...

```

when clicked
  switch to background Default
  set Speed to 1
  set volume to 20 %
  forever
    play sound WaterRunning
    rest for 1.2 beats
  end

when space key pressed
  broadcast Spin

when up arrow key pressed
  change Speed by 1

when down arrow key pressed
  if Speed > 1
    change Speed by -1

when q key pressed
  switch to background Bye!
  broadcast Game Over! and wait
  stop all
  
```

```

Cat1
when clicked
  forever
    point in direction 180
    set x to pick random -160 to 230
    set y to pick random 150 to 170
    repeat until y position < -150
      move Speed steps
    end

  when I receive Spin
    repeat 24
      turn 15 degrees
    end

  when I receive Game Over!
    play sound Cat until done
  
```

lecture2_05.sb3

```

 Speed
set Speed to 0
change Speed by 1
show variable Speed
hide variable Speed
  
```



#5 Count, Sum, & Compute

- Το Scratch είναι περισσότερο από μία γλώσσα δημιουργίας παιχνιδιών
 - Το Scratch είναι ικανό για **γενικού σκοπού υπολογισμούς**
 - Παράδειγμα: “Κλίση” το παράδειγμα που κάναμε
- **Γενικά πρότυπα υπολογισμών:**
 - **μέτρηση-counting:** 1, 2, 3, ..., N
 - **άθροιση-summing:** $1 + 2 + 3 + \dots + N$
 - **υπολογισμοί-computing:** πρόσθεση, πολλαπλασιασμός, τετραγωνική ρίζα, ημίτονο, συνημίτονο, ...



Μέτρηση – Counting

- Υποθέστε ότι θέλετε να μετρήσετε από 0 .. N ανά 5
 - 0, 5, 10, 15, 20, 25, ...

- Δημιούργησε μία μεταβλητή "Count"
- Δημιούργησε μία μεταβλητή "N"
- Θέσε την Count σε 0
- Θέσε την N στην τελευταία τιμή
- Repeat until Count > N:
 - Κάνε τους υπολογισμούς με την Count
 - Άλλαξε την Count κατά 5



```
set Count to 0
set N to 30
repeat until Count > N
  say Count for 1 secs
  change Count by 5
```

Counting by 5's

lecture2_06.sb3

Άθροιση- Summing

- Υποθέστε ότι θέλετε να αθροίσετε τις τιμές 1 .. N
 - $1 + 2 + 3 + 4 + 5 + \dots + N$

- Δημιούργησε μία μεταβλητή "Count"
- Δημιούργησε μία μεταβλητή "N"
- Δημιούργησε μία μεταβλητή "Sum"
- Θέσε την Count σε 1
- Θέσε την N στην τελευταία τιμή
- Θέσε την Sum σε 0
- Repeat until Count > N:
 - Άλλαξε τη Sum κατά Count
 - Άλλαξε την Count κατά 1




```
set Count to 1
set N to 10
set Sum to 0
repeat until (Count > N)
  change Sum by Count
  change Count by 1
say Sum for 4 secs
```

lecture2_06.sb3

$$\text{Έλεγχος: } \text{sum}(1..N) = \frac{N * (N+1)}{2}$$

Υπολογισμοί – Computing

- Παράδειγμα: Νόμοι της κίνησης του Νεύτωνα
 - Αυτή η εξίσωση υπολογίζει το ύψος H ενός αντικειμένου, όταν το πετάμε ευθεία επάνω, χωρίς να υπολογίσουμε την αντίσταση του αέρα.



Ας υπολογίσουμε το ύψος που θα φτάσει μία μπάλα σε ένα ποδοσφαιρικό αγώνα.

$$H = V_0 * T - 1/2g * T^2$$

ύψος

Αρχική
ταχύτητα

χρόνος

Επιτάχυνση
της βαρύτητας
(9,8 m/sec²)

Νόμος της Κίνησης του Νεύτωνα

- Ας εφαρμόσουμε το νόμο:
 - Ο παίκτης πετάει τη μπάλα με περίπου 140m/sec
 - Ας υπολογίσουμε το ύψος με $T=1, 2, 3, 4, 5$

□ Δημιούργησε τις μεταβλητές που χρειαζόμαστε:

- μέτρηση: T, N
- υπολογισμός: H, V_0, g

□ Θέσε το g σε 9.8

□ Θέσε το V_0 σε 27.2

□ Θέσε το T σε 1

□ Θέσε το N σε 5

□ Repeat until $T > N$:

- Θέσε το H σε $V_0 * T - \frac{1}{2} * g * T * T$
- Πες H
- Αλλάξε το T κατά 1



```
set g to 9.8
set V0 to 27.2
set T to 1
set N to 5
repeat until T > N
  set H to V0 * T - 1 / 2 * g * T * T
  say join Στο T= join T join , το ύψος είναι H for 4 secs
  change T by 1
```

Υπολογισμός το ύψους που θα φτάσει μία μπάλα σε έναν αγώνα

αρχική επιτάχυνση: 27.2 m/sec

lecture2_06.sb3

Τι είδαμε μέχρι τώρα;

- Αυτό ήταν λοιπόν, **είστε έτοιμοι!** Έχετε αποκτήσει τις ακόλουθες δεξιότητες:
 - **Αλληλουχία** των εντολών
 - **Επιλογή** ανάμεσα σε διάφορες εναλλακτικές
 - Αξιοποίηση της **Επανάληψης**
 - Γνώση των **μεταβλητών**
 - Έχετε δει μερικά **πρότυπα**
 - Απλά χρειάζεστε περισσότερη **εμπειρία** και μεγάλη **αποφασιστικότητα!!!**
- Συνεχίστε να μαθαίνετε:
 - <http://scratch.mit.edu/discuss/>
- Για περισσότερη έμπνευση::
 - <http://scratch.mit.edu/explore/projects/featured/>

Are
You
Ready?

Και λίγη δουλειά για το σπίτι...

- Υπολογίστε την **πραγματική τροχιά μιας μπάλας**.
- Θεωρείστε ότι ο παίκτης στέκεται στο σημείο $(0,0)$ και πετάει τη μπάλα με **θετική x και y κατεύθυνση** με μία **γωνία A** .
- Για να υπολογιστεί η θέση της μπάλας τη **χρονική στιγμή T** χρησιμοποιείτε τους ακόλουθους τύπους:
 - $x_T = \cos(A) * v_0 * T$, $y_T = \sin(A) * v_0 * T - 1/2 * g * T^2$
- Ο χρήστης θα πρέπει να δώσει τη **γωνία** σαν **είσοδο**.





forever
imagine
program
share

