# Παιγνιώδης μάθηση και Παράλληλος προγραμματισμός σε Scratch

Σχεδιάζω, Υλοποιώ, Μοιράζομαι:
Καλές πρακτικές στη διδασκαλία της Πληροφορικής
Δευτέρα, 31 Ιανουαρίου 2022
Εισηγητής: Ψιλιάγκος Γιάννης ΠΕ86-03

# Γιατί **παράλληλος** προγραμματισμός;

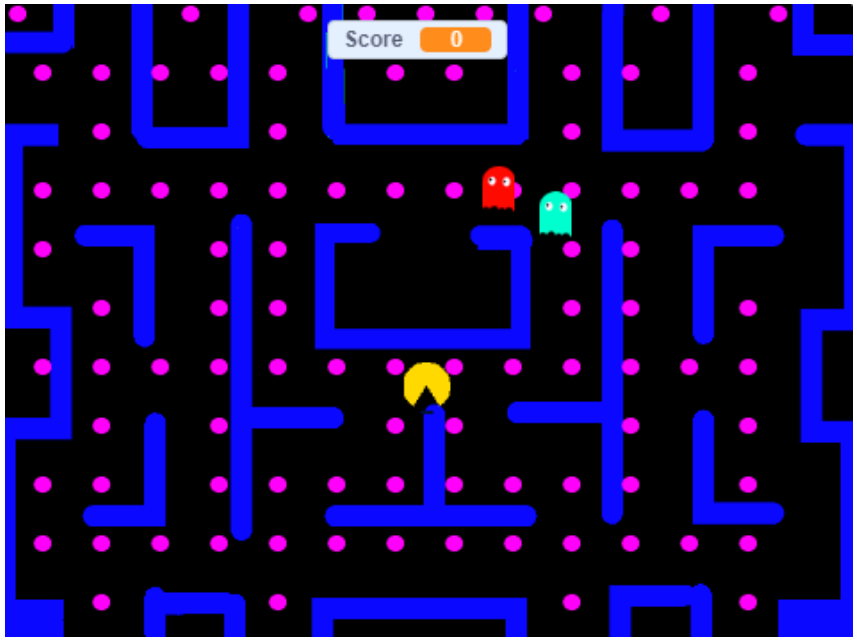
- Με βάση έναν αποδεκτό ορισμό: Είναι η χρήση πολλαπλών πόρων, στην προκειμένη περίπτωση, επεξεργαστών, για την επίλυση ενός προβλήματος. Αυτός ο τύπος προγραμματισμού αντιμετωπίζει ένα πρόβλημα, το αναλύει σε μια σειρά από μικρότερα βήματα, παραδίδει οδηγίες και οι επεξεργαστές (ή οι διεργασίες στον **ταυτοχρονισμό**) εκτελούν τις λύσεις ταυτόχρονα.
- Είναι ένα προγραμματιστικό παράδειγμα: μια μεγάλη διεύρυνση του σειριακού προγραμματισμού

# Κίνητρα-Ιστορικό-Δομή

- Πρόκειται για σενάριο με διάρκεια 6 δίωρα στο μάθημα πληροφορικής Α’ Λυκείου
- Εντάσσεται στα πλαίσια της διδασκαλίας του κεφαλαίου 7 – υλοποίηση εφαρμογών σε προγραμματιστικά περιβάλλοντα.
- Η ιδέα ήταν να δοθεί μια πρόταση για ολοκληρωμένη σχεδίαση παιχνιδιού Pac-man σε Scratch στα παιδιά.
- Έχει διδαχθεί (σε πρώιμη φάση) στο Λύκειο Ιερισσού τη χρονιά 2020-21 (περίοδος Νοεμβρίου-Ιανουαρίου) και τη χρονιά 2021-22 (περίοδος Νοεμβρίου-Φεβρουαρίου).
- Στόχος είναι να αποκτήσουν οι μαθητές μια πιο άμεση εμπειρία προγραμματισμού μέσα από κάτι που οι ίδιοι κατασκευάζουν
- Η διδασκαλία γίνεται σε ένα εισαγωγικό μάθημα και άλλα 5 δίωρα με αντίστοιχα φύλλα εργασιών που δίνονται με βοήθεια κοινόχρηστου έργου Scratch που έχει υλοποιημένα τα προηγούμενα βήματα. Παράλληλα δίνεται εργασία για το σπίτι.

# προγραμματισμός στην πράξη

- Υπάρχει το παιχνίδι- πρόβλημα (Pac-man και λαβύρινθοι) και είναι γνωστό στα παιδιά – το ίδιο το παιχνίδι γίνεται το εργαλείο μάθησης.
- Δε λέμε πολλά γι'αυτό: προγραμματίζουμε κατευθείαν.
- Δε μπορεί να είναι απόλυτα πιστό στην αντιγραφή του αρχικού Pac-man. Κάποια πράγματα όμως καλό είναι να μείνουν **κοντά στις γνώσεις των παιδιών για το παιχνίδι**. Κατά τα άλλα αυτό που θα προκύψει τελικά, θα είναι μια **διασκευή** του αρχικού, ώστε να εξαλειφθούν κάποιες δυσκολίες



# Το παιχνίδι είναι δημοφιλές και σε ακαδημαϊκά – εκπαιδευτικά περιβάλλοντα…

- Έχει χρησιμοποιηθεί σε διδασκαλία για:
  - Αντικειμενοστραφή προγραμματισμό
  - AI
  - (Παράλληλο) και Agent-Driven προγραμματισμό
- Αλγόριθμοι που μπορεί να συναντήσει κανείς μέσω του παιχνιδιού:
  - Maze Solving (πχ BFS, A*,Wall Following)
  - Concurrent Algorithms

# Οι παρακάτω διαφάνειες θα χωριστούν σε δύο τμήματα:

1. Τα βήματα για τη δημιουργία του παιχνιδιού
2. Σημεία σχετικά με τον παράλληλο προγραμματισμό

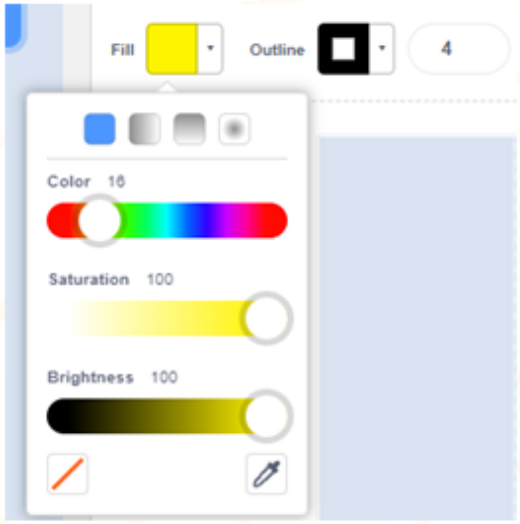
Γιατί παράλληλο προγραμματισμό;

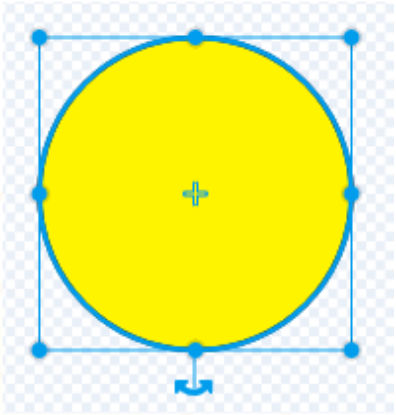
- Γιατί διδάσκεται λίγο (ή καθόλου) στο λύκειο
- Γιατί υπάρχει χρόνος και ωριμότητα σ'αυτή την ηλικία να τον κατανοήσουν
- Γιατί το Scratch προσφέρεται σε αυτή την ηλικία και διαθέτει τις δομές ώστε να τον κατανοήσουν γρήγορα τα παιδιά (αν και έχει κάποιες 'ατέλειες' στην υλοποίηση των δομών

# Βήματα για τη δημιουργία του Pac-man (1ο σχεδίαση του Pac-man)

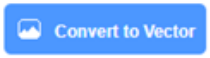
## Σχεδίαση χαρακτήρα

**Βήμα 1:** Σχεδιάζουμε ένα κύκλο (σε Vector περιβάλλον) με κέντρο στο κεντρικό σημείο του καμβά (το σημείο φαίνεται από το σταυρό στο κέντρο ). Βάλτε ένα φωτεινό κίτρινο ως γέμισμα και περίγραμμα μεγέθους 4 ή παραπάνω.



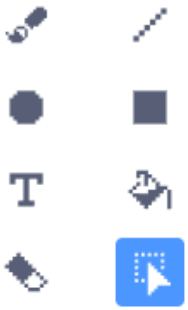



# Παράλληλα γίνεται μια μικρή αντιπαραβολή: γραφικά Bitmap vs Vector

ένα κουμπί 

το οποίο μας επαναφέρει τα εργαλεία στην προηγούμενη κατάσταση. Γενικά όταν θα δουλεύουμε σε μία από τις δύο καταστάσεις (με δυνατότητα εναλλαγής συνεχώς). Όταν δουλεύουμε σε Vector κάθε (*ζωγραφικό*) αντικείμενο που φτιάχνουμε εναποτίθεται πάνω από τα προηγούμενα, και μπορούμε να το επιλέξουμε με το εργαλείο επιλογής αντικειμένων (  )

Όταν δουλεύουμε σε Bitmap οι εικόνες μας είναι ψηφιδωτά από pixel οπότε ό,τι εργαλεία υπάρχουν εκεί βοηθάνε στην εργασία με πολλά pixel.Εδώ δε μπορούμε να επιλέξουμε ολόκληρα 'αντικείμενα' (κύκλους, γραμμές, γράμματα, ή ομαδοποιήσεις αυτών) Δείτε λίγο τα εργαλεία σε αντιπαραβολή και θα γίνει πιο κατανοητό:

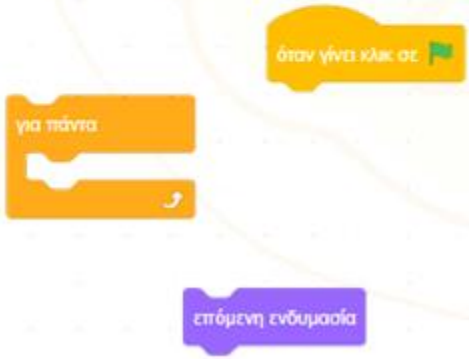
| Vector | Bitmap |
| --- | --- |
|  |  |

# Στο φύλλο εργασίας του βήματος 1 γίνεται μια σύντομη εισαγωγή στο Animation του Sprite

Δραστηριότητα 1

Έστω ότι μας δίνονται οι παρακάτω εντολές. Δείτε τον Pacman σαν μια 'κίτρινη ζωγραφιά' που αναβοσβήνει που τη μια στιγμή φαίνεται σαν κύκλος και την άλλη σαν κομμάτι πίτσας (του ...Obelix). Το ανοιγοκλείσιμο του στόματος γίνεται με μία από τις εντολές που φαίνεται παρακάτω. Ποια είναι αυτή η εντολή;

Σημειώστε εδώ:_____



# Βήματα για τη δημιουργία του Pac-man (2ο σχεδίαση του υποβάθρου)

## Φύλλο εργασίας 2

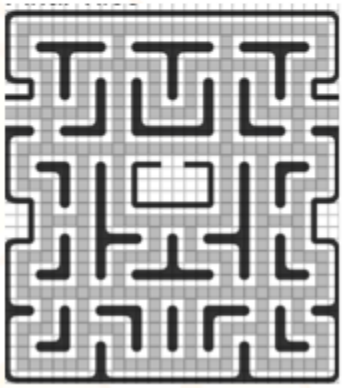
*Όπου φτιάχνουμε το υπόβαθρο με τους τοίχους και κάνουμε τον Pacman να μην περνάει από τους τοίχους*

Βήμα 0: Κάνουμε signinμε τους κωδικούς μας στο https://scratch.mit.edu/

Βήμα 1: πληκτρολογώντας 'Pacman maze generation' μεταβείτε στη διεύθυνση https://shaunlebron.github.io/pacman-mazegen/ και πιέστε στο πλήκτρο

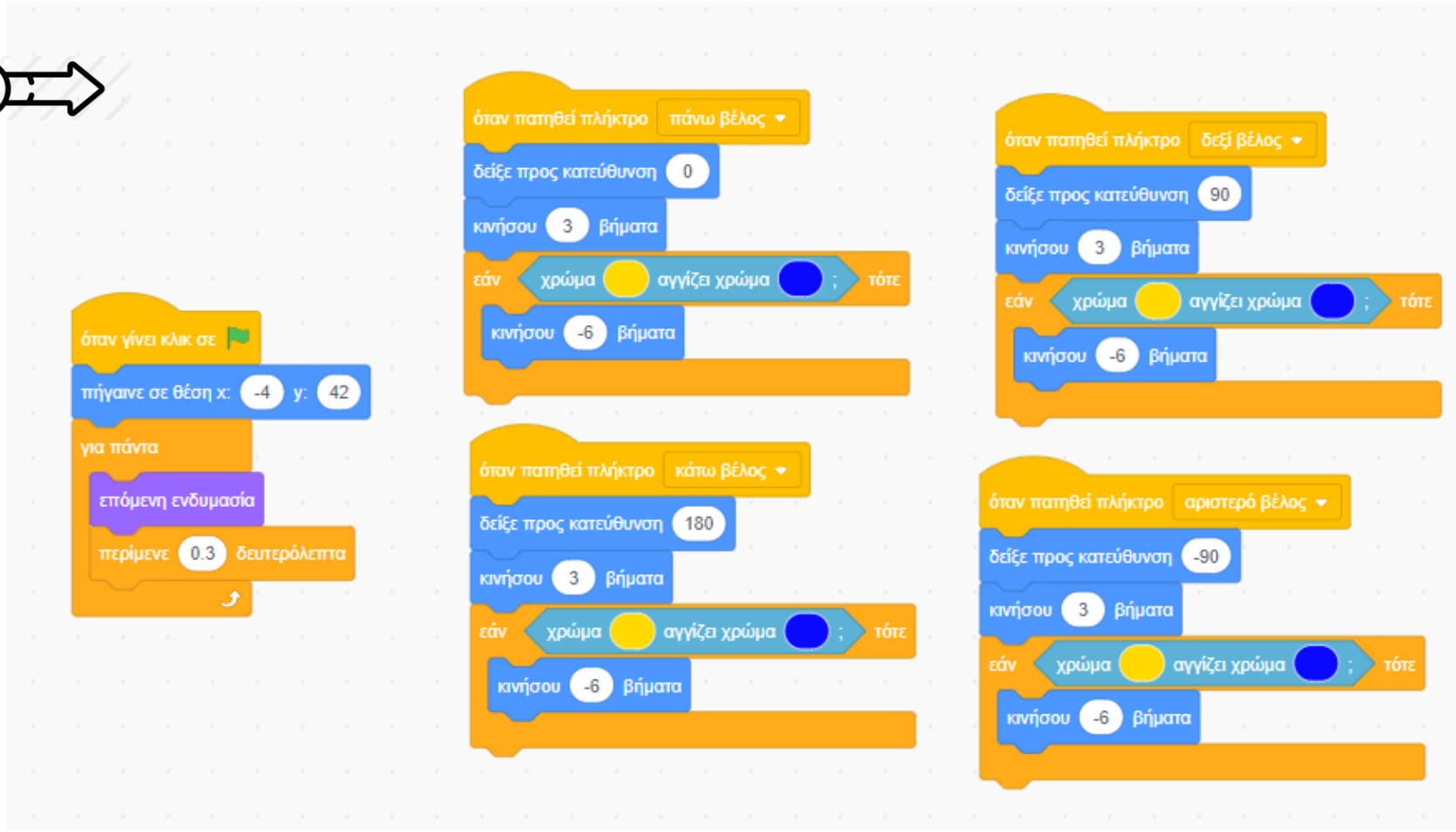
Click to generate new example

Βήμα 2: χρησιμοποιώντας το εργαλείο lightshot στον Browser, ή το sniping tool των Windows αποθηκεύστε μια εικόνα όπως αυτή στα δεξιά.



Βήμα 3:Κάνουμε μεταφόρτωση (upload ) από τον υπολογιστή το screenshotπου μόλις πήραμε, μέσα στο scratch. Όντας σε Vectormode αυτή η εικόνα πρέπει να 'τεντωθεί' ισόμορφα. Πως μπορεί να γίνει αυτό; (ποιο από τα χειριστήρια της εικόνας πρέπει να χρησιμοποιήσετε;

# Κίνηση του Pac-man ώστε να μη χτυπά σε τοίχους (1η εκδοχή όπου ο κώδικας στα πλήκτρα βέλους είναι επιβαρυμένος)

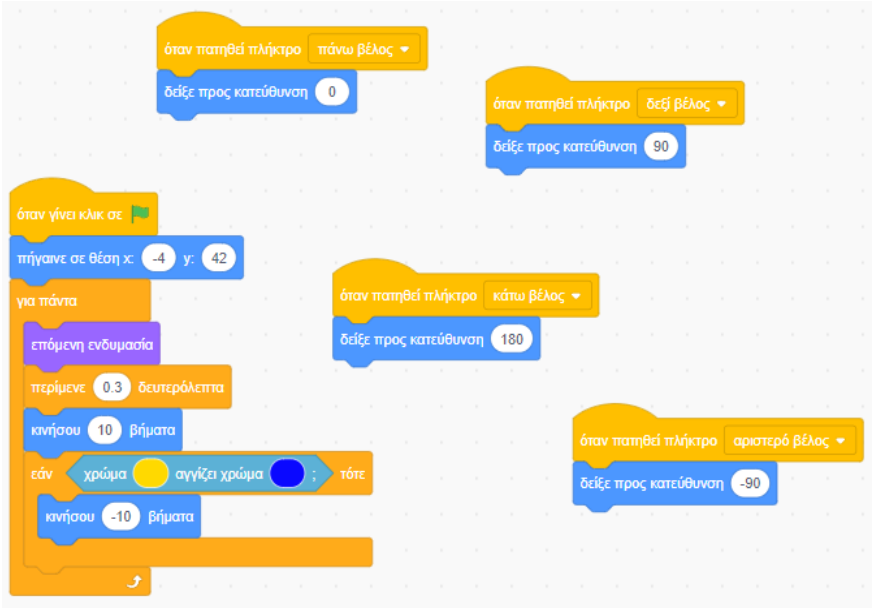


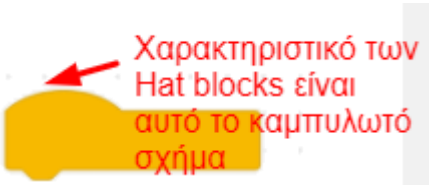
# Μια επισκόπηση των προβλημάτω δίνει το παρακάτω βίντεο

Χαρακτηριστικό πρόβλημα στην ταυτόχρονη εκτέλεση κάποιων εντολών είναι τα Racing Conditions: Εδώ ένα χαρακτηριστικό παράδειγμα όταν αυξάνεται το βήμα μετακίνησης με τη βοήθεια επιτάχυνσης


# Κίνηση του Pac-man ώστε να μη χτυπά σε τοίχους (2η εκδοχή)

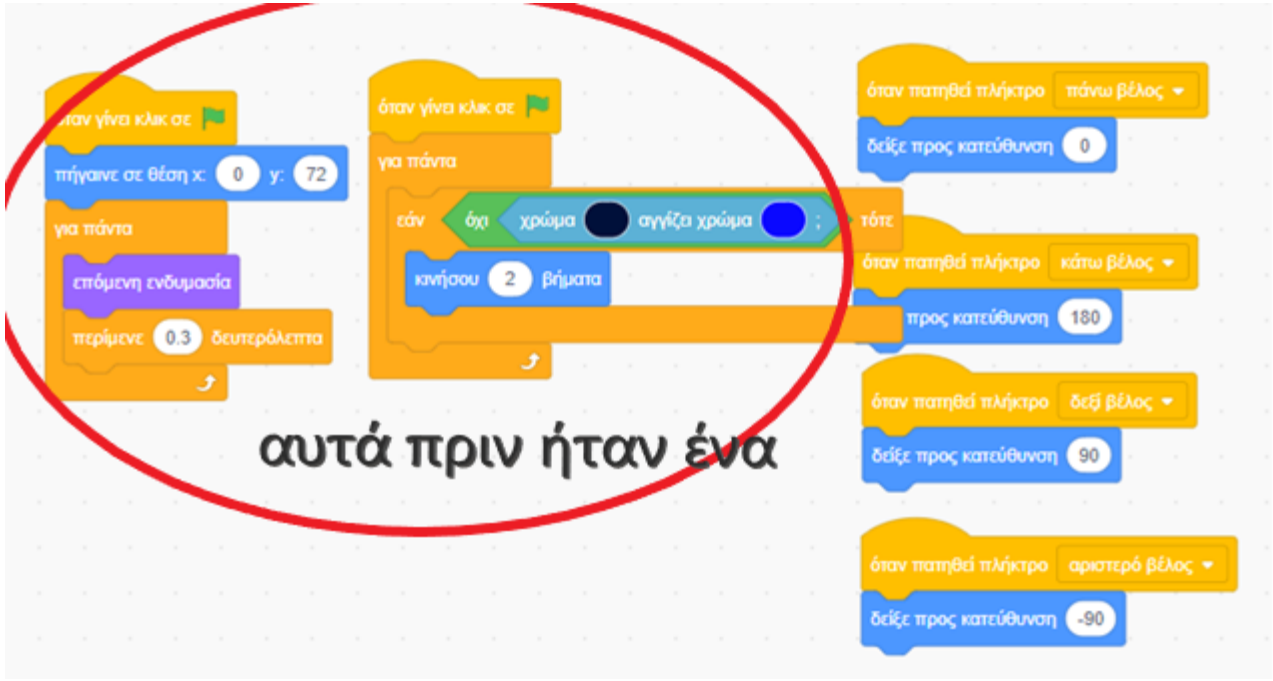
- Η παρακάτω εκδοχή βελτιώνει πολύ τα βέλη ωστόσο έχει ένα μειονέκτημα: πρέπει να περιμένει ο Pac-man να ανοιγοκλείσει το στόμα, και μετά να κάνει **ένα** βήμα.
- Αυτό δημιουργεί μια σπασμωδική κίνηση. Η διόρθωση για τη σπασμωδική κίνηση **είναι να εκμεταλλευτούμε την παράλληλη εκτέλεση που** συμβαίνει εντός του sprite όποτε χρησιμοποιούμε κάποιο hat block επιπλέον **στο Scratch** και να κάνουμε δύο ξεχωριστά προγράμματα για τον Pac-man (ή και 3 αν απαιτηθεί)
  - *(Σημείωση Τα Hat blocks σε κάθε sprite εκτελούνται παράλληλα γιατί εκ των έσω τρέχουν σε διαφορετικά threads)*



# Τελική εκδοχή του προγράμματος κίνησης του Sprite χωρίς τα προηγούμενα προβλήματα

Μια σημαντική βελτίωση του κώδικα στον παράλληλο προγραμματισμό είναι να εντοπίζονται σημεία όπου ο κώδικας ‘βελτιώνεται’ όταν σπάει σε δύο κομμάτια

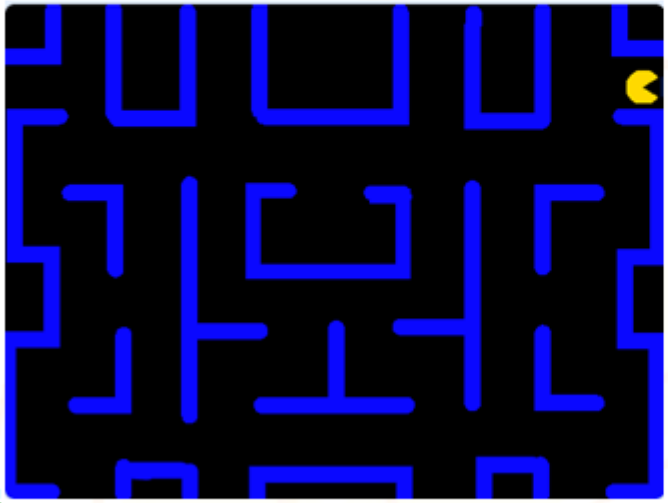


# Στο 3ο βήμα φροντίζουμε για τις εξόδους διαφυγής του Pac-man

## Φύλλο εργασίας 3

*Έξοδος και επανείσοδος στα σημεία διαφυγής*

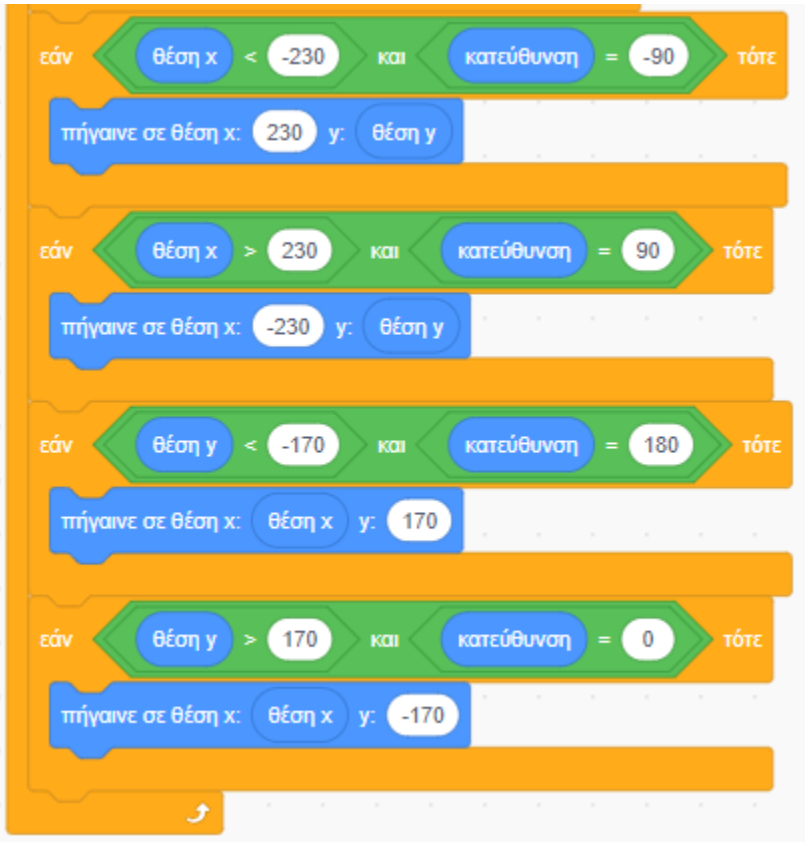
Υπάρχουν περιπτώσεις που ο Pacmanθέλει να ξεφύγει στην πίστα από τα φαντασματάκια. Δείτε παράδειγμα την παρακάτω εικόνα:



*Εικόνα9 Στο σημείο που είναι ο Pacmanυπάρχει άνοιγμα από δεξιά στον τοίχο. Μπορεί να βγει από δεξιά και να μπει από αριστερά του τοίχου στο ίδιο ύψος*

# Οι έξοδοι διαφυγής δίνουν μια ευκαιρία να εξεταστεί το σύστημα συντεταγμένων του παραθύρου του Scratch

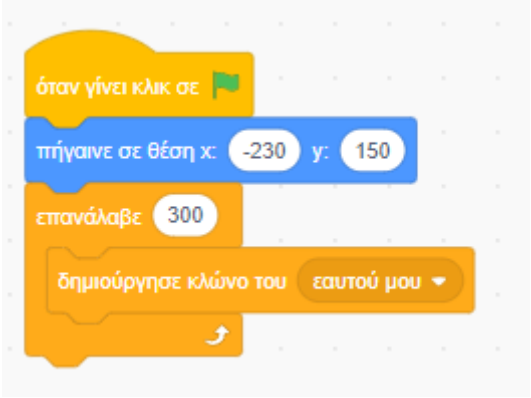
- Στο φύλλο εργασίας 3 ζητείται να τοποθετηθεί ο κώδικας στο κατάλληλο τμήμα του κώδικα του Pac-man



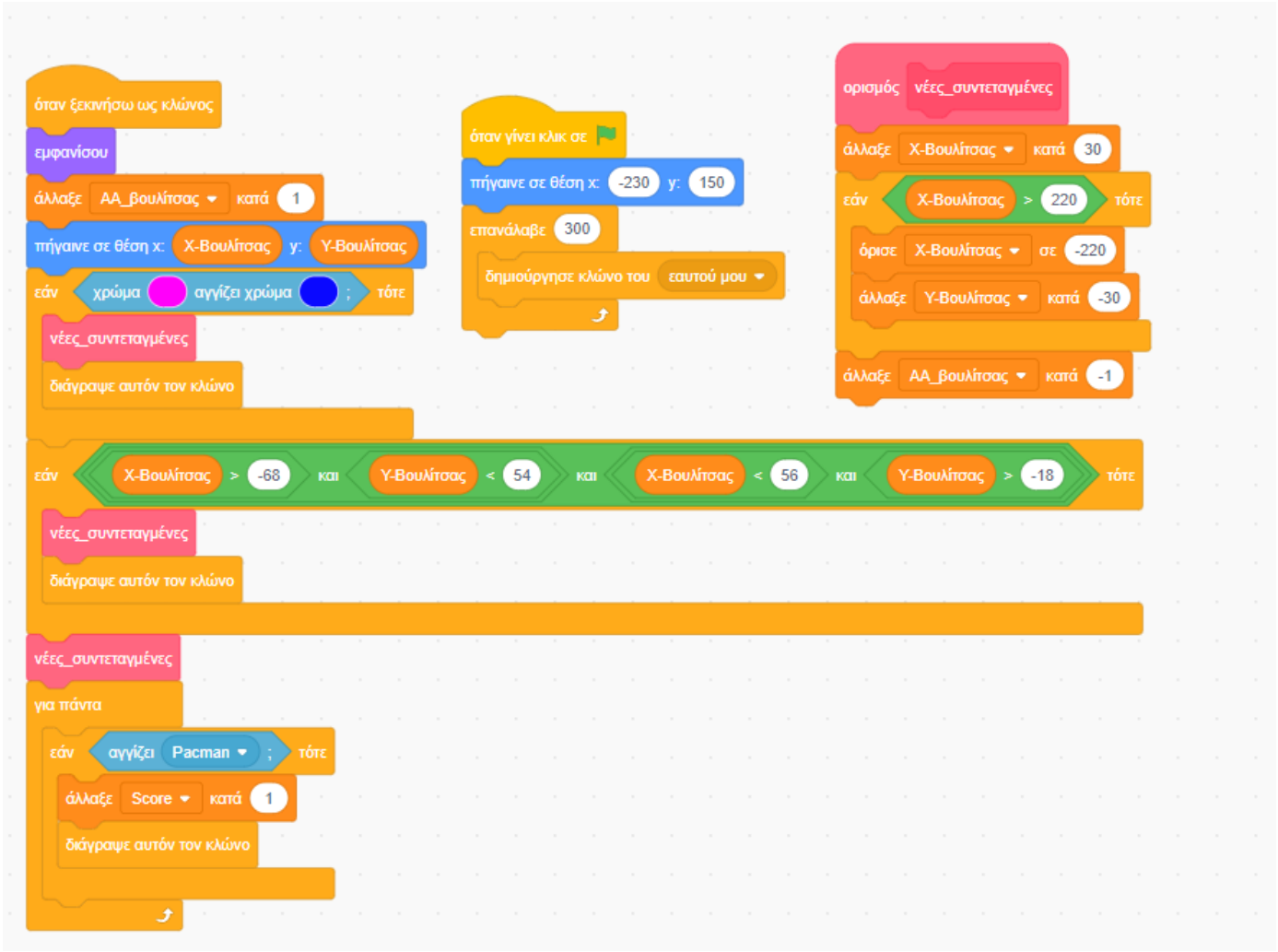
# Βήμα 4 Δημιουργία των cookies

(ή και peperoni)


- Ο κώδικας δημιουργίας τους χρησιμοποιεί το μηχανισμό κλωνοποίησης των αντικειμένων που διαθέτει το Pac-man
- Ο μηχανισμός αυτός προσφέρεται για εξέταση ζητημάτων παράλληλης πρόσβασης σε μεταβλητές (καθώς ο ΑΑ κάθε κουκίδας είναι παγκόσμια μεταβλητή, ενώ οι συντεταγμένες τους είναι τοπικές



# Ο κώδικας της κλωνοποίησης γεμίζει με αυτόματο τρόπο το χώρο της πίστας

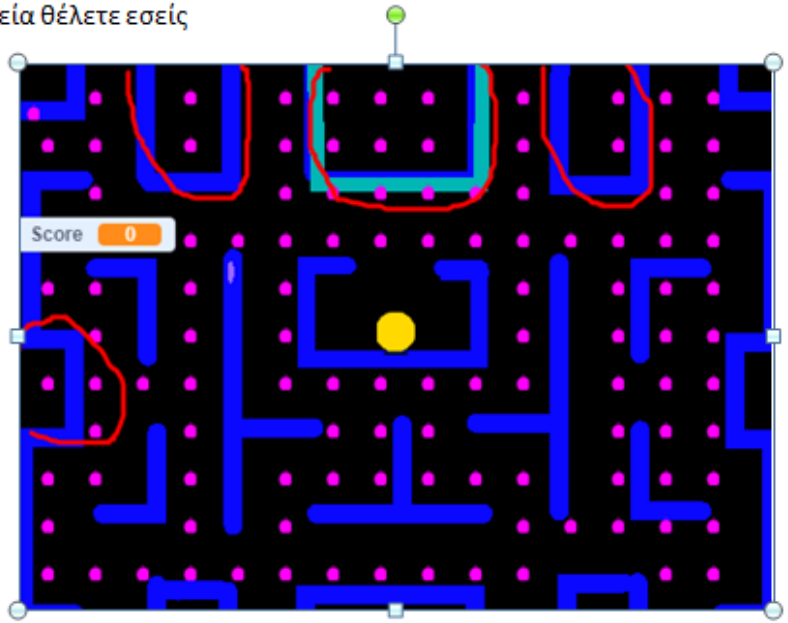


# Εργασία βήματος 4


## Φύλλο εργασίας 4ου βήματος

https://scratch.mit.edu/projects/625515966/editor

1. Παρατηρείστε ότι ο κώδικας που γεμίζει τις τελείες στην πίστα του σχήματος, δε γεμίζει το πρώτο 'κούφιο' τετραγωνάκι ούτε το κουτί από όπου βγαίνουν τα φαντασματάκια, ενώ γεμίζει τα υπόλοιπα. Διορθώστε ώστε να μη βγαίνουν τελείες –peperoni και στα άλλα κουτάκια που είναι σημειωμένα (ή και σε όποια άλλα σημεία θέλετε εσείς



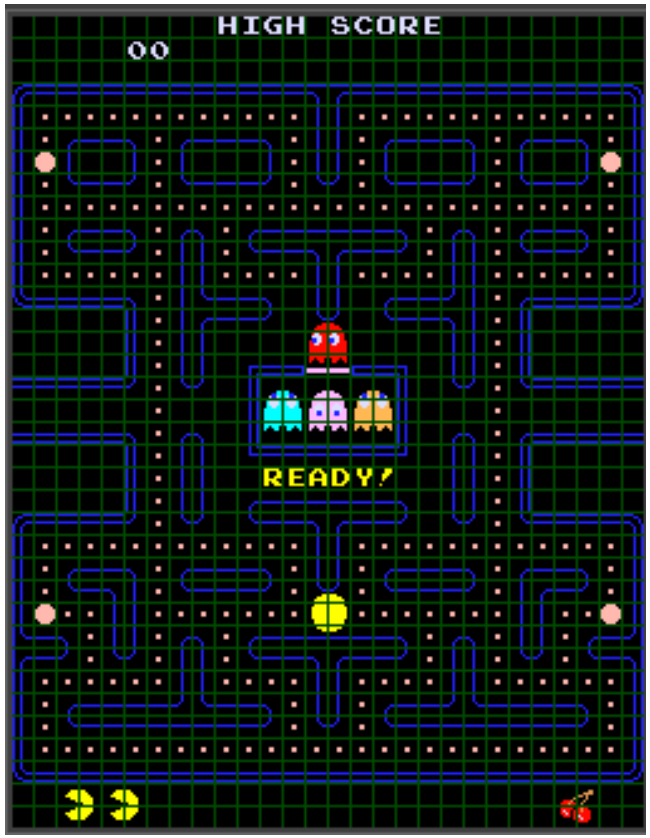
# 5ο βήμα :Προσοχή!:...φαντάσματα...

- Το 1999, ο Μπίλι Μίτσελ από το Χόλιγουντ της Φλόριντα έγινε ο πρώτος άνθρωπος που έλαβε τέλεια βαθμολογία 3.333.360 στο Pac-Man, τρώγοντας κάθε δυνατή κουκκίδα, energizer, φάντασμα και μπόνους σε κάθε επίπεδο χωρίς να χάσει ούτε μια μόνη ζωή στη διαδικασία.
- Ίσως το πιο εκπληκτικό είναι το γεγονός ότι μπορούσε να παίξει **χωρίς απομνημονευμένες** ρουτίνες ευρέως γνωστές ως «μοτίβα». Αντίθετα, βασιζόταν στην εξοικείωσή του με το πώς το **καθένα φάντασμα** συμπεριφέρεται καθώς κινείται μέσα στο λαβύρινθο, χρησιμοποιώντας αυτή τη γνώση για να κρατήσει τον Pac-Man ένα βήμα μπροστά από τους εχθρούς του ανά πάσα στιγμή

# Πως τα φαντάσματα βρίσκουν τον Pac-man

- Μεγάλο μέρος του σχεδιασμού και της μηχανικής του Pac-Man περιστρέφεται γύρω από την ιδέα να χωριστεί η πίστα σε πλακίδια. (διάστασης 40X40 pixels περίπου)
- Στο αρχικό παιχνίδι, ένα φάντασμα θεωρείται ότι έπιασε τον Pac-man όταν καταλαμβάνει το ίδιο πλακίδιο με αυτόν. Επιπλέον, κάθε πέλλετ στον λαβύρινθο βρίσκεται στο κέντρο του δικού του πλακιδίου.

  *Θα πρέπει να σημειωθεί ότι δεδομένου ότι τα sprites για το Pac-Man και τα φαντάσματα είναι μεγαλύτερα από ένα πλακίδιο σε μέγεθος, δεν περιέχονται ποτέ πλήρως σε ένα μόνο πλακίδιο. Εξαιτίας αυτού, για τους σκοπούς του παιχνιδιού, ο χαρακτήρας θεωρείται ότι καταλαμβάνει όποιο πλακίδιο περιέχει το κεντρικό του σημείο. Αυτή είναι σημαντική γνώση όταν αποφεύγετε τα φαντάσματα, καθώς ο Pac-Man θα συλληφθεί μόνο εάν ένα φάντασμα καταφέρει να μετακινήσει το κεντρικό του σημείο στο ίδιο πλακίδιο με αυτό του Pac-Man*.
- Το κλειδί για την κατανόηση της συμπεριφοράς φαντασμάτων είναι η έννοια του πλακιδίου στόχου. Τις περισσότερες φορές, κάθε φάντασμα έχει ένα συγκεκριμένο πλακίδιο που προσπαθεί να φτάσε. Όλα τα φαντάσματα χρησιμοποιούν πανομοιότυπες μεθόδους για να ταξιδέψουν προς τους στόχους τους, αλλά οι διαφορετικές προσωπικότητες φαντάσματα προκύπτουν λόγω του ατομικού τρόπου με τον οποίο κάθε φάντασμα επιλέγει το πλακίδιο-στόχο του.

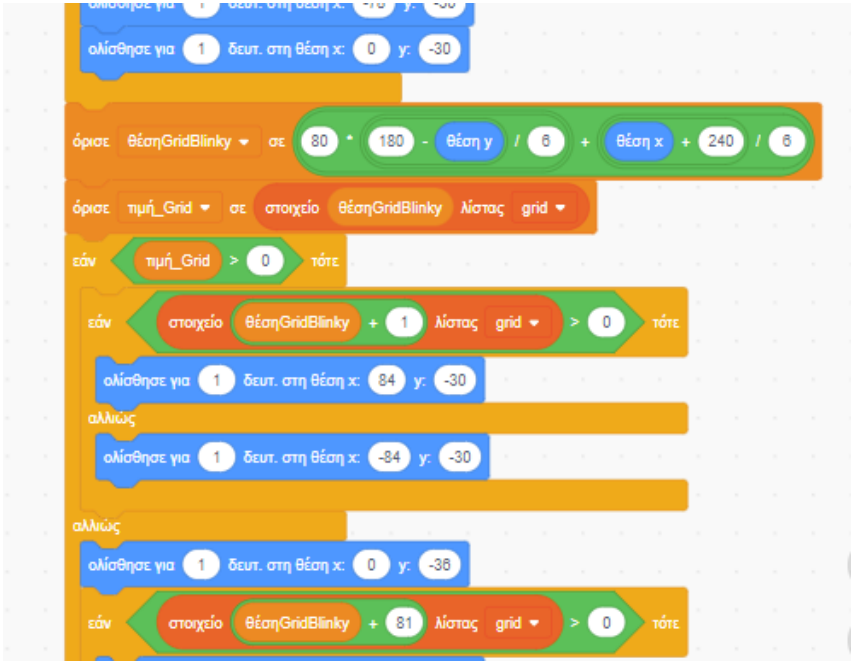


# Λειτουργίες κίνησης φαντασμάτων

- Τα  φαντάσματα βρίσκονται πάντα σε μία από τις τρεις πιθανές λειτουργίες: Chase, Scatter ή Frightened.
- Η "κανονική" λειτουργία με τα φαντάσματα που επιδιώκουν το Pac-Man είναι η Chase, και αυτή είναι αυτή στην οποία περνούν τον περισσότερο χρόνο τους. Ενώ βρίσκονται στη λειτουργία Chase, όλα τα φαντάσματα χρησιμοποιούν τη θέση του Pac-Man ως παράγοντα για την επιλογή του στόχου τους πλακάκι, αν και είναι πιο σημαντικό για ορισμένα φαντάσματα από άλλα.
- Στη λειτουργία Scatter, κάθε φάντασμα έχει ένα σταθερό πλακίδιο στόχο, καθένα από τα οποία βρίσκεται ακριβώς έξω από μια διαφορετική γωνία του λαβύρινθου. Αυτό κάνει τα τέσσερα φαντάσματα να διασκορπίζονται στις γωνίες όποτε βρίσκονται σε αυτήν τη λειτουργία.
- Η λειτουργία Frightened είναι μοναδική επειδή τα φαντάσματα δεν έχουν συγκεκριμένο πλακίδιο στόχο ενώ βρίσκονται σε αυτήν τη λειτουργία. Αντίθετα, αποφασίζουν ψευδοτυχαία ποιες στροφές θα κάνουν σε κάθε διασταύρωση. Ένα φάντασμα στη λειτουργία Frightened γίνεται επίσης σκούρο μπλε, κινείται πολύ πιο αργά και μπορεί να το φάει ο Pac-Man. Ωστόσο, η διάρκεια της λειτουργίας Frightened μειώνεται καθώς ο παίκτης προχωρά στα επίπεδα και εξαλείφεται εντελώς από το επίπεδο 19 και μετά.

# Η συγκεκριμένη υλοποίηση χρησιμοποιεί την παραπάνω τεχνική για τη Blinky (το γρήγορο φάντασμα)

- Γίνεται χρήση λίστας με την ονομασία Grid για την δημιουργία ενός πλέγματος που καταχωρεί κάθε πέρασμα του Pac-man από ένα 6x6 κουτάκι.
- Τα κουτάκια 6x6 που καλύπτουν όλο το χώρο κίνησης του Pac-man είναι 2840!



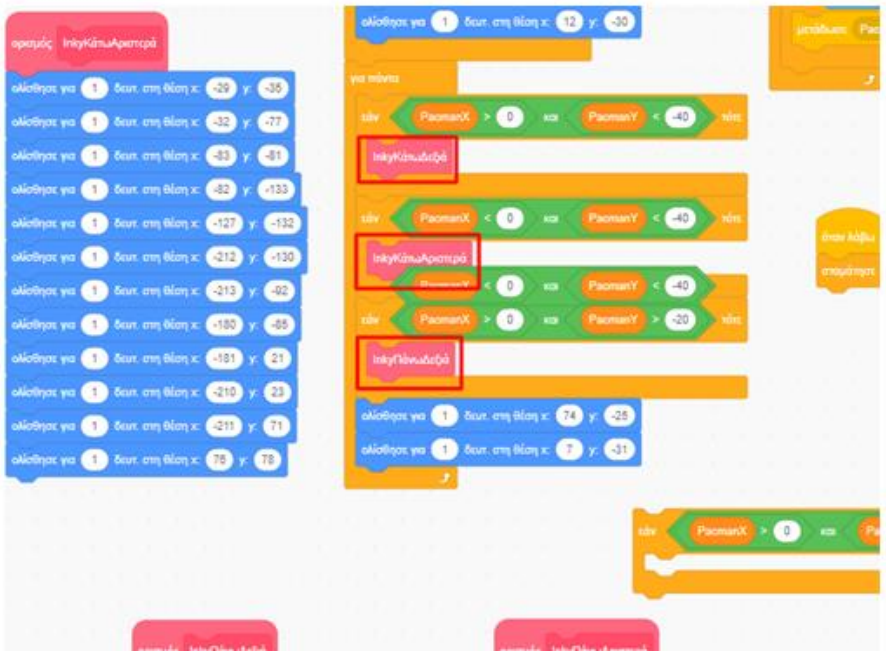
# Ενώ η Inky υλοποιείται με προ-αποθηκευμένες διαδρομές


## Φύλλο (1ο) εργασίας βήματος 5: Inky

*Όπου λέμε στην Inky το μπλε φάντασμα- μικρή σουπιά να 'στρέφει' την προσοχή της προς τα εκεί που είναι ο Pacman(σημ. το 4ο βήμα δείτε το πληροφοριακά μόνο)*

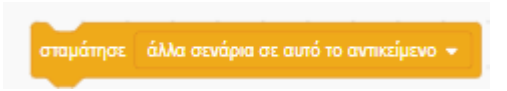
Για το φύλλο αυτό χρησιμοποιείστε την εξής υπερσύνδεση:

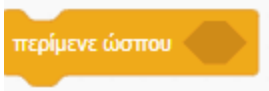
https://scratch.mit.edu/projects/636205523/editor/



# Παράλληλος προγραμματισμός

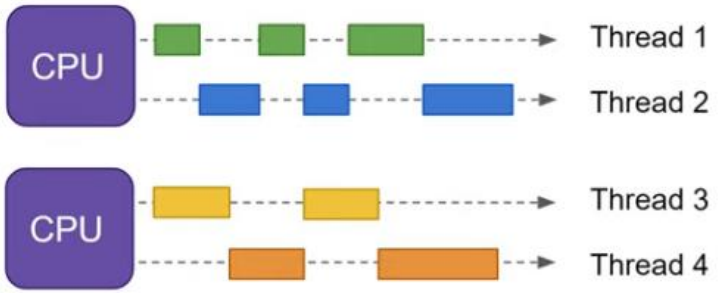
- Γενικά, θεωρούμε ότι σε ένα πρόγραμμα με παράλληλη εκτέλεση «συμβαίνουν πράγματα κατά την ίδια στιγμή», και ταυτόχρονα υπάρχουν συμβάντα (events) που «επιτρέπουν στα πράγματα να ξέρουν ότι κάτι συμβαίνει γύρω τους»
- Στο Scratch η ταυτόχρονη εκτέλεση συμβαίνει σε επίπεδο sprite και σε επίπεδο σεναρίων εντός του κάθε Sprite



# Παραλληλία και ταυτόχρονη εκτέλεση (ταυτοχρονισμός-concurrency) στο Scratch

- Στο υψηλό αυτό επίπεδο δε χρειάζεται να διακρίνουμε ανάμεσα σε παραλληλία και ταυτοχρονισμό. Η γλώσσα Scratch διαθέτει η ικανότητά του να εκτελεί πολλαπλά μπλοκ κώδικα ταυτόχρονα.
- Ο υπολογιστής δεν χρειάζεται να περιμένει να ολοκληρωθεί η εκτέλεση ενός μπλοκ κώδικα πριν εκτελέσει το επόμενο μπλοκ. Αυτό είναι γνωστό ως ταυτοχρονισμός.
- Όταν χρησιμοποιεί κάποιος το Scratch, δεν χρειάζεται να ανησυχεί για τη σειρά εκτέλεσης των μπλοκ κώδικα, μπορείτε απλώς να υποθέσετε ότι θα εκτελούνται όλα ταυτόχρονα.

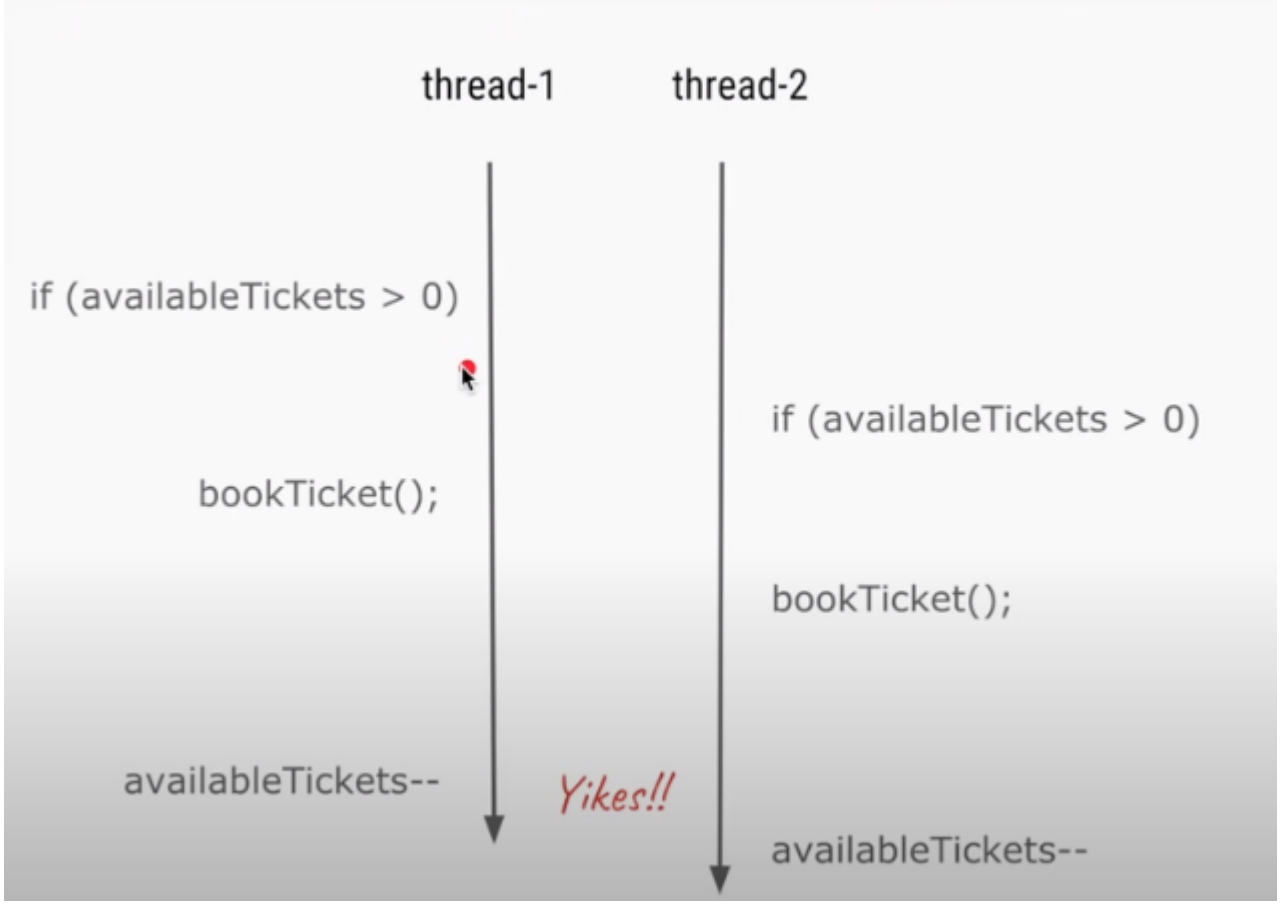


# Πρακτικά προβλήματα κατά τον προγραμματισμό (1) : κίνηση μέσα από τοίχους

- Όταν η ταχύτητα του Pac-man είναι πάνω από κάποιο όριο, αυτός αρχίζει να ‘περνάει’ μέσα από τοίχους
- Αυτό οφείλεται στον κώδικα που ελέγχει την κίνηση

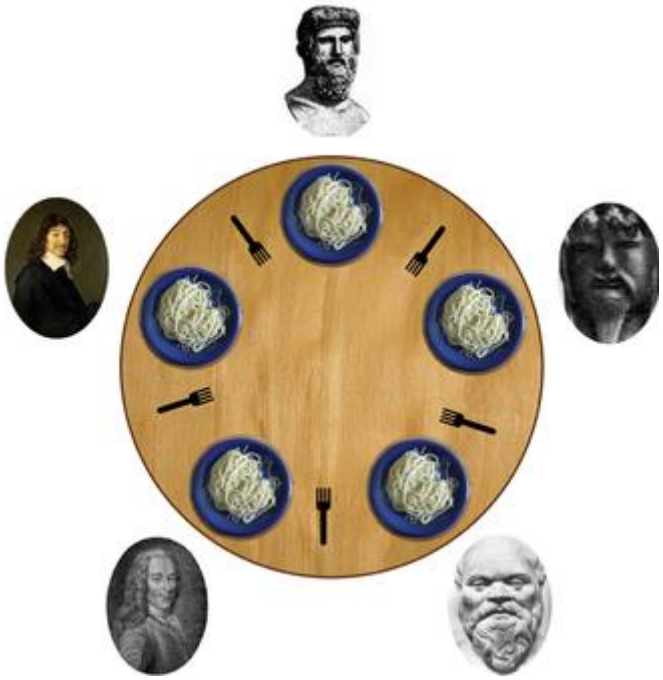
# Πρακτικά προβλήματα κατά τον προγραμματισμό (2): ταυτόχρονη χρήση διαμοιραζόμενων μεταβλητών

# Κατάσταση όπου 2 threads ανταγωνίζονται για το ίδιο resource



# Αυτό είναι γνωστό και ως dining philosophers problem

- 5 φιλόσοφοι τρώνε και σκέφτονται, σκέφτονται ή τρώνε (με όποια σειρά θέλει ο καθένας)
- Πρέπει όταν τρώνε να χρησιμοποιούν υποχρεωτικά 2 πηρούνια. Πως μπορεί να γίνει η διαδικασία χωρίς να πεινάσει κάποιος στο τέλος;



# Λύσεις στα προβλήματα παραλληλισμού μέσω Scratch

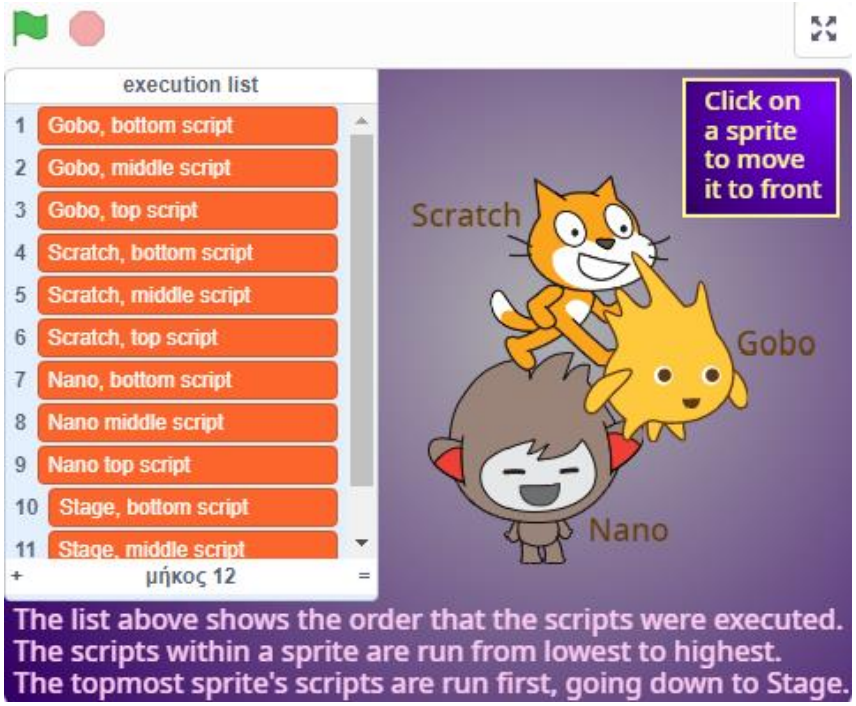
- Η χρήση μηνυμάτων
- Broadcasting
- Εντολές περίμενε..ώσπου
- Εντολές σταματήματος σεναρίων αντικειμένου

# Η σειρά εκτέλεσης των Hat Εντολών

- Οι Hat εντολές είναι αυτές που εκτελούνται ‘ταυτόχρονα’ όταν γίνει κλικ στη σημαία
- Η σειρά με την οποία ξεκινάν είναι η

1. Η σειρά με την οποία βρίσκονται τα Sprites το ένα πάνω από το άλλο (Layering)
2. Εντός κάθε Sprite η σειρά με την οποία εισήχθησαν στο Project

- Για περισσότερες λεπτομέρειες κάποιος μπορεί να τρέξει τον παρακάτω κώδικα




https://scratch.mit.edu/projects/18490761/

# Μερικές βασικές αρχές για τον παράλληλο προγραμματισμό σε Scratch

- Ο προγραμματιστής δεν πρέπει να ανησυχεί για τον ταυτοχρονισμό (concurrency) – το Scratch εκτελείται σε ένα μόνο νήμα και δεν κάνει καμία «διαπλοκή» της εκτέλεσης μπλοκ μεταξύ σεναρίων, εκτός εάν ένα σενάριο φτάσει σε ένα καλά καθορισμένο «σημείο απόδοσης». (τέλος ενός βρόχου, κάποιο είδος «αναμονής» ή τέλος του σεναρίου.
- Ωστόσο, εδώ είναι μερικά πράγματα που πρέπει να ληφθούν υπόψη: Όταν αποστέλεται ένα broadcast, όλα τα σενάρια δεκτών σε sprites/clones/Stage προστίθενται ως νέα threads.Ομοίως για άλλα είδη συμβάντων (π.χ. πάτημα πλήκτρων, όταν γίνεται κλικ, κ.λπ.)
- Πριν από κάθε ανανέωση οθόνης 1/30 του δευτερολέπτου, το Scratch θα κοιτάζει κάθε τρέχον προγραμματισμένο νήμα και θα τρέχει τα επόμενα μπλοκ στο σενάριό του μέχρι να εμφανιστεί, οπότε το Scratch θα προχωρήσει στο επόμενο νήμα. Μόλις περάσει από όλα τα νήματα, σηματοδοτεί στο ότι είναι έτοιμο για ανανέωση οθόνης. [Λάβετε υπόψη ότι σε λειτουργία turbo ή εάν κανένα από τα νήματα δεν έχει εκτελέσει μπλοκ που πιθανώς έκαναν αλλαγές στην οθόνη, το Scratch θα εκτελέσει όσα περάσματα μπορεί μέσα από τα νήματα εντός του χρόνου ανανέωσης 1/30 του δευτερολέπτου.]
- Ο «σωστός» τρόπος για να γίνουν πράγματα στο Scratch είναι να υπάρξει έλεγχος του προγραμματισμού της εκτέλεσης χρησιμοποιώντας Broadcast.

# Αποτίμηση του σεναρίου

- Οι μαθητές έχουν δείξει ενθουσιασμό ήδη από την πρώτη εφαρμογή (2021) της προσέγγισης αυτής σε μορφή project για το σπίτι.
- Η πιο αναλυτική προσέγγιση με τα ενδιάμεσα φύλλα εργασιών προσφέρει καλύτερη εστίαση σε συγκεκριμένες προγραμματιστικές δομές
- Χρειάζεται μεγαλύτερη επιμέρους βελτίωση ως προς την επεξήγηση κάποιων δύσκολων εννοιών

# Πηγές- ενδιαφέροντα projects

- Pac-man Dossier
- Τάσος Λαδιάς: Scratch3
- https://stabyourself.net/notPac-man/