Τοπολογίες παράλληλου προγραμματισμού στο Scratch και κατηγοριοποίησή τους με χρήση της ταξινομίας SOLO

Αναστάσιος Λαδιάς[1], Θεόδωρος Καρβουνίδης[2]

[1]Δρ.Εκπαιδευτικός Πληροφορικής
tkarv@unipi.gr
[2]Δρ. Πληροφορικής
ladiastas@gmail.com

Περίληψη

Στο προγραμματιστικό περιβάλλον του Scratch μπορούν να συντρέχουν ταυτόχρονα προγραμματιστικά σενάρια. Η αλληλεξάρτηση μεταξύ των συντρεχόντων σεναρίων, η μεταξύ των επικοινωνία για συγχρονισμό, οι αιτίες που τα ενεργοποιούν / απενεργοποιούν και οι ροές νημάτων που σχηματίζονται, δημιουργούν τοπολογίες κώδικα παράλληλου προγραμματισμού. Στην παρούσα εργασία επιχειρείται αυτές οι τοπολογίες να ομαδοποιηθούν με βάση ενδογενή κριτήρια και να αντιστοιχηθούν στα πέντε επίπεδα της ταξινομίας SOLO. Η υιοθέτηση αυτής της κατάταξης μπορεί να οδηγήσει αφενός στη δημιουργία αξόνων για την ανάπτυξη ενός προγράμματος σπουδών βασισμένου στον προγραμματισμό Η/Υ στην υποχρεωτική εκπαίδευση και αφετέρου σε ένα σύστημα αξιολόγησης με μετρήσιμα κριτήρια του κώδικα των μαθητών όσον αφορά το βαθμό παραλληλίας των σεναρίων. Η παρούσα εργασία εντάσσεται στα πλαίσια ενός έργου αξιολόγησης του κώδικα αλλά και σε ένα ευρύτερο πλαίσιο αξιολόγησης έργων εκπαιδευτικής ρομποτικής.

Λέξεις κλειδιά: Παράλληλος προγραμματισμός, Scratch, ταξινομία SOLO

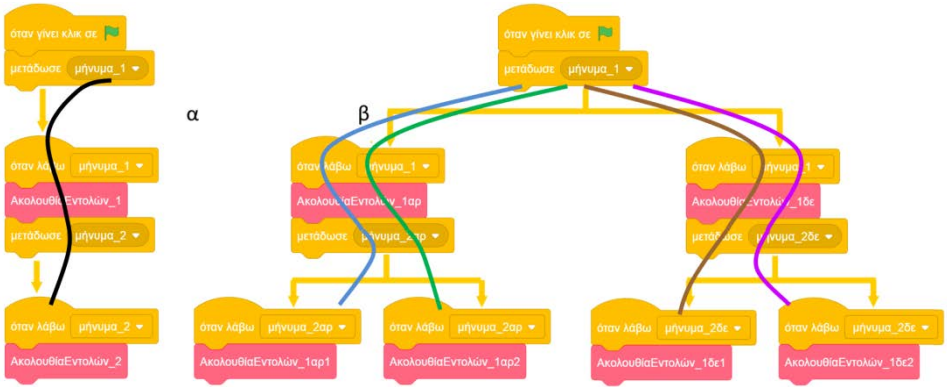
1. Εισαγωγή

Το Scratch είναι ένα εκπαιδευτικό περιβάλλον οπτικού προγραμματισμού με πλακίδια, στο οποίο ο αρχάριος προγραμματιστής μπορεί να διαχειριστεί μέσω κώδικα -με σχετική ευκολία- πολυμεσικά στοιχεία και να εμπλακεί σε αυθεντικά ψηφιακά έργα δημιουργώντας animation, προσομοιώσεις και παιχνίδια. Το προγραμματιστικό περιβάλλον του Scratch μπορεί να υποστηρίξει διάφορα προγραμματιστικά στυλ όπως δομημένο (τμηματικό και ιεραρχικό) προγραμματισμό, προγραμματισμό καθοδηγούμενο από συμβάντα, προγραμματισμό βασισμένο σε αντικείμενα, μια μορφή αντικειμενοστραφούς προγραμματισμού (με τη χρήση κλώνων) και παράλληλο προγραμματισμό.

Έχει παρατηρηθεί ότι οι κώδικες σε Scratch πολλών μαθητών, εμπεριέχουν τμήματα παράλληλου προγραμματισμού, αν και σπάνια αυτοί οι μαθητές έχουν διδαχθεί από

ISBN: 978-960-578-084-5

τους εκπαιδευτικούς τους στοιχεία παράλληλου προγραμματισμού. Αυτό οφείλεται στο γεγονός ότι «ο παράλληλος προγραμματισμός είναι πιο κοντά στο φυσικό τρόπο σκέψης των μαθητών, αφού στον πραγματικό κόσμο ανεξάρτητες οντότητες λειτουργούν ταυτόχρονα και επικοινωνούν μεταξύ τους» (Kahn, 1996). Συνεπώς το Scratch προσφέρεται για τη διδασκαλία στοιχείων παράλληλου προγραμματισμού σε αρχάριους, λόγω του συνδυασμού αφενός της ευκολίας εκμάθησης και χειρισμού του και αφετέρου του ότι ως μια γλώσσα προγραμματισμού έχει τη δυνατότητα να εκτελούνται ταυτόχρονα τμήματα προγραμμάτων (διεργασίες). Για αυτά τα τμήματα προγραμμάτων στο Scratch, στο εξής θα χρησιμοποιείται ο όρος σενάρια, ενώ ως πρόγραμμα θα νοείται το σύνολο των σεναρίων. Τα σενάρια ελέγχουν τη συμπεριφορά αντικειμένων (objects) που εμφανίζονται στη σκηνή του Scratch ως ρόλοι (sprites). Στο εσωτερικό της ακολουθίας των εντολών ενός σεναρίου είναι δυνατόν να εμπεριέχονται και άλλες εντολές (όπως π.χ. η «μετάδωσε μήνυμα…»), που δημιουργούν νήματα (threads) σεναρίων (Σχήμα 1).



Σχήμα 1: *Σενάρια με: (α) γραμμικό νήμα και (β) νήματα σε δενδροειδή δομή.*

Για πληρέστερη κατανόηση του κώδικα που απεικονίζεται στα σχήματα προτείνεται κάθε διαδικασία «ΑκολουθίαΕντολών…» να αντιστοιχίζεται σε ένα βρόχο «Επανάλαβε ώσπου…» ώστε να υποδηλώνεται μια απροσδιόριστη χρονική διάρκεια εκτέλεσης της διαδικασίας.

Η εργασία αυτή είναι συνέχεια άλλης εργασίας (Λαδιάς, Καρβουνίδης & Λαδιάς, 2020) που προτείνει την εισαγωγή στοιχείων παράλληλου προγραμματισμού Η/Υ στην υποχρεωτική εκπαίδευση. Αυτές οι εργασίες αποτελούν τμήμα μιας ευρύτερης έρευνας η οποία επιχειρεί να καθορίσει ένα πλαίσιο αξιολόγησης του κώδικα του προγραμματιστικού περιβάλλοντος Scratch (Karvounidis, Argyriou, Ladias & Douligeris, 2017). Το προαναφερθέν πλαίσιο αναπτύσσεται σε δύο άξονες που αφορούν την ανατομία και τη λειτουργικότητα του κώδικα. Στόχος της παρούσης εργασίας είναι η ένταξη στον άξονα της λειτουργικότητας, των διαφόρων τοπολογιών παράλληλου προγραμματισμού στα επίπεδα της ταξινομίας SOLO (Structure of the Observed Learning Outcome). Η ταξινομία SOLO προτείνει την αξιολόγηση της

ISBN: 978-960-578-084-5

γνώσης με βάση τη δομή του παρατηρούμενου μαθησιακού αποτελέσματος (Biggs & Collis, 1982). Εξειδικεύοντας στον προγραμματισμό Η/Υ, η ταξινομία SOLO έχει χρησιμοποιηθεί για την αξιολόγηση του κώδικα (Jimoyiannis, 2011). Σε περαιτέρω εξειδίκευση η ταξινομία SOLO σε συνδυασμό με την αναθεωρημένη ταξινομία του Bloom έχει χρησιμοποιηθεί στην αξιολόγηση γνώσεων προγραμματισμού στο Scratch αναλύοντας σε βάθος τις γνώσεις των μαθητών (Meerbaum-Salant, Armoni & Ben-Ari, 2013). Τέλος οι Μπέλλου & Μικρόπουλος (2008) προτείνουν το μοντέλο της Ιεραρχικής Αξιολόγησης γνώσεων Προγραμματισμού, που βασίζεται στην ταξινομία SOLO και προτείνει πέντε ιεραρχικά επίπεδα που ορίζονται με άξονες αφενός την ανάπτυξη αλγοριθμικής σκέψης για την επίλυση προβλήματος και αφετέρου τις δεξιότητες στη γλώσσα προγραμματισμού. Τα προτεινόμενα επίπεδα είναι: (α) Το προδομικό επίπεδο, στο οποίο γίνεται αναφορά ή χρήση μη συνδεδεμένων και ανοργάνωτων πληροφοριών που δεν έχουν νόημα. (β) Το μονοδομικό επίπεδο, όπου παρατηρείται μια περιορισμένη οπτική -κυρίως χρησιμοποιείται ή τονίζεται ένα στοιχείο ή μια πτυχή- ενώ παραλείπονται οι υπόλοιπες συνιστώσες και δεν πραγματοποιούνται σημαντικές συνδέσεις μεταξύ των μερών. (γ) Το πολυδομικό επίπεδο, στο οποίο υπάρχει μια προοπτική πολλαπλών σημείων -χρησιμοποιούνται ή αναγνωρίζονται διάφορα σχετικά στοιχεία ή πτυχές- αλλά δεν υπάρχουν σημαντικές συνδέσεις και δεν έχει διαμορφωθεί ακόμη μια ολοκληρωμένη εικόνα. (δ) Το σχεσιακό επίπεδο, στο οποίο υπάρχει μια ολιστική προοπτική όπου οι μετα-συνδέσεις μεταξύ των μερών γίνονται αντιληπτές και η σημασία των τμημάτων σε σχέση με το σύνολο αποδεικνύεται και εκτιμάται και (ε) Το επίπεδο της εκτεταμένης γενίκευσης, στο οποίο επιπλέον των χαρακτηριστικών του προηγούμενου συσχετιστικού επιπέδου, το περιεχόμενο αντιμετωπίζεται ως ένα στιγμιότυπο μιας γενικότερης περίπτωσης.

Σύμφωνα με το Δημακόπουλο (2017), «ο παράλληλος προγραμματισμός παραμένει ένα είδος τέχνης... καθώς η ποιότητα ενός παράλληλου προγράμματος εξαρτάται από την εμπειρία και την ικανότητα του προγραμματιστή, που θα λάβει υπόψη του και τις επιδόσεις του υπολογιστικού συστήματος που θα φιλοξενήσει το πρόγραμμα». Με δεδομένο ότι το Scratch "τρέχει" σε μη παράλληλους επεξεργαστές, θα επιδιωχθεί να κατηγοριοποιηθεί στα επίπεδα της ταξινομίας SOLO, ο (ψευδο)παράλληλος κώδικας στο προγραμματιστικό περιβάλλον του Scratch.

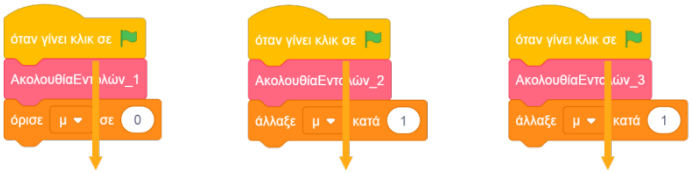
2. Τοπολογίες παράλληλου προγραμματισμού και SOLO

Οι Λαδιάς κ.ά., (2020) έχουν παρουσιάσει διάφορα στοιχεία τοπολογίας παράλληλων σεναρίων ενώ στην παρούσα εργασία οι τοπολογίες θα αξιολογηθούν και θα αντιστοιχηθούν στα προαναφερόμενα πέντε επίπεδα της ταξινομίας SOLO.

Προδομικό επίπεδο της SOLO και παράλληλα σενάρια στο Scratch. Εξετάζοντας κώδικες αρχαρίων προγραμματιστών διαπιστώνεται ότι γίνεται μεν χρήση παράλληλων σεναρίων όμως ο τρόπος που χρησιμοποιούνται υποδηλώνει ότι οι μαθητές έχουν περιορισμένη αντίληψη για τη διαχείρισή τους και έτσι αυτοί οι κώδικες

ISBN: 978-960-578-084-5

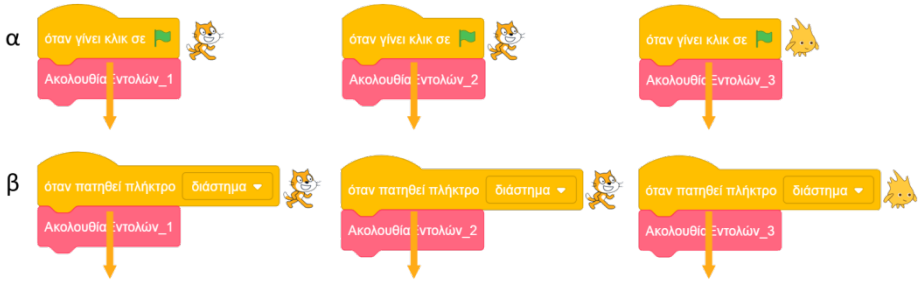
κατατάσσονται στο προδομικό επίπεδο. Παράδειγμα τα παράλληλα σενάρια του Σχήματος 2 που ξεκινούν ταυτόχρονα χωρίς να λαμβάνουν υπόψη ότι κάποια επιβάλλεται να προηγηθούν των άλλων. Να σημειωθεί εδώ ότι λόγω της μη-αιτιοκρατικής συμπεριφοράς που χαρακτηρίζει τον παράλληλο προγραμματισμό (Λαδιάς κ.ά., 2020) το πρόγραμμα δεν "τρέχει" πάντοτε χωρίς προβλήματα.



Σχήμα 2: *Παράλληλα σενάρια που δεν λαμβάνουν υπόψη τους ότι η εκτέλεση της εντολής "όρισε μ=0" ως αρχικοποίηση της τιμής του μετρητή πρέπει να προηγηθεί της εκτέλεσης των εντολών "άλλαξε το μ κατά 1" των υπολοίπων σεναρίων.*

Επίσης επειδή σύμφωνα με τον Bustard (1990) «ένα σειριακό πρόγραμμα στην πραγματικότητα είναι ένα παράλληλο πρόγραμμα στο οποίο έχει οριστεί μόνο μια δραστηριότητα» και υπό αυτήν την έννοια, όλα τα σειριακά προγράμματα (εξεταζόμενα υπό το πρίσμα του παράλληλου προγραμματισμού) μπορούν να ταξινομηθούν στο προδομικό επίπεδο της ταξινομίας SOLO.

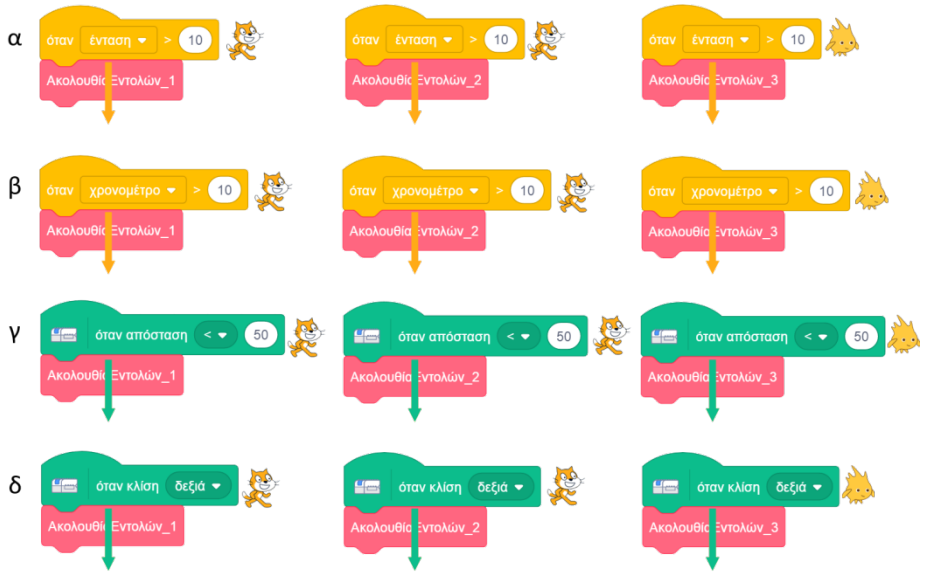
Μονοδομικό επίπεδο της SOLO και παράλληλα σενάρια στο Scratch. Στο μονοδομικό επίπεδο της SOLO, ο κώδικας χαρακτηρίζεται από παράλληλα σενάρια που λειτουργούν ανεξάρτητα το ένα από το άλλο ως νήματα (χωρίς την ανάγκη να επικοινωνούν / συγχρονίζονται μεταξύ τους), που ξεκινούν ταυτόχρονα, ενεργοποιούμενα από την ίδια αιτία.



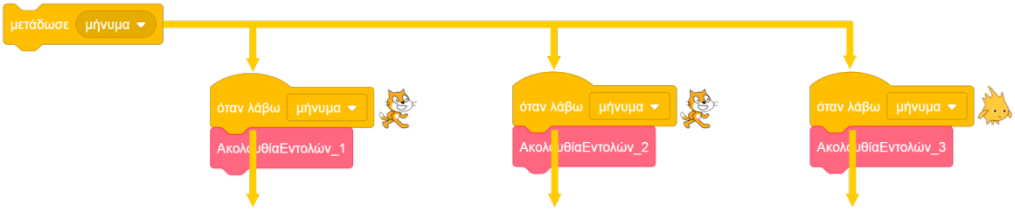
Σχήμα 3: *Δύο περιπτώσεις (α, β) παράλληλων σεναρίων που ξεκινούν ταυτόχρονα, που είναι νήματα ανεξάρτητα το ένα από το άλλο, ενεργοποιούμενα από την αλληλεπίδραση με το χρήστη και που μπορούν να κατανέμονται σε έναν ή σε διαφορετικούς ρόλους αντικειμένων.*

Αιτία μπορεί να είναι είτε εξωτερικά συμβάντα όπως η αλληλεπίδραση με το χρήστη (Σχήμα 3) ή συσκευές (Σχήμα 4) είτε εσωτερικά συμβάντα όπως η μετάδοση μηνυμάτων (Σχήμα 5) ή η δημιουργία κλώνων (Σχήμα 6).

ISBN: 978-960-578-084-5



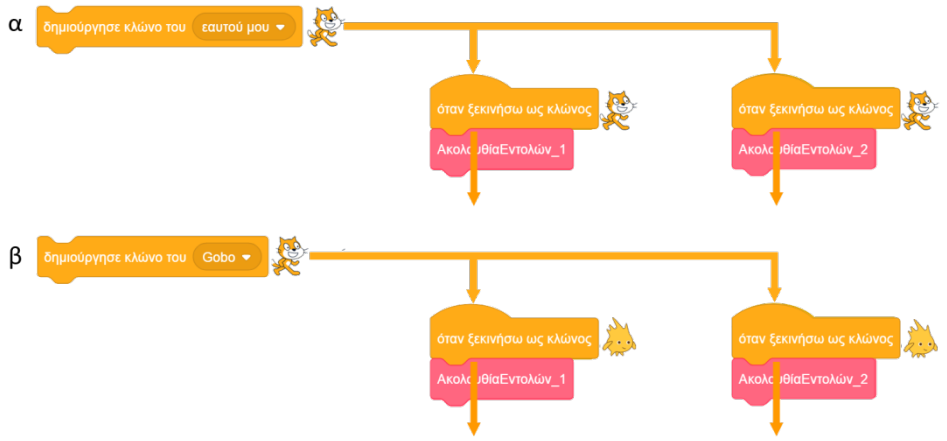
Σχήμα 4: *Τέσσερες περιπτώσεις παράλληλων σεναρίων που ξεκινούν ταυτόχρονα, που είναι ανεξάρτητα νήματα το ένα από το άλλο, ενεργοποιούμενα από συσκευές (α: μικρόφωνο, β: χρονόμετρο, γ: αισθητήρας απόστασης και δ: αισθητήρας κλίσης) και που μπορούν να κατανέμονται σε έναν ή σε διαφορετικούς ρόλους αντικειμένων.*



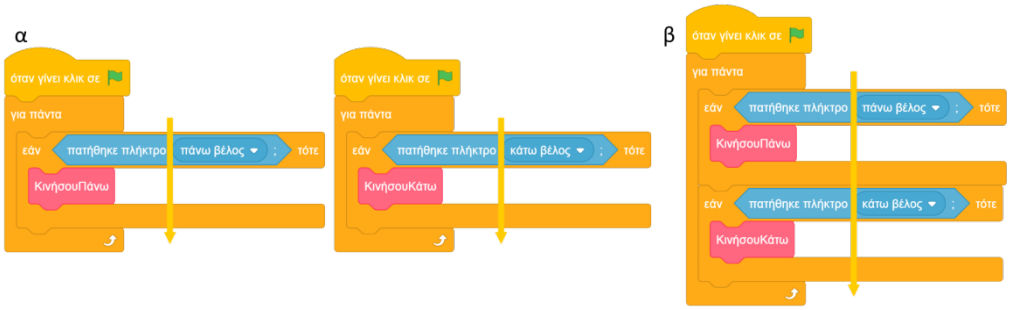
Σχήμα 5: *Τρία παράλληλα σενάρια που ξεκινούν ταυτόχρονα, που είναι νήματα ανεξάρτητα το ένα από το άλλο, ενεργοποιούμενα από τη λήψη ενός μηνύματος (εσωτερικό συμβάν) τα οποία μπορούν να κατανέμονται σε έναν ή σε διαφορετικούς ρόλους αντικειμένων.*

Κατά την αξιολόγηση θα πρέπει να συνεκτιμηθεί κατά πόσο ο μαθητής μπορεί να διακρίνει μια λύση με παράλληλο κώδικα από μια λύση με σειριακό κώδικα συγκρίνοντας τις ομοιότητες ή διαφορές και τα πλεονεκτήματα και μειονεκτήματα της κάθε λύσης. Παράδειγμα οι κώδικες του σχήματος 7 όπου στην πρώτη περίπτωση υπάρχουν δύο παράλληλα σενάρια (μονοδομικό επίπεδο SOLO) ενώ στη δεύτερη περίπτωση στο σειριακό σενάριο οι έλεγχοι γίνονται σειριακά (προδομικό επίπεδο SOLO).

ISBN: 978-960-578-084-5



Σχήμα 6: *Δύο περιπτώσεις παράλληλων σεναρίων που ξεκινούν ταυτόχρονα, που είναι νήματα ανεξάρτητα το ένα από το άλλο, ενεργοποιούμενα ως δημιουργούμενος κλώνος (εσωτερικό συμβάν), τα οποία μπορούν να κατανέμονται (α) σε έναν ή (β) σε διαφορετικούς ρόλους αντικειμένων.*

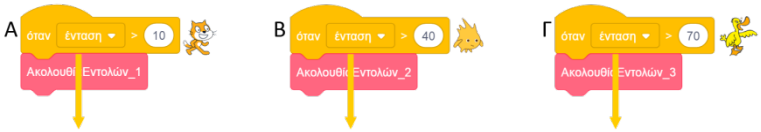


Σχήμα 7: *(α) Τα δύο σενάρια που τρέχουν ταυτόχρονα σε ένα πρόγραμμα με προσέγγιση παράλληλου προγραμματισμού και (β) το ισοδύναμο σειριακό πρόγραμμα.*

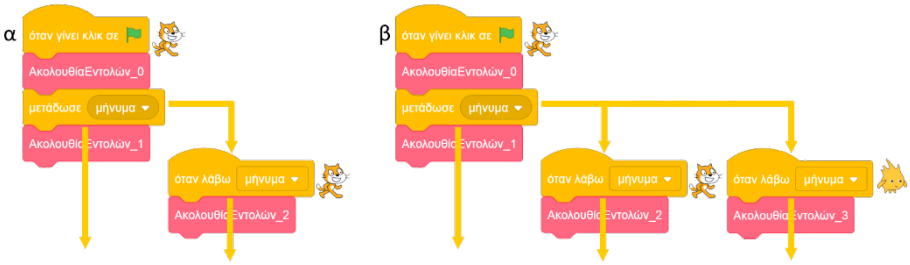
Πολυδομικό επίπεδο της SOLO και παράλληλα σενάρια στο Scratch. Στο πολυδομικό επίπεδο ο κώδικας χαρακτηρίζεται από παράλληλα σενάρια που λειτουργούν ως ανεξάρτητα (το ένα από το άλλο) νήματα (χωρίς την ανάγκη να επικοινωνούν / συγχρονίζονται μεταξύ τους), τα οποία δεν ξεκινούν ταυτόχρονα επειδή ενεργοποιούνται από διαφορετικές αιτίες, που μπορεί να είναι είτε εξωτερικά συμβάντα όπως η αλληλεπίδραση με το χρήστη (Σχήμα 8) ή συσκευές (Σχήμα 9) είτε εσωτερικά συμβάντα όπως η μετάδοση μηνυμάτων (Σχήμα 10) ή δημιουργία κλώνων (Σχήμα 11) ή τέλος συνδυασμός αυτών (Σχήμα 12).

ISBN: 978-960-578-084-5

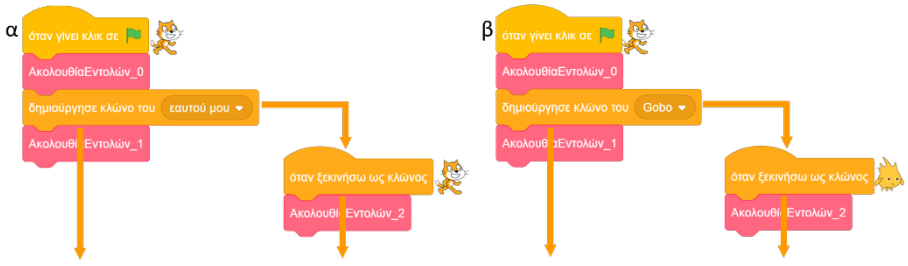
Σχήμα 8: *Δύο περιπτώσεις παράλληλων σεναρίων που ΔΕΝ ξεκινούν ταυτόχρονα, που είναι νήματα ανεξάρτητα το ένα από το άλλο, ενεργοποιούμενα από την αλληλεπίδραση με το χρήστη που μπορούν να κατανέμονται (α) στον ίδιο ή (β) σε διαφορετικούς ρόλους αντικειμένων.*



Σχήμα 9: *Παράλληλα σενάρια που ΔΕΝ ξεκινούν ταυτόχρονα, είναι ανεξάρτητα, ενεργοποιούμενα από συσκευή, κατανεμημένα σε διαφορετικούς ρόλους.*



Σχήμα 10: *Δύο παράλληλα σενάρια που ΔΕΝ ξεκινούν ταυτόχρονα, που είναι ανεξάρτητα νήματα το ένα από το άλλο, ενεργοποιούμενα από εσωτερικά συμβάντα όπως η μετάδοση μηνυμάτων, τα οποία μπορούν να κατανέμονται σε έναν ή σε διαφορετικούς ρόλους αντικειμένων.*

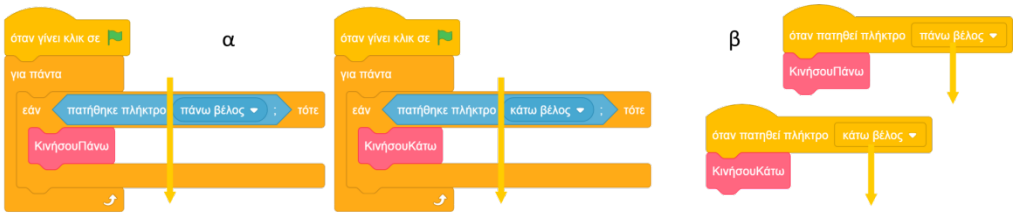


Σχήμα 11: *Δύο παράλληλα σενάρια που ΔΕΝ ξεκινούν ταυτόχρονα, που είναι ανεξάρτητα νήματα το ένα από το άλλο, ενεργοποιούμενα από εσωτερικά συμβάντα όπως η δημιουργία κλώνων.*

ISBN: 978-960-578-084-5

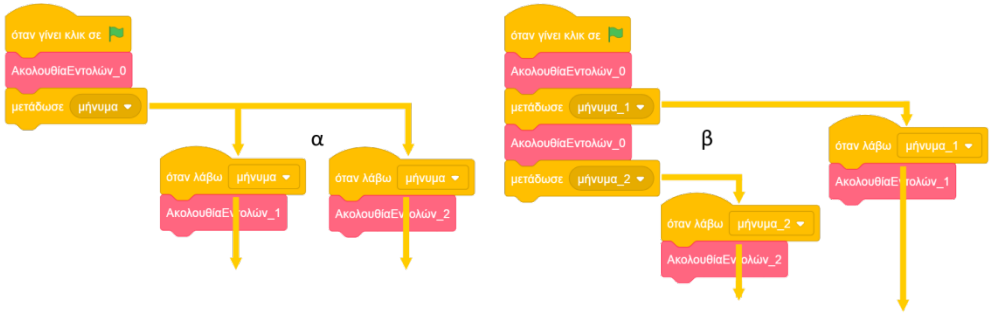
Σχήμα 12: *Τέσσερα παράλληλα σενάρια που ΔΕΝ ξεκινούν ταυτόχρονα, που είναι ανεξάρτητα νήματα το ένα από το άλλο, που ενεργοποιούνται από ποικιλία συμβάντων.*

Κατά την αξιολόγηση θα πρέπει να συνεκτιμηθεί κατά πόσο ο μαθητής μπορεί να διακρίνει μια λύση στην οποία τα παράλληλα σενάρια ξεκινούν ταυτόχρονα, είναι ενεργά σε κατάσταση εκτέλεσης (Bustard, 1990) και ανήκει στο μονοδομικό επίπεδο (Σχήμα 13α), από μια λύση στην οποία τα παράλληλα σενάρια δεν ξεκινούν ταυτόχρονα, είναι ενεργά σε κατάσταση επαγρύπνησης που ανήκει στο πολυδομικό επίπεδο της SOLO (Σχήμα 13β).



Σχήμα 13: *Διάκριση παράλληλων σεναρίων που (α) ξεκινούν ταυτόχρονα (μονοδομικό επίπεδο της SOLO) και (β) δεν ξεκινούν ταυτόχρονα (πολυδομικό επίπεδο της SOLO).*

Μια ανάλογη περίπτωση που κώδικες παραλλήλων σεναρίων διαφοροποιούνται ως προς το χρόνο ενεργοποίησης (ταυτόχρονα / μονοδομικό επίπεδο SOLO και διαδοχικά πολυδομικό επίπεδο SOLO) είναι αυτή του σχήματος 14.

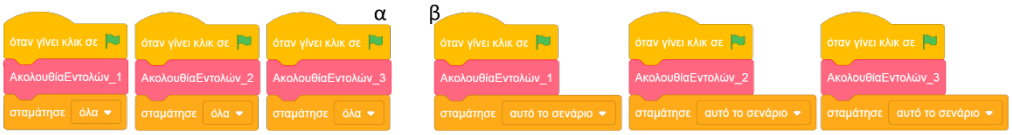


Σχήμα 14: *Δύο περιπτώσεις στις οποίες τα παράλληλα σενάρια ενεργοποιούνται: (α) ταυτόχρονα (μονοδομικό επίπεδο της SOLO) και (β) διαδοχικά (πολυδομικό επίπεδο της SOLO).*

Πέραν της εξέτασης για την έναρξη των παράλληλων σεναρίων, τα δύο παράλληλα προγράμματα του σχήματος 15 εξετάζονται στο χρόνο και τρόπο που αυτά τερματίζονται. Έτσι το πρόγραμμα του Σχήματος 15α τελειώνει μόλις κάποιο από τα τρία παράλληλα σενάρια τερματίσει πρώτο (μονοδομικό επίπεδο SOLO), ενώ το

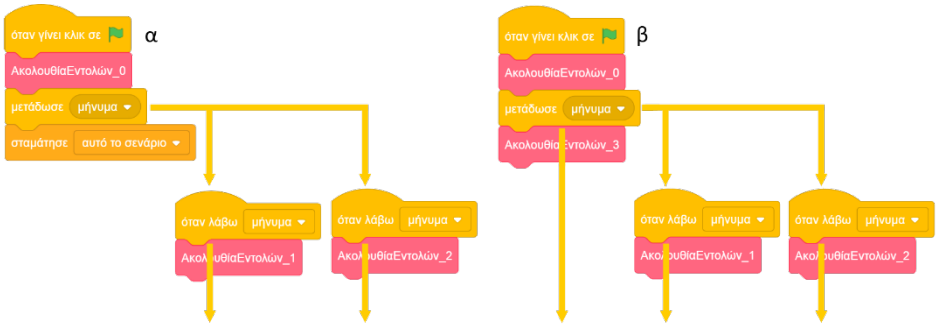
ISBN: 978-960-578-084-5

πρόγραμμα του Σχήματος 15β τελειώνει όταν τερματίσει και το τελευταίο παράλληλο σενάριο (πολυδομικό επίπεδο SOLO).



Σχήμα 15: *Δύο τρόποι τερματισμού παράλληλων σεναρίων: (α) το πρόγραμμα ολοκληρώνεται όταν τερματίσει πρώτο κάποιο σενάριο και (β) όταν τελειώσουν όλα τα σενάρια. Η πρώτη περίπτωση λειτουργεί σαν λογική σύζευξη ενώ η δεύτερη σαν λογική διάζευξη.*

Ένα επιπλέον σημείο που πρέπει να επισημανθεί κατά το ξεκίνημα των σεναρίων είναι το τι απογίνεται το "πατρικό" σενάριο. Στο Σχήμα 16α το "πατρικό" σενάριο μεταπίπτει σε κατάσταση λήξης (Bustard, 1990) μετά τη μετάδοση του μηνύματος και τη δημιουργία των δύο παράλληλων σεναρίων-απογόνων, ενώ στο Σχήμα 16β το "πατρικό" σενάριο συνεχίζει να "τρέχει" και αυτό παράλληλα με τα δύο νέα παράλληλα σενάρια-απογόνους.



Σχήμα 16: *Το πατρικό σενάριο μετά τη δημιουργία των παράλληλων σεναρίων (α) τερματίζεται ενώ στο (β) συνεχίζει να «τρέχει» παράλληλα με τα νέα παράλληλα σενάρια.*

Σχεσιακό επίπεδο της SOLO και παράλληλα σενάρια στο Scratch. Σύμφωνα με τον Bustard (1990) υπάρχουν δύο κύρια μοντέλα εκτέλεσης παράλληλων σεναρίων: (α) εκείνα όπου τα δημιουργούμενα παράλληλα σενάρια αυτονομούνται από το "πατρικό" σενάριο που τα δημιούργησε (και είναι αυτά που περιγράφηκαν στα προηγούμενα επίπεδα της ταξινομίας SOLO) και (β) εκείνα όπου τα δημιουργούμενα παράλληλα σενάρια αφού ολοκληρώσουν τα καθήκοντά τους επανέρχονται στη ροή του αρχικού σειριακού σεναρίου (Σχήμα 17).

Στο σχεσιακό επίπεδο ο κώδικας χαρακτηρίζεται από αλληλοεξαρτώμενα παράλληλα σενάρια, για τα οποία υπάρχει ανάγκη να επικοινωνούν μεταξύ τους. Στο Σχήμα 17β και στο Σχήμα 18, η εντολή "μετάδωσε... και περίμενε" διαθέτει ένα εσωτερικό

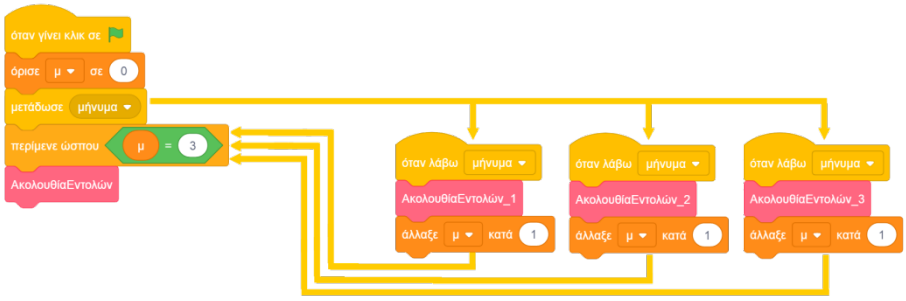
ISBN: 978-960-578-084-5

μηχανισμό που ελέγχει αν τερμάτισαν όλα τα παράλληλα σενάρια που ενεργοποιεί και τότε μόνο επιτρέπει να εκτελεστεί η επόμενη "ΑκολουθίαΕντολών".

Σχήμα 17: *Τα δύο κύρια μοντέλα εκτέλεσης παράλληλων σεναρίων όπου το δημιουργούμενο σενάριο: (α) αυτονομείται από το σενάριο που το δημιούργησε και (β) αφού ολοκληρώσει τα καθήκοντά του επανέρχεται στη ροή του αρχικού σεναρίου.*

Σχήμα 18: *Η εντολή «μετάδωσε … και περίμενε» εμπεριέχει ένα μηχανισμό ελέγχου του τερματισμού όλων των παράλληλων σεναρίων που πυροδότησε.*

Ένας ισοδύναμος τρόπος ελέγχου μιας ίδιας τοπολογίας παραλλήλων σεναρίων μπορεί να επιτευχθεί με την τεχνική του διαμοιρασμού μνήμης που χρησιμοποιεί καθολικές (global) μεταβλητές στις οποίες έχουν πρόσβαση όλα τα αντικείμενα του προγράμματος (Σχήμα 19).

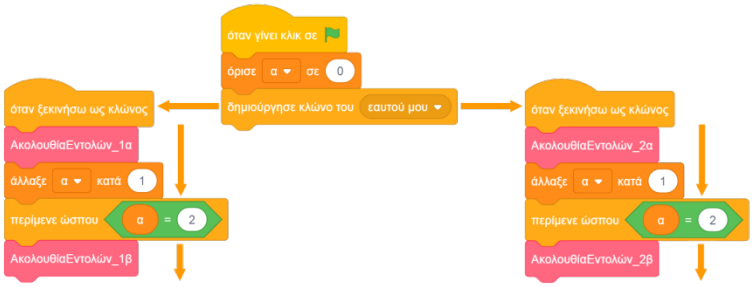


Σχήμα 19: *Μηχανισμός επικοινωνίας (κάνοντας χρήση της τεχνικής διαμοιρασμού μνήμης με τη βοήθεια της καθολικής μεταβλητής "α") μεταξύ του αρχικού σεναρίου και των τριών σεναρίων που εκτελούνται παράλληλα. Το αρχικό σενάριο θα περιμένει να ολοκληρωθούν και τα τρία παράλληλα σενάρια πριν συνεχίσει στην εκτέλεση της διαδικασίας «ΑκολουθίαΕντολών».*

«Κατά την παράλληλη εκτέλεση των νημάτων υπάρχει πολλές φορές και η ανάγκη για συγχρονισμό, δηλαδή για αναμονή των νημάτων σε προκαθορισμένα σημεία του κώδικα έως ότου, ακόμα και τα πιο αργά νήματα να φτάσουν εκεί ή κάποιο σημαντικό

ISBN: 978-960-578-084-5

γεγονός να συμβεί» (Δημακόπουλος, 2017). Σε αυτή την περίπτωση κατά την οποία παράλληλα σενάρια πρέπει σε κάποιο σημείο τους να συγχρονιστούν χρησιμοποιείται μια τεχνική με διαμοιραζόμενη καθολική μεταβλητή (Σχήμα 20).



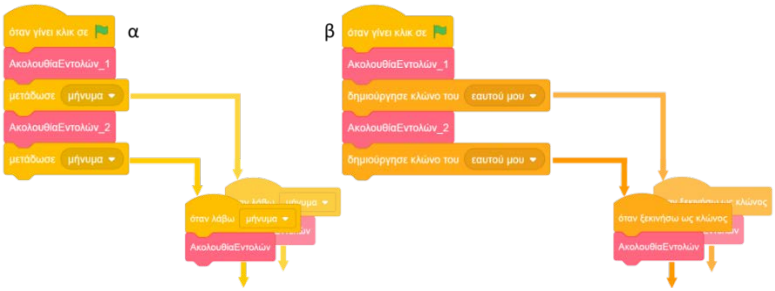
Σχήμα 20: *Πρόγραμμα στο οποίο δύο παράλληλα σενάρια συγχρονίζονται κατά την εκτέλεσή τους, με τη βοήθεια της καθολικής μεταβλητής "α", ώστε να συνεχίσουν ταυτόχρονα στις διαδικασίες «ΑκολουθίαΕντολών_1β» και «ΑκολουθίαΕντολών_2β».*

Ο συγχρονισμός μεταξύ των σεναρίων που τρέχουν παράλληλα προϋποθέτει τη μεταξύ τους σύγχρονη ή ασύγχρονη επικοινωνία. Στη σύγχρονη επικοινωνία τα σενάρια συγχρονίζονται με την ανταλλαγή μηνυμάτων (Σχήματα 17β και 18), ενώ στην ασύγχρονη ένα σενάριο που παρέχει δεδομένα μπορεί να τα αποθέσει προσωρινά σε ένα buffer επικοινωνίας (μια κοινή δομή δεδομένων) στον οποίον ένα άλλο σενάριο-αποδέκτης θα τα αναζητήσει όταν θα είναι σε θέση να τα λάβει (Σχήματα 19 και 20).

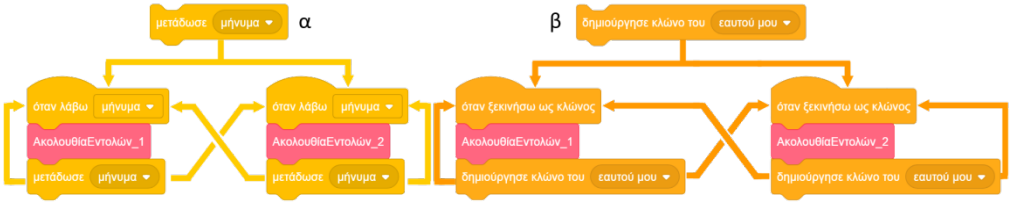
Σχεσιακό επίπεδο της SOLO και παράλληλα σενάρια στο Scratch. Σε όλες τις προηγούμενες τοπολογίες προγραμμάτων τα σενάρια που εκτελούνταν παράλληλα ήταν διαφορετικά / διακριτά, π.χ. στο πρόγραμμα του Σχήματος 18 υπήρχαν τρία διαφορετικά σενάρια που εκτελούνταν παράλληλα. Όμως είναι δυνατόν το ίδιο σενάριο να κληθεί προς εκτέλεση πολλαπλές φορές και έτσι να "τρέχει" παράλληλα με τον εαυτό του όπως στο Σχήμα 21 (να υπενθυμίσουμε εδώ αυτό που αναφέρθηκε στην αρχή: "... κάθε διαδικασία «ΑκολουθίαΕντολών…» μπορεί να αντιστοιχίζεται σε ένα βρόχο «Επανάλαβε ώσπου…» ώστε να υποδηλώνεται μια απροσδιόριστη χρονική διάρκεια εκτέλεσης της διαδικασίας"). Στην περίπτωση αυτή έχουμε τη δημιουργία δύο νημάτων που διαθέτουν τον ίδιο κώδικα αλλά διαφορετικά δεδομένα κάθε φορά.

Στην προηγούμενη περίπτωση αλλά και σε αυτές των Σχημάτων 22 και 23 κατά την εκτέλεση του προγράμματος μπορεί να δημιουργηθούν απροσδιόριστες καταστάσεις, που να οδηγήσουν σε μια ανεξέλεγκτη μη-αιτιοκρατική συμπεριφορά του προγράμματος.

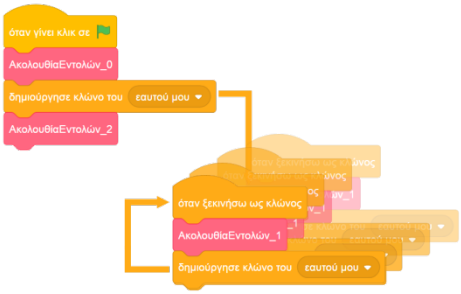
ISBN: 978-960-578-084-5



Σχήμα 21: *Παραλληλία μέσω της πολλαπλής εκτέλεσης του ιδίου σεναρίου υλοποιούμενου: (α) με μηνύματα και (β) με κλώνους.*



Σχήμα 22: *Παράλληλα προγράμματα που παρουσιάζουν κατά την εκτέλεσή τους μη-αιτιοκρατική συμπεριφορά υλοποιούμενα (α) με μηνύματα και (β) με κλώνους.*



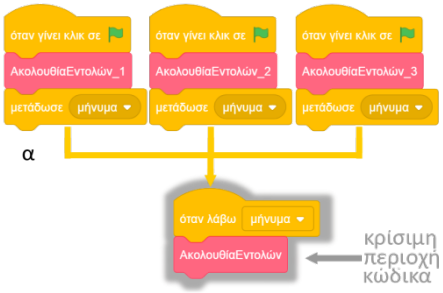
Σχήμα 23: *Παράλληλη εκτέλεση του ιδίου σεναρίου που καθορίζει τη συμπεριφορά κλώνου, η οποία παρουσιάζει μη-αιτιοκρατική συμπεριφορά.*

Τέτοιας φύσης μη αιτιοκρατικά προβλήματα παρουσιάζονται όταν σενάρια διεκδικούν την πρόσβασή σε κοινόχρηστους πόρους π.χ. μια κρίσιμη περιοχή κώδικα η οποία μπορεί να δίνει πρόσβαση σε μια λίστα δεδομένων (Σχήμα 24).

Σύμφωνα με το Δημακόπουλο (2017), «ο έλεγχος και η διαχείριση τέτοιων ανεξέλεγκτων καταστάσεων είναι ευθύνη του προγραμματιστή. Αυτός θα πρέπει να αναγνωρίσει τις κρίσιμες περιοχές κώδικα και σε αυτόν επαφίεται η χρήση ή όχι κατάλληλων μηχανισμών...». Ο Bustard (1990) σχετικά με τη διαχείριση κοινόχρηστων πόρων αναφέρει ότι «μια διαδικασία που επιθυμεί να χρησιμοποιήσει έναν κοινόχρηστο πόρο πρέπει πρώτα να λάβει άδεια πρόσβασης σε αυτόν, ενώ όταν ο πόρος δεν απαιτείται πλέον, να τον απελευθερώνει. Εάν μια διαδικασία δεν είναι σε θέση να αποκτήσει έναν πόρο, τότε η εκτέλεση της αναστέλλεται έως ότου αυτός ο

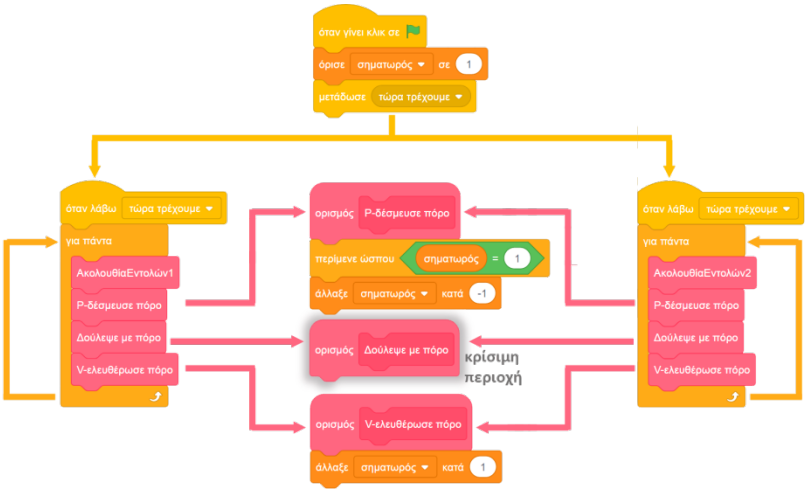
ISBN: 978-960-578-084-5

πόρος είναι διαθέσιμος. Επίσης, ορισμένοι πόροι μπορεί να αποκτηθούν από μια διαδικασία για αποκλειστική χρήση, ενώ άλλοι μπορεί να κοινοποιηθούν εάν χρησιμοποιούνται με συγκεκριμένο τρόπο. Για παράδειγμα, πολλές διαδικασίες μπορούν να διαβάσουν ταυτόχρονα το περιεχόμενο μιας δομής δεδομένων, αλλά μόνο μία διαδικασία κάθε φορά μπορεί να το τροποποιήσει / γράψει...».



Σχήμα 24: *Πρόγραμμα όπου τρία παράλληλα σενάρια, διεκδικούν πρόσβαση σε μια κρίσιμη περιοχή κώδικα με αποτέλεσμα τη μη-προβλέψιμη συμπεριφορά τους*

Ένας από τους μηχανισμούς διαχείρισης τέτοιων προβλημάτων που χρησιμοποιούνται συχνά για τον έλεγχο του αμοιβαίου αποκλεισμού μεταξύ παραλλήλων σεναρίων είναι οι σηματωροί (Ben-Ari, 1982). Αυτοί είναι απλές διαμοιραζόμενες μεταβλητές, με μια όμως υψηλής αφαίρεσης νοητική αναπαράσταση, που σκοπό έχουν την προστασία ενός κοινόχρηστου πόρου.



Σχήμα 25: *Παράδειγμα σε Scratch μηχανισμού ελέγχου της πρόσβασης σε κρίσιμη περιοχή / κοινόχρηστο πόρο, από δύο ανταγωνιστικά παράλληλα σενάρια, με τη χρήση σηματωρού και των διαδικασιών P και V.*

Οι σηματωροί διαχειρίζονται είτε την πρόσβαση ενός παράλληλου σεναρίου (που ανταγωνίζεται άλλα) σε μια κρίσιμη περιοχή κώδικα ή πόρο, είτε την απεμπλοκή του

ISBN: 978-960-578-084-5

από αυτόν. Αυτό επιτυγχάνεται αντίστοιχα με τις λειτουργίες P ("wait if necessary") και V ("signal") (Dijkstra 1968). Ένα παράδειγμα του τρόπου λειτουργίας των σηματωρών για το Scratch φαίνεται στο Σχήμα 25. Στο επίπεδο της εκτεταμένης γενίκευσης κατά την αξιολόγηση θα πρέπει να εκτιμηθεί κατά πόσον ο μαθητής μπορεί να αναγνωρίσει την ύπαρξη παράλληλων σεναρίων που ανταγωνίζονται για πρόσβαση σε μια κρίσιμη περιοχή κώδικα ή σε ένα κοινόχρηστο πόρο και με ποιούς τρόπους επιλύει αυτό το πρόβλημα.

3. Συμπεράσματα

Στο Scratch, ένα προγραμματιστικό περιβάλλον οπτικού προγραμματισμού με πλακίδια, που βασίζεται σε αντικείμενα (object based) και είναι προσανατολισμένο στον προγραμματισμό βασισμένο σε γεγονότα (event driven programming), παρέχεται η δυνατότητα να συντρέχουν ταυτόχρονα διεργασίες-σενάρια (στο εσωτερικό των οποίων μπορούν να εφαρμόζονται τεχνικές δομημένου προγραμματισμού) με ψευδοπαράλληλο τρόπο.

Τα συντρέχοντα / παράλληλα σενάρια διαφοροποιούνται ως προς τα αίτια που τα πυροδοτούν, ως προς τον τρόπο που ενεργοποιούνται ή τερματίζονται και ως προς το χρόνο που ξεκινούν να εκτελούνται ή να τελειώνουν, ως προς τον τρόπο που επικοινωνούν (σύγχρονα ή ασύγχρονα) μεταξύ τους και συγχρονίζονται, αν ανήκουν στο ίδιο ή σε διαφορετικά αντικείμενα ή κλώνους, σχηματίζοντας λόγω του οπτικού προγραμματισμού τοπολογίες. Οι διάφορες τοπολογίες του παράλληλου προγραμματισμού που μπορούν να χρησιμοποιηθούν στο Scratch, κατατάσσονται στα πέντε επίπεδα της ταξινομίας SOLO όπως φαίνεται στον Πίνακα 1.

Πίνακας 1. *Κατάταξη σε επίπεδα της SOLO των τοπολογιών παράλληλου προγραμματισμού στο Scratch.*

Επίπεδα SOLO	**Περιγραφή σεναρίων. Προγράμματα με παράλληλα / συντρέχοντα σενάρια που…**
προδομικό	… δεν λαμβάνουν υπόψη τους τη μη-αιτιοκρατική συμπεριφορά τους ή είναι σειριακά προγράμματα.
μονοδομικό	… λειτουργούν ανεξάρτητα και ξεκινούν ταυτόχρονα, ενεργοποιούμενα από την ίδια αιτία.
πολυδομικό	… λειτουργούν ανεξάρτητα που δεν ξεκινούν ταυτόχρονα επειδή ενεργοποιούνται από διαφορετικά αίτια.
σχεσιακό	… είναι αλληλοεξαρτώμενα και κατά συνέπεια υπάρχει ανάγκη να επικοινωνούν μεταξύ τους.
εκτεταμένης γενίκευσης	… επιβάλλεται ο προγραμματιστής να επιλύσει προβλήματα όπως η (χωρίς αμοιβαίο αποκλεισμό) πρόσβαση σε κρίσιμη περιοχή κώδικα διαχείρισης κοινόχρηστου πόρου.

ISBN: 978-960-578-084-5

Η υιοθέτηση αυτής της κατάταξης των σεναρίων (με κριτήριο τον τρόπο που αυτά χρησιμοποιούνται στο προγραμματιστικό στυλ του παράλληλου προγραμματισμού) μπορεί να οδηγήσει αφενός στην ανάπτυξη αξόνων στους οποίους μπορεί να αξιοποιήσει η ανάπτυξη ενός προγράμματος σπουδών βασισμένου στον προγραμματισμό Η/Υ για την υποχρεωτική εκπαίδευση (Λαδιάς & Γώγουλος, 2017) και αφετέρου σε ένα σύστημα αξιολόγησης με το οποίο ο εκπαιδευτικός μπορεί να κρίνει με μετρήσιμα κριτήρια τον κώδικα που γράφουν οι μαθητές του όσον αφορά το βαθμό παραλληλίας των σεναρίων. Η παρούσα εργασία εντάσσεται στα πλαίσια ενός έργου αξιολόγησης του κώδικα Scratch, αλλά και σε ένα ευρύτερο πλαίσιο αξιολόγησης έργων εκπαιδευτικής ρομποτικής.

Τέλος σχεδιάζονται δύο έρευνες, η μία μεταξύ εκπαιδευτικών πληροφορικής, σχετική με τη διδακτική αξιοποίηση του παράλληλου προγραμματισμού σε περιβάλλον Scratch σε συνδυασμό με τη χρήση του ΚωδικΟράματος (Λαδιάς & Λαδιάς, 2016) στην υποχρεωτική εκπαίδευση και η άλλη εργασία σχετική με τον τρόπο που χρησιμοποιούν οι μαθητές τον παράλληλο προγραμματισμό στον κώδικά τους για να ελέγξουν ρομποτικές κατασκευές στην οποία θα αξιοποιηθεί το αποθετήριο έργων του Πανελλήνιου Διαγωνισμού Εκπαιδευτικής Ρομποτικής του WRO-Hellas.

Αναφορές

Ben-Ari, M. (1982). *Principles of Concurrent Programming.* Prentice-Hall International.

Biggs, J. B., & Collis, K. F. (1982). *Evaluating the quality of learning. The SOLO taxonomy*. NY: Academic Press.

Bustard, W., D. (1990). Concepts of Concurrent Programming. *Software Engineering Institute / Carnegie Mellon University (SEI-CM-24).*

Dijkstra, E. W. (1968). Cooperating Sequential Processes. Programming Languages, F. Genuys, ed. Academic Press, 43-112.

Jimoyiannis, A. (2011). Using SOLO taxonomy to explore students' mental models of the programming variable and the assignment statement. *Themes in Science & Technology Education, 4(2)*, 53-74.

Kahn, K. (1996). Drawing on napkins, video-game animation, and other ways to programm computers. *Communications of the ACM, 39(8)*, 49-59.

Karvounidis, T., Argyriou, I., Ladias, A., Douligeris, Chr. (2017). A Design and Evaluation Framework for Visual Programming Codes. *The IEEE Global Engineering Education Conference (EDUCON2017)*. Athens.

Meerbaum-Salant, O., Armoni, M.,& Ben-Ari, M. (2013). Learning computer science concepts with Scratch. *Computer Science Education, 23(3)*, 239-264.

ISBN: 978-960-578-084-5

Δημακόπουλος, Β. (2017). *Παράλληλα Συστήματα και Προγραμματισμός*. ΣΕΑΒ. Ζωγράφου.

Λαδιάς, Α., Γώγουλος, Γ. (2017). Ο προγραμματισμός υπολογιστικών συσκευών ως κύριος άξονας ενός Προγράμματος Σπουδών στην Πληροφορική. *Έρκυνα, 13*, 158-168.

Λαδιάς, Δ., Καρβουνίδης, Θ., Λαδιάς, Α. (2020). Αξιοποίηση τεχνικών παράλληλου προγραμματισμού στο περιβάλλον Scratch. *12th CIE2020 - Η Πληροφορική στην Εκπαίδευση*. Πειραιάς.

Λαδιάς, Α. & Λαδιάς, Δ. (2016). Η αναπαράσταση του αλγορίθμου με τη βοήθεια του κωδικΟράματος σε περιβάλλοντα οπτικού προγραμματισμού. *Θέματα Επιστημών και Τεχνολογίας στην Εκπαίδευση, 9(2),* (103-117).

Μπέλλου, Ι. & Μικρόπουλος, Α. (2008). Μέθοδος για την Ιεραρχική Αξιολόγηση Γνώσεων Προγραμματισμού. *4ο Πανελλήνιο Συνέδριο Διδακτική της Πληροφορικής*, Πάτρα.

Abstract

In the Scratch programming environment, more than one scenarios can run concurrently. The interdependence between these concurrently running scenarios, the communication between them for synchronization, the reasons that activate or disactivate them and the thread flows that are formed, they create parallel programming code topologies. In the present work, we attempt to group these topologies based on internal criteria, aiming to correspond to the five levels of the SOLO taxonomy. The adoption of this classification may lead on the one hand to the development of a curriculum based on computer programming for the compulsory education and on the other hand to an assessment system with measurable criteria based the degree of parallelism of scenarios of the student codes. The present work is part of a code assessment but in a broader context it may be extended to the assessment of educational robotics projects.

Keywords: Parallel programming, Scratch, SOLOtaxonomy.

ISBN: 978-960-578-084-5