STEMKIT 4SCHOOLS

ΣΧΕΔΙΑΣΗ & ΑΝΑΠΤΥΞΗ ΤΟΥ ΟΔΗΓΟΥ ΣΠΟΥΔΩΝ STEMKIT

Αναγνωριστικό Αποτελέσματος: O2A1

Co-funded by the Erasmus+ Programme of the European Union

Αυτό το έργο χρηματοδοτήθηκε με την υποστήριξη της Ευρωπαϊκής Επιτροπής.

Πίνακας Περιεχομένων

1 Γενική εισαγωγή στο πρόγραμμα

Τα παιδιά σήμερα γεννιούνται μέσα στην τεχνολογία, γι' αυτό και η χρήση της τους έρχεται φυσικά. Παρ' όλα αυτά, χρειάζεται να αποκτήσουν τεχνολογικές δεξιότητες, όπως ο προγραμματισμός. Το εργατικό δυναμικό που κατέχει δεξιότητες STEM έχει υψηλή ζήτηση στην Ευρώπη, η οποία θα συνεχίσει να αυξάνεται λόγω της έλευσης της Βιομηχανίας 4.0 και των Ανεπτυγμένων Βιομηχανικών Τεχνολογιών. Νέοι τρόποι προσέλκυσης παιδιών στον προγραμματισμό και το STEM είναι απαραίτητοι, ωστόσο, ο περισσότερος χρόνος μπροστά στις οθόνες δεν είναι η καλύτερη προσέγγιση. Το ενεργό παιχνίδι είναι πιο διασκεδαστικό και, πολλές φορές, πιο εκπαιδευτικό.
Η γεφύρωση των «online» και των «offline» κόσμων δύναται να προσφέρει ένα πιο συναρπαστικό και υγιές περιβάλλον για τα παιδιά, ούτως ώστε να μάθουν προγραμματισμό και ν' αναπτύξουν δεξιότητες STEM.

Το πρόγραμμα Erasmus+, STEMKIT4Schools, έχει ως βασικό του στόχο την παραγωγή προσεγγίσεων κι εργαλείων, που θα υποστηρίξουν όσους δουλεύουν με παιδιά να τα πλησιάσουν και να τα βοηθήσουν στην ενασχόληση με τον προγραμματισμό και στην ανάπτυξη δεξιοτήτων σχετικών με το STEM. Ο στόχος αυτός δε θα επιτευχθεί αυξάνοντας το χρόνο μπροστά στην οθόνη, αλλά ενθαρρύνοντας το ενεργό παιχνίδι, μέσω της δημιουργίας παιχνιδιών ικανών να παιχτούν σε ένα ρετρό, DIY (κάν'το-μόνος-σου) ξύλινο υπολογιστή σε συνδυασμό με σχετικές με θέματα STEM ηλεκτρονικές μικροσυσκευές.

2 Στόχοι αυτού του οδηγού σπουδών

Ένας ολοκληρωμένος οδηγός σπουδών, που ανταποκρίνεται στους στόχους του STEMKIT4Schools, σχεδιάστηκε από τους συνεργάτες του προγράμματος, και περιλαμβάνει προγράμματα μαθημάτων για τη χρήση των Minecraft Pi/Scratch/Python/Kits με τον υπολογιστή STEMKIT στην τάξη. Τα προγράμματα μαθημάτων αποτελούν μέρος του Οδηγού Εκπαιδευτών για δασκάλους. Τα σετ ηλεκτρονικών είναι σχεδιασμένα και κατασκευασμένα για να συμπληρώνουν τη διδασκαλία του προγραμματισμού, του εμπράγματου προγραμματισμού και των μαθημάτων STEM.

Το τελευταίο μάθημα STEMKIT διδάσκει βασικά στοιχεία παιχνιδιών με τα Minecraft Pi, Python, και Scratch, όπως επίσης εμπράγματο προγραμματισμό (με τη χρήση βασικών ψηφιακών, αναλογικών και ηλεκτρομηχανικών στοιχείων) και συνεργασία (ενασχόληση και μοίρασμα με άλλους).

Ο DIY υπολογιστής STEMKIT βασίζεται στο Raspberry Pi, που σημαίνει ότι μπορεί να αξιοποιήσει ισχύ και ουσιαστικά απεριόριστους πόρους για το Raspberry, ενώ τα χαρακτηριστικά του λειτουργικού συστήματος Raspbian παρέχουν επιπλέον δυνατότητες.

Ο αναθεωρημένος οδηγός σπουδών STEMKIT, χρησιμοποιώντας αρχές βιομηχανικού σχεδιασμού, μπορεί να βρεθεί στην Πύλη Μάθησης υπό τη μορφή διαδραστικών Στόχων Μάθησης, και θα παραδοθεί στους μαθητευόμενους, οι οποίοι θα μπορούν να ακολουθήσουν το διαδραστικό μάθημα στο δικό τους χρόνο και τόπο, ενώ θα μπορούν να αξιοποιήσουν τα εργαλεία κοινωνικής μάθησης που προσφέρονται για την ενασχόληση με τους

συνομηλίκους τους (ενασχόληση με σχόλια, forum, chat). Επίσης, θα προσφέρονται επιπλέον χαρακτηριστικά συνεργασίας, όπως ειδοποιήσεις, ομαδική ροή ειδήσεων, blog, μοίρασμα αρχείων, ερωτήσεις/απαντήσεις, ψηφοφορίες.

Στις επόμενες ενότητες παρουσιάζονται τα εξής 5 θέματα: Εισαγωγή στο Scratch 2.0 [P6-ARC], το Scratch GPIO (Έλεγχος ακίδων GPIO/λήψη καταχωρήσεων) [P3-DANMAR], Εισαγωγή στην Έκδοση Raspberry Pi του Minecraft [P4-HESO / P5-SCHOLE], Προγραμματισμός του Raspberry Pi GPIO με τη χρήση Python [P2-AKNOW], και Εμπράγματος Προγραμματισμός [P1-ECAM].

3 Εισαγωγή στο Scratch 2.0 [P6-ARC]

3.1 Λεξικό όρων

Όρος / Έννοια	Ορισμός/ Εξήγηση
Scratch 2.0	Το Scratch είναι μια γλώσσα προγραμματισμού, που δημιουργήθηκε από το Εργαστήρι ΜΜΕ του MIT, και είναι ένα περιβάλλον ανάπτυξης λογισμικού ανοιχτού κώδικα, το οποίο διευκολύνει τη δημιουργία διαδραστικής τέχνης, ιστοριών, προσομοιώσεων και παιχνιδιών. Έχει σκοπό να εκπαιδεύσει άτομα με λίγη ή καμία εμπειρία στον προγραμματισμό, και κυρίως παιδιά ηλικιών 8 έως 16. Το Scratch 2.0, επίσης γνωστό και ως Scratch 2,[1], που ήταν η δεύτερη σημαντικότερη εκδοχή του Scratch, μετά το Scratch 1.4., διέθετε επανασχεδιασμένο μεταγλωττιστή γλώσσας προγραμματισμού και ιστοσελίδα, και ήταν η πρώτη εκδοχή που περιλάμβανε μεταγλωττιστή γλώσσας προγραμματισμού και σε σύνδεση (online) και εκτός σύνδεσης (offline). Το Scratch ξαναγράφτηκε πλήρως στο Adobe Flash για την έκδοση 2.0, αλλά ακόμη εκτελούσε προγράμματα από παλαιότερες εκδόσεις του Scratch. Παρέμενε εντελώς δωρεάν και χωρίς διαφημίσεις.
Blocks (2.0)	Τα Μπλοκ (τετράγωνα εντολών με γραφικά) ήταν σχήματα σαν κομμάτια παζλ, που χρησιμοποιούνταν για τη δημιουργία κώδικα στο Scratch. Τα μπλοκ συνδέονταν μεταξύ τους κάθετα, όμοια με παζλ, όπου ο κάθε τύπος δεδομένων (hat, stack, reporter, boolean, cap) είχε τη δική του μορφή και μια ειδικά διαμορφωμένη υποδοχή στην οποία μπορούσε να εισαχθεί, κάτι που απέτρεπε τα λάθη συντακτικού. Οι σειρές των συνδεδεμένων μπλοκ λέγονταν σενάρια (scripts).
Επεξεργασία χρωμάτων των Μπλοκ	Στον online μεταγλωττιστή γλώσσας προγραμματισμού Scratch 2.0, πατώντας το πλήκτρο «shift» και κάνοντας κλικ το ποντίκι στο μενού επεξεργασίας, εμφανιζόταν μια επιλογή που λεγόταν «Επεξεργασία χρωμάτων των μπλοκ». Με την επιλογή αυτή, εμφανιζόταν ένα μενού με 3 ρυθμιστές HSL και εργαλεία για την τροποποίηση των χρωμάτων των μπλοκ μιας συγκεκριμένης κατηγορίας μπλοκ.

3.2 Κυρίως περιεχόμενο

Το **Scratch** είναι μια **οπτική γλώσσα προγραμματισμού** που βασίζεται σε μπλοκ (τετράγωνα εντολών με γραφικά) και μια ιστοσελίδα απευθυνόμενη κυρίως σε παιδιά, που επιτρέπει στους χρήστες να μάθουν προγραμματισμό, ενώ δουλεύουν πάνω σε έργα με προσωπικό νόημα, όπως κινούμενες ιστορίες και παιχνίδια (Maloney, 2010).

Το Scratch χρησιμοποιείται από σχολεία σε πολλαπλούς τομείς (μαθηματικά, πληροφορική, εκμάθηση γλωσσών, κοινωνιολογία).

Εικόνα 1. Στιγμιότυπο οθόνης από πρόγραμμα στο Scratch (Πηγή https://www.thomasbuxton.towerhamlets.sch.uk/blogs/year3/2017/11/17/year-3-computing-scratch-projects/)

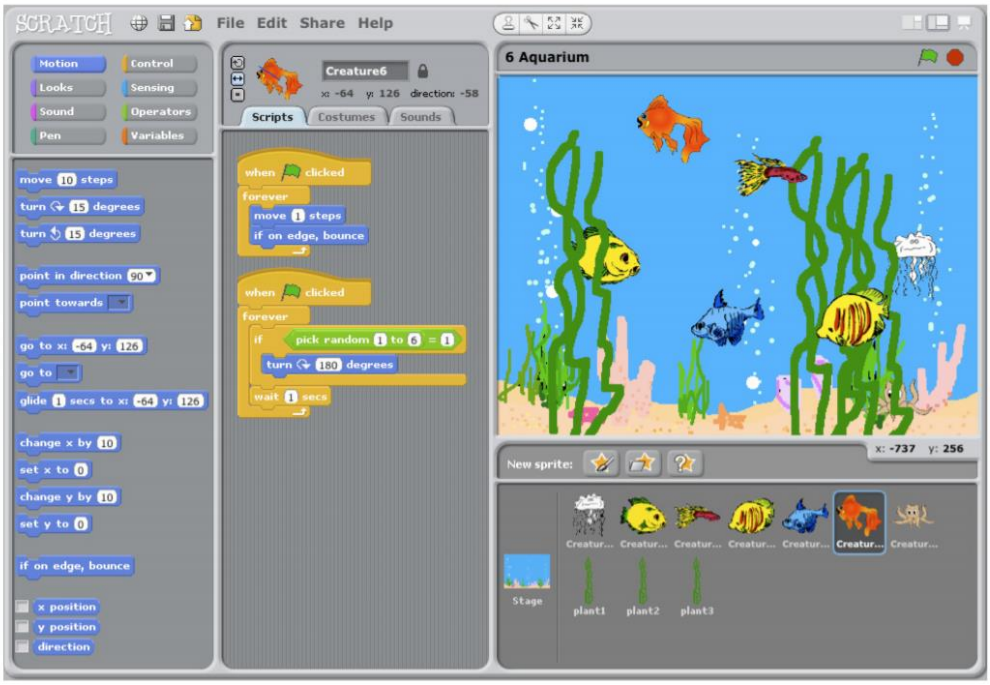
Ο αρχικός σχεδιασμός του Scratch «γεννήθηκε» από τις ανάγκες και τα ενδιαφέροντα νέων ανθρώπων (ηλικιών 8 έως 16) σε εξωσχολικά κέντρα υπολογιστών, όπως τα Intel Computer Clubhouses (Resnick, 2003). Στην αρχή, το Scratch χρησιμοποιούνταν κυρίως σε ανεπίσημους χώρους μάθησης, όπως σε πολιτιστικά κέντρα, εξωσχολικές λέσχες, βιβλιοθήκες και σπίτια˙ χρησιμοποιούνταν επίσης και σε σχολεία. (*http://web.media.mit.edu/~jmaloney/papers/ScratchLangAndEnvironment.pdf*)

3.2.1 Εισαγωγή στις βασικές λειτουργίες του Scratch 2.0

Η γλώσσα προγραμματισμού Scratch ξεκίνησε το 2003, ενώ το λογισμικό Scratch κυκλοφόρησε στο ευρύ κοινό το 2007. Το Scratch είναι δωρεάν, διαθέσιμό σε σχεδόν 50 γλώσσες, και συχνά αναδιανέμεται από σχολικά συστήματα κι εκπαιδευτικούς οργανισμούς.

- ✓ Ένας σημαντικός στόχος του Scratch είναι να παρουσιάσει τον προγραμματισμό σε όσους δεν έχουν προηγούμενη εμπειρία με αυτόν.
- ✓ Το Scratch απευθύνεται σε νεότερους χρήστες σε σχέση με τα άλλα δυο συστήματα, εστιάζει στην αυτο-κατευθυνόμενη μάθηση και περιλαμβάνει εργαλεία για τη ζωγραφική εικόνων και την καταγραφή ήχων.

- ✓ Το Scratch βασίζεται στις κατασκευαστικές ιδέες του Logo (Kay, 2010, Steinmetz, 2002), ώστε να βοηθήσει τους χρήστες να κάνουν τα έργα τους ελκυστικά, παρακινητικά και ουσιαστικά.
- ✓ Το Scratch διευκολύνει την εισαγωγή ή τη δημιουργία πολλών ειδών πολυμέσων (εικόνες, ήχοι, μουσική): σχεδιάστηκε ώστε να ενθαρρύνει τη γλώσσα προγραμματισμού, να παρέχει άμεση ανατροφοδότηση (feedback) για την εκτέλεση των σεναρίων, αλλά και να εμφανίζει την εκτέλεση και τα δεδομένα.
- ✓ Το περιβάλλον χρήστη του Scratch επιδιώκει να διευκολύνει την πλοήγηση.
- ✓ Το Scratch διαμορφώνεται εύκολα, επειδή αφήνει τους χρήστες να πειραματιστούν με εντολές και αποσπάσματα κώδικα, με τον ίδιο τρόπο που κάποιος θα «πείραζε» μηχανικά ή ηλεκτρονικά εξαρτήματα.
- ✓ Στο Scratch, ο χρήστης δεν απαιτείται να δημιουργήσει ολοκληρωμένα σενάρια πριν να «τρέξει» τα προγράμματα. Αποσπάσματα του έργου μπορούν να αφεθούν στην οθόνη καταχώρησης κώδικα και σώζονται με το πρόγραμμα (Maloney, 2010).

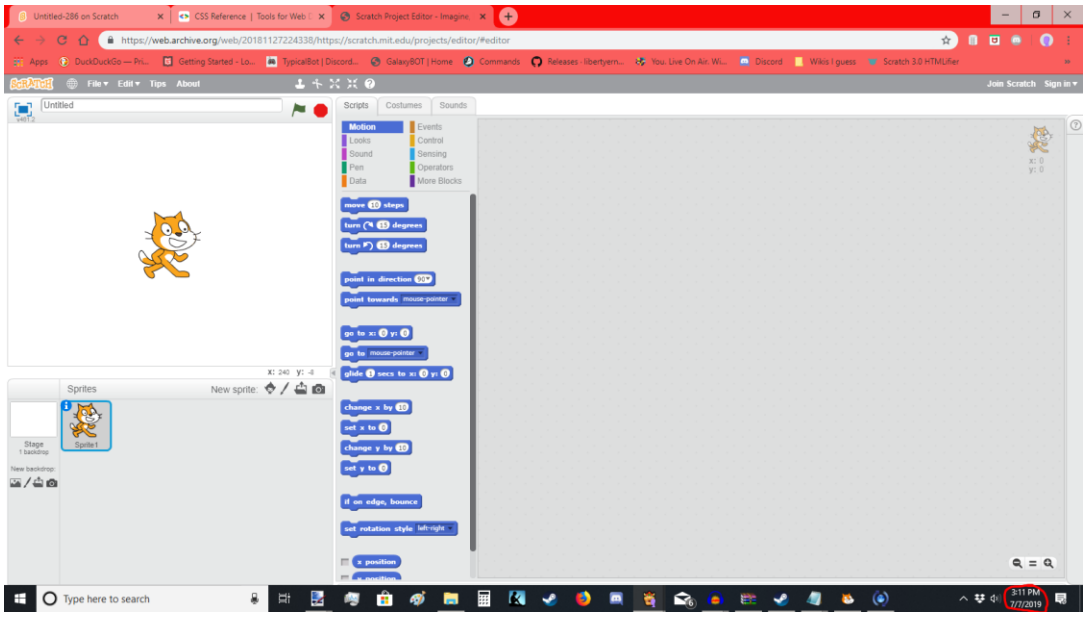


Εικόνα 2. Το περιβάλλον χρήστη του Scratch (Maloney, 2010)

- ✓ Το Scratch παρέχει εικονική ανατροφοδότηση, που δείχνει την εκτέλεση του κώδικα.
- ✓ Το Scratch εμφανίζει επίσης την ακολουθία εντολών και τη ροή ελέγχου. Η ενεργοποίηση του single-stepping (επιλέγοντάς το απ' το μενού) κάνει τα μπλοκ να αναβοσβήνουν καθώς «τρέχουν».
- ✓ Είναι σχεδιασμένο να χρησιμοποιεί ένα μοναδικό παράθυρο και πολλαπλές σελίδες, ώστε να εξασφαλίσει ότι τα βασικά μέρη είναι πάντα ορατά.
- ✓ Το Scratch αποφεύγει τις κινητές παλέτες: στην αριστερή περιοχή είναι η παλέτα εντολών με κουμπιά για την επιλογή κατηγοριών, ενώ στη μέση τα σενάρια για το τρέχον επιλεγμένο αντικείμενο, με καρτέλες φακέλων για την προβολή κι επεξεργασία των ενδυμασιών (εικόνων) και των ήχων που κατέχει αυτό το αντικείμενο. Η μεγάλη περιοχή πάνω δεξιά είναι το σκηνικό, όπου λαμβάνει χώρα η δράση. Η

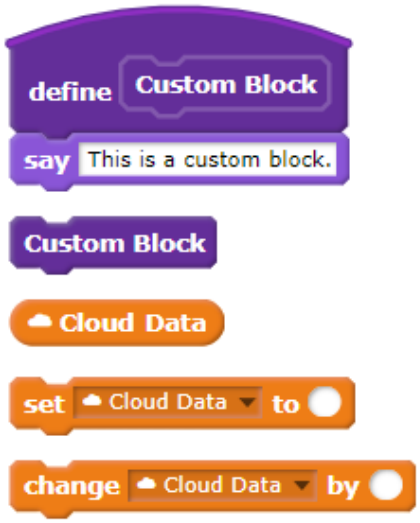
περιοχή κάτω δεξιά δείχνει μικρογραφίες όλων των αντικειμένων στο πρόγραμμα, με το τρέχον επιλεγμένο αντικείμενο υπογραμμισμένο.

- ✓ Η παλέτα εντολών είναι πάντα ορατή, ώστε να ενθαρρύνει την καταχώριση κώδικα. Οι εντολές χωρίζονται σε οχτώ κατηγορίες, όπως Κίνηση, Εμφάνιση, Ήχος, και Έλεγχος. Μ' αυτόν τον τρόπο αποφεύγονται οι μακροσκελείς, πιθανώς αχανείς λίστες εντολών: στις περισσότερες παλέτες, όλες οι εντολές είναι ορατές χωρίς την κύλιση του δρομέα (scrolling). Σε κάθε κατηγορία, οι πιο αυτονόητες και χρήσιμες εντολές εμφανίζονται κοντά στην κορυφή τις παλέτας εντολών. Τα μπλοκ εντολών είναι κωδικοποιημένα με χρώμα ανά κατηγορία, βοηθώντας τους χρήστες να βρουν τα σχετικά μπλοκ *(Πηγή: http://web.media.mit.edu/~jmaloney/papers/ScratchLangAndEnvironment.pdf).*



Εικόνα 3. Ο μεταγλωττιστής Scratch 2.0 σε λειτουργία online (https://scratch.mit.edu/)

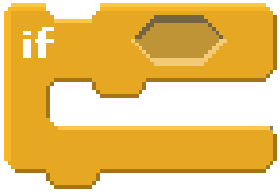
Το Scratch ενημερώνει τη διάταξη μετά από κάθε εντολή. Το να βλέπει κανείς το αποτέλεσμα κάθε εντολής στην οθόνη, ακόμα και στιγμιαία, παρέχει σημαντικές οπτικές ενδείξεις για την αντιμετώπιση προβλημάτων.
Το Scratch 1.0 είχε 92 μπλοκ εντολών. Καθώς το Scratch εξελίσσεται, υπάρχει μία συνεχής προσπάθεια να περιοριστεί ο αριθμός των εντολών.
Περισσότερες εντολές έχουν προστεθεί απ' ό,τι αφαιρεθεί. Το Scratch 1.4 έχει 125 μπλοκ εντολών, αν και κάποια απ' αυτά δεν εμφανίζονται μέχρι να χρειαστούν.
Αν ο υπολογιστής σας είναι παλιός ή δεν μπορείτε να εγκαταστήσετε τον εκτός σύνδεσης μεταγλωττιστή γλώσσας προγραμματισμού Scratch 2.0, μπορείτε να προσπαθήσετε να εγκαταστήσετε το Scratch 1.4.
Αν είστε διαχειριστής δικτύου: ένα Scratch 2.0 MSI έχει δημιουργηθεί και διατηρηθεί από ένα μέλος της κοινότητας και παρέχεται για λήψη στο http://llk.github.io/scratch-msi/.
Ένα από τα σφάλματα (bug) στο Scratch 2.0 ήταν η ικανότητα των «Scratcher» να ακολουθούν χρήστες, τους οποίους δε θα μπορούσαν κανονικά ν' ακολουθήσουν, όπως τους εαυτούς τους ή διαγραμμένους χρήστες.



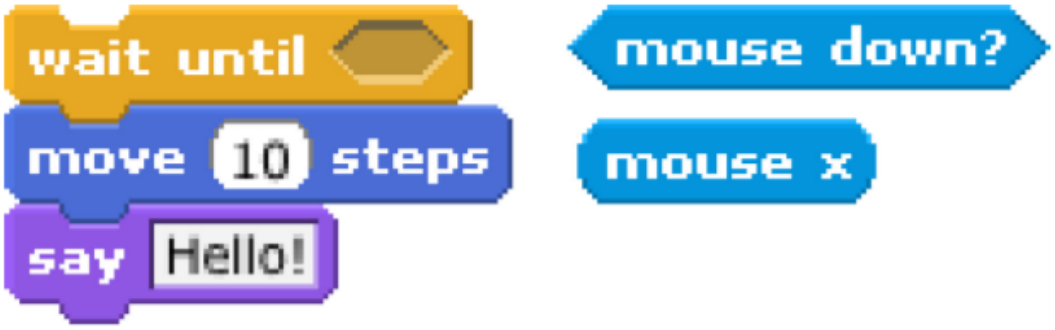
Εικόνα 4. Παραδείγματα των Νέων Χαρακτηριστικών και των Μπλοκ

Το Scratch επιτρέπει στα μπλοκ να συνδεθούν μόνο με ουσιαστικούς τρόπους. Ένα μπλοκ εντολής συνδέεται όταν το τοποθετούμε σε μια ακολουθία εντολών, αλλά ένα μπλοκ λειτουργίας δε θα συνδεθεί αν το τοποθετήσουμε στο ίδιο σημείο. Καθώς το χρήστης «σύρει» ένα μπλοκ, το Scratch δίνει οπτικά σχόλια που υποδεικνύουν πιθανά σημεία εισαγωγής στην ακολουθία (μπλοκ εντολών) ή στόχους υποδοχής παραμέτρων (μπλοκ λειτουργίας) (Maloney, 2010).

Πίνακας 1. Τύποι Μπλοκ του Scratch (Maloney, 2010)

	Ένα μπλοκ *εντολής (command)* που έχει μία εγκοπή στην πάνω πλευρά και ένα ταιριαστό εξόγκωμα στην κάτω. Τα μπλοκ εντολών μπορούν να ενωθούν, ώστε να δημιουργήσουν μια ακολουθία εντολών που ονομάζεται *στοίβα (stack).*
	Ένα μπλοκ *λειτουργίας (function)* επιστρέφει μια τιμή. Τα μπλοκ λειτουργίας δεν έχουν εγκοπές.
	Ένα μπλοκ *πυροδότησης (trigger)* έχει καμπυλωτή κορυφή. Εκτελεί το σενάριο που ακολουθεί όταν λαμβάνει χώρα το συμβάν που πυροδοτεί.
	Τα μπλοκ εντολών *ελέγχου δομής (control structure)* έχουν εσοχές για να υποστηρίζουν το «φώλιασμα» των ακολουθιών εντολών.

Στο Scratch, ένα μπλοκ ελέγχου δομής είναι μια αδιαίρετη μονάδα. Ο βραχίονας που κλείνει το βρόχο (λούπα) ή ένα μπλοκ προϋποθέσεων είναι μέρος του μπλοκ αυτού καθ' αυτού —δεν μπορεί να είναι εσφαλμένο— και το «φώλιασμα» της εσώκλειστης ακολουθίας εντολών είναι έκδηλο.

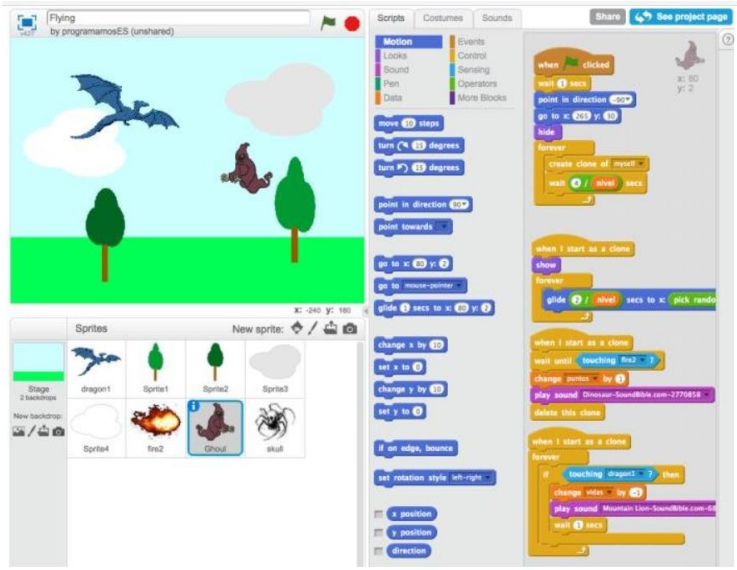


Εικόνα 5. Τα σχήματα υποδεικνύουν τον τύπο. Στα αριστερά, μπλοκ εντολών με υποδοχές παραμέτρων για δυαδικές (Boolean), αριθμητικές και αλφαριθμητικές παραμέτρους. Στα δεξιά, δυαδικά (Boolean) και αριθμητικά μπλοκ λειτουργίας (Maloney, 2010).

3.2.2 Πρακτική εφαρμογή των προγραμμάτων Scratch

Η γλώσσα των μπλοκ του Scratch εξαλείφει τα λάθη συντακτικού, επιτρέποντας έτσι στους χρήστες να εστιάσουν σε ενδιαφέροντα προβλήματα αμέσως. Τα μηνύματα σφαλμάτων χρόνου εκτέλεσης αποφεύγονται μέσω εντολών μετάπτωσης λειτουργίας, ενώ ένα προσεκτικά σχεδιασμένο μοντέλο συγχρονισμού αποφεύγει τις καταστάσεις αγώνα.
Για καθεμία εκπαιδευτική δραστηριότητα, μπορείτε να προσπαθήσετε να παρακολουθήσετε το Tutorial, να «κατεβάσετε» ένα σετ Καρτών Κωδικοποίησης ή να συμβουλευτείτε τους Οδηγούς Εκπαιδευτικών.

Η γλώσσα προγραμματισμού Scratch δίνει έμφαση στην απλότητα. Το σύστημα τύπων και το μοντέλο αντικειμένων σχεδιάστηκαν ώστε να λειτουργούν ομαλά, χωρίς να χρειάζονται εξηγήσεις προηγουμένως, όμως να έχουν απόλυτο νόημα με μια πιο προσεκτική εξέταση.



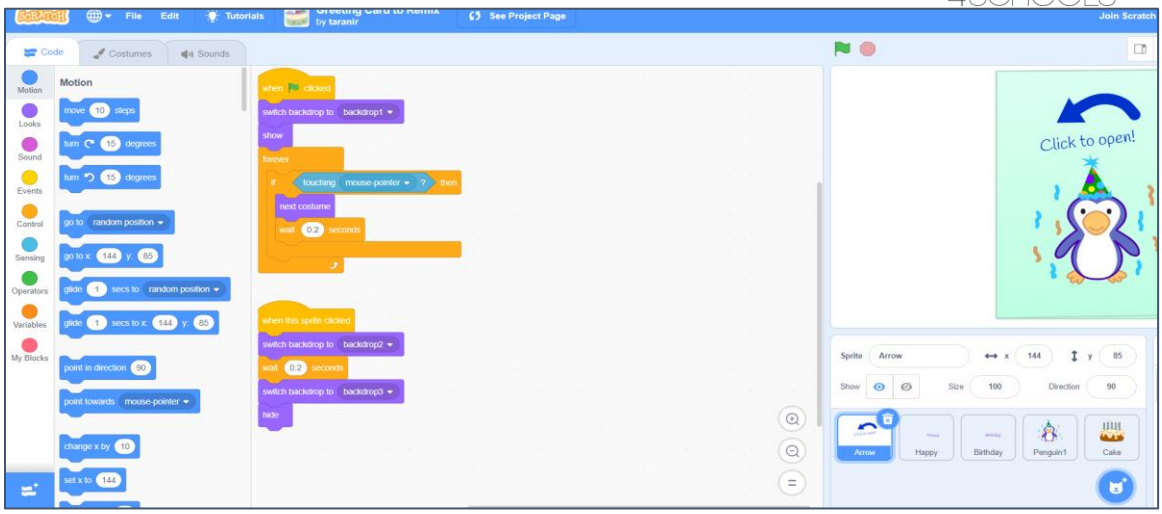
Εικόνα 6. Παράδειγμα στιγμιοτύπου οθόνης ενός έργου στο Scratch. Στα δεξιά, τα οπτικά στοιχεία που χρησιμοποιούνται για τον προγραμματισμό στο περιβάλλον του Scratch. (*ComputerProgrammingInTheEnglishClassroom.pdf)*

Η δομή του περιβάλλοντος χρήστη του Scratch μας διευκολύνει να εξερευνήσουμε και να «παίξουμε» με ιδέες. Για να δημιουργήσουμε ένα παιχνίδι, μια ιστορία ή κινούμενα σχέδια στο Scratch, βάζουμε τα μπλοκ σε στοίβες ώστε να σχηματίσουμε ένα σενάριο, το οποίο δίνει οδηγίες στα αντικείμενα του έργου.

Καθώς δημιουργούμε έργα, αξιολογούμε τη δουλειά μας και διαπιστώνουμε αν τα αποτελέσματα ανταποκρίνονται στις προσδοκίες μας. Αυτό είναι πολύ εύκολο, καθώς όλα συμβαίνουν μέσα σ' ένα περιβάλλον χρήστη.

Δραστηριότητα – Ευχετήρια Κάρτα (https://scratch.mit.edu/projects/11739928)
Οι εκπαιδευτές μπορούν να διασκευάσουν ιδέες:
- Επεξεργασία της κάρτας για διαφορετική περίσταση
- Αλλαγή των εικόνων ώστε να ταιριάζουν με το θέμα
- Αναπαραγωγή κινουμένων σχεδίων μέσα στην κάρτα

Οι μαθητές μπορούν να ενσωματώσουν τις φωνές και τις εικόνες τους μέσα σ' ένα έργο, δημιουργώντας έτσι κάτι που βοηθάει άλλος να μάθουν για τους εαυτούς τους και τους ανθρώπους, για ζητήματα και πράγματα που τους απασχολούν. Με τις Κάρτες Κωδικοποίησης του Scratch, οι μαθητές μπορούν να μάθουν να δημιουργούν διαδραστικά παιχνίδια, ιστορίες, μουσικές, κινούμενα σχέδια και πολλά ακόμα!
Ο παίχτης θα πρέπει να έχει ένα σενάριο, το οποίο ελέγχει τις κινήσεις του χρησιμοποιώντας τον κέρσορα του ποντικιού. Επίσης, ο στόχος θα πρέπει να προγραμματιστεί να εμφανίζεται σε τυχαία σημεία. Αν ο σχεδιασμός του παιχνιδιού σας είναι διαφορετικός, ίσως να χρειάζεται να προσαρμόσετε το σύστημα τιμών ώστε αυτό να ταιριάζει με το έργο σας στο Scratch.



Εικόνα 7. Παράδειγμα στιγμιοτύπου οθόνης ενός έργου στο Scratch (https://scratch.mit.edu/projects/11739928/editor/)

Ο κώδικας που θα κατασκευάσετε απαιτεί τη χρήση μιας *μεταβλητής*. Μια μεταβλητή είναι ένας παράγοντας που μπορεί ν' αλλάξει. Μπορείτε να δημιουργήσετε μια μεταβλητή βαθμολογίας, η οποία θα χρησιμοποιηθεί για να διατηρεί τον αριθμό των πόντων.

Ένα μπλοκ κωδικοποίησης θα χρησιμοποιηθεί για να αυξήσει τη *βαθμολογία (score)* μέσω ενός προσδιορισμένου αριθμού πόντων. 

Ένα άλλο μπλοκ κωδικοποίησης θα επαναφέρει τη μεταβλητή *βαθμολογίας* στο μηδέν, όταν ξεκινήσει ένα καινούριο παιχνίδι. 

Το περιβάλλον προγραμματισμού και η γλώσσα του Scratch λειτουργούν μαζί, ώστε να δημιουργήσουν ένα σύστημα εξαιρετικά γρήγορο να το μάθει κανείς, κινώντας το ενδιαφέρον των χρηστών για χρόνια˙ οι χρήστες μπορούν να προγραμματίζουν μέσα σε 15 λεπτά.

3.2.3 Συμπεράσματα

Το Scratch έχει ένα μικρό αριθμό εντολών, επιτρέπει στα αντικείμενα να ανταλλαχθούν χωρίς να διασπαστούν οι εξαρτήσεις, ενθαρρύνοντας έτσι τη συνεργασία και το μοίρασμα του κώδικα. Το σύστημα είναι πάντα ζωντανό, χωρίς διακόπτη εκτέλεσης/επεξεργασίας, ώστε οι εντολές ή τα αποσπάσματα κώδικα να μπορούν να εκτελεστούν με ένα κλικ, και τα παραστατικά σχόλια δείχνουν την εκτέλεση. Οι μεταβλητές και οι λίστες έχουν σαφείς απεικονίσεις, οπότε τα αποτελέσματα των χειρισμών των δεδομένων φανερώνονται αμέσως (Maloney, 2010).

Η ικανότητα γραψίματος κώδικα για προγράμματα υπολογιστών είναι ένα σημαντικό κομμάτι αλφαβητισμού στη σημερινή κοινωνία. Όταν κάποιος μαθαίνει να γράφει κώδικα στο Scratch, τότε μαθαίνει σημαντικές στρατηγικές για την επίλυση προβλημάτων, το σχεδιασμό έργων και την έκφραση ιδεών.

Τα Έργα Scratch – βασισμένα σε εκπαιδευτικές ενότητες - θα περιλαμβάνουν διαφορετικές ασκήσεις, και αυτή η νέα προοπτική θα επιτρέψει στους μαθητές να εφαρμόσουν όσα μαθαίνουν σε νέες καταστάσεις, με αποτέλεσμα τη βαθύτερη μάθηση: οι μαθητές θα ασχοληθούν με δραστηριότητες σχεδίου, ακολουθώντας τα προσωπικά τους ενδιαφέροντα, αλληλεπιδρώντας μέσω δημιουργικών συνεργασιών, συλλογιζόμενοι τις χρήσιμες εμπειρίες.

3.3 Τεστ αξιολόγησης

1. Το Scratch δεν είναι μόνο για το μάθημα της πληροφορικής. Το Scratch μπορεί να ενσωματωθεί σε οποιοδήποτε περιεχόμενο σε όλες τις τάξεις.
   1. **Ναι**
   2. Όχι

2. Το Scratch είναι διαθέσιμο και σε σύνδεση (online) και ως αρχείο που μπορεί να το κατεβάσει κανείς.
   1. **Ναι**
   2. Όχι

3. Βάζοντας τους μαθητές να δημιουργήσουν ένα παιχνίδι μαθηματικών με το πρόγραμμα Scratch, μπορούν να εφαρμόσουν όσα έμαθαν στα μαθήματα σε μια πρακτική άσκηση.
   1. **Ναι**
   2. Όχι

4. Οι μαθητές μπορούν να ενσωματώσουν τις φωνές και τις εικόνες τους σε ένα έργο στο Scratch.
   1. **Ναι**
   2. Όχι

5. Μπορείτε να κρατήσετε τη βαθμολογία (score) στο Scratch δημιουργώντας...
   1. Ένα αντικείμενο
   2. Ένα βρόχο (λούπα)
   3. **Μια μεταβλητή**
   4. Μια λειτουργία

6. Αρκετές δηλώσεις ενωμένες μαζί αποτελούν ...
   1. Μια εκτέλεση
   2. **Μια ακολουθία**
   3. Ένα βρόχο
   4. Μια μεταβλητή

7. Πώς δημιουργείτε ένα βρόχο (λούπα) στο Scratch?
   1. **Χρησιμοποιώντας ένα μπλοκ επανάληψης**
   2. «Σπάζοντας» ένα μπλοκ
   3. Χρησιμοποιώντας ένα μπλοκ προϋποθέσεων
   4. Χρησιμοποιώντας μια μεταβλητή

8. Σωστό ή Λάθος: μπορείτε να προσθέσετε τις δικές σας εικόνες στα έργα σας στο Scratch
   1. **Σωστό**
   2. Λάθος

9. Η εκτέλεση μιας ενέργειας ανάλογα με το ΑΝ πληρείται ένα κριτήριο ονομάζεται...
    1. ακολουθία
    2. δήλωση
    3. βρόχος (λούπα)
    4. **προϋπόθεση**

10. Η επανάληψη μιας δήλωσης παραπάνω από μια φορά ονομάζεται...
    1. **Βρόχος (λούπα)**
    2. Επανάληψη
    3. Γεγονός
    4. Ακολουθία

3.4 Παραπομπές

- KAY, A. 2010. Squeak etoys, children, and learning. http://www.squeak-land.org/resources/articles
- Resnick, M., Maloney, j., Monroy-Hernandez, 2009. Scratch: Programming for all. Comm. ACM 52, 11, 60–67.
- ComputerProgrammingInTheEnglishClassroom.pdf
- Maloney, J., Resnick, M., Rusk, N., Silverman, B., and Eastmond, E. 2010. *The scratch programming language and environment*. ACM Trans. Comput. Educ. 10, 4, Article 16 (November 2010), 15 pages. DOI = 10.1145/1868358.1868363. http://doi.acm.org/10.1145/1868358.1868363
- https://education.abc.net.au/home#!/media/1214681/intro-to-scratch-20
- https://scratch.mit.edu
- http://web.media.mit.edu/~jmaloney/papers/ScratchLangAndEnvironment.pdf
- https://www.thomasbuxton.towerhamlets.sch.uk/blogs/year3/2017/11/17/year-3-computing-scratch-projects/

4 Scratch GPIO (Έλεγχος ακίδων GPIO/λήψη καταχωρήσεων) [P3-DANMAR]

4.1 Λεξικό όρων

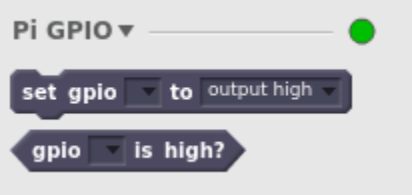
Όρος / Έννοια	Ορισμός / Εξήγηση
Επέκταση Pi GPIO	Η επέκταση Pi GPIO είναι ένα πρόσθετο στοιχείο στο Scratch, που επιτρέπει την αλληλεπίδραση με τις ακίδες Raspberry GPIO. Αυτή η επέκταση παρέχει δύο επιπλέον μπλοκ που «διαβάζουν» και καθορίζουν την κατάσταση όλων των 28 ακίδων GPIO που παρέχονται από το Raspberry.
Raspbian	Το Raspbian είναι ένα λειτουργικό σύστημα που βασίζεται στο Debian και έχει βελτιστοποιηθεί, ώστε να χρησιμοποιείται στις συσκευές Raspberry. Παρέχει ένα απλό περιβάλλον χρήστη GUI και συνοδεύεται από την εγκατάσταση του Scratch και άλλα εργαλεία προγραμματισμού, που επιτρέπουν στους χρήστες να αλληλεπιδρούν με το λειτουργικό (hardware) του Raspberry.

4.2 Κυρίως περιεχόμενο

Το Scratch συχνά θεωρείται ως εισαγωγή στον κόσμο του προγραμματισμού. Απλοί αλγόριθμοι μπορούν να αναπτυχθούν με τη χρήση αυτού του λογισμικού, ενώ το επίπεδο παρουσίασης, αποτελούμενο από πολλά οπτικά ερεθίσματα, είναι ελκυστικό. Παρ' όλα αυτά, το Scratch μπορεί να φαίνεται κάπως περιορισμένο όσον αφορά πιο προχωρημένες κι ενδιαφέρουσες εφαρμογές. Ευτυχώς, η υποστήριξη GPIO που μπορεί να ενεργοποιηθεί, επιτρέπει τη χρήση των ακίδων Raspberry για τον έλεγχο πολλών διαφορετικών φυσικών κυκλωμάτων, μηχανισμών και συσκευών.

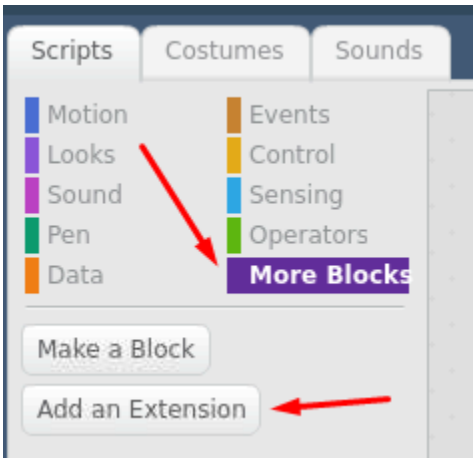
4.2.1 Βασικές πληροφορίες για το Scratch GPIO

Το περιβάλλον του Scratch είναι τέλειο για όσους ξεκινούν με προγραμματισμό μικροελεγκτών (microcontrollers). Με τη χρήση του Scratch, είναι δυνατό όχι μόνο να σχεδιάσει κανείς συγκεκριμένους αλγορίθμους και να τους «τρέξει», αλλά και να ελέγχει τις θύρες GPIO του Raspberry, ώστε να βελτιώσει τη λειτουργικότητα ολόκληρου του συστήματος. Το Scratch υποστηρίζει τη χρήση του GPIO, κάτι που μπορεί να επιτευχθεί με την πρόσθεση μιας ειδικής επέκτασης Pi GPIO.
Αφού η επέκταση GPIO προστεθεί στο Scratch, δύο νέα μπλοκ είναι διαθέσιμα για περαιτέρω χρήση. Αυτά είναι: τα μπλοκ στοίβας (stack block) και τα δυαδικά (Boolean) μπλοκ. Το μπλοκ στοίβας είναι μια απλή εντολή που μπορεί να χρησιμοποιηθεί για να ρυθμίσει οποιαδήποτε θύρα GPIO είτε σε υψηλή είτε χαμηλή κατάσταση. Το δυαδικά μπλοκ απ' την άλλη, μπορεί να χρησιμοποιηθεί για να ελέγξει αν οποιαδήποτε ακίδα GPIO είναι σε υψηλή ή χαμηλή κατάσταση. Αυτό επιτρέπει στο Scratch να ελέγχει όλες τις διαθέσιμες ακίδες GPIO στην πλακέτα Raspberry.



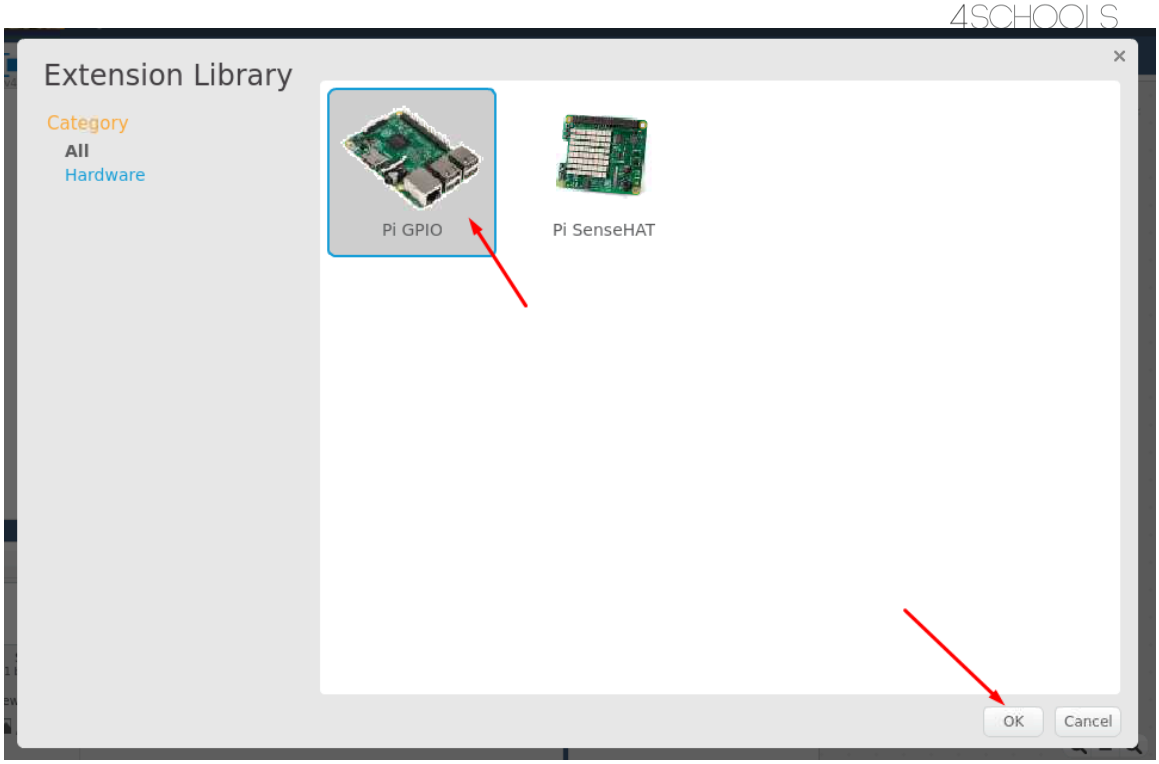
ΕΙΚΟΝΑ 1. ΕΠΙΠΡΟΣΘΕΤΑ ΜΠΛΟΚ ΠΟΥ ΠΑΡΕΧΟΝΤΑΙ ΑΠΟ ΤΗΝ ΕΠΕΚΤΑΣΗ GPIO

Γενικά, η επέκταση GPIO για το Scratch «ανοίγει» ένα ευρύ φάσμα νέων δυνατοτήτων, καθώς ο χρήστης δεν περιορίζεται πλέον στο σχεδιασμό αλγορίθμων μέσα στο λογισμικό του Scratch, αλλά, αντίθετα, είναι ικανός να αλληλεπιδρά με φυσικές συσκευές, όπως διόδους, βομβητές (buzzer) κ.λπ. Τα μπλοκ που παρέχονται από την επέκταση είναι πλήρως συμβατά με τα κανονικά (προ-εγκατεστημένα) μπλοκ του Scratch, επεκτείνοντας έτσι το ευρύ φάσμα των επιλογών έτι περαιτέρω.
Για να προσθέσετε την επέκταση GPIO στο Scratch, χρειάζεται να ανοίξετε την εφαρμογή και να επιλέξετε την τελευταία επιλογή -More Blocks- από την καρτέλα των Σεναρίων (Scripts). Αφού κάνετε κλικ στο «More Blocks», έπειτα χρειάζεται να κάνετε κλικ στο «Προσθέστε μια Επέκταση» (Add an Extension).



ΕΙΚΟΝΑ 2. ΕΝΕΡΓΟΠΟΙΗΣΗ ΤΗΣ ΕΠΕΚΤΑΣΗΣ GPIO

Όταν εμφανιστεί στην οθόνη η Βιβλιοθήκη Επεκτάσεων (Extension Library), πρέπει να επιλέξετε την επέκταση Pi GPIO και η επιλογή αυτή πρέπει να επιβεβαιωθεί (κάνοντας κλικ στο OK).

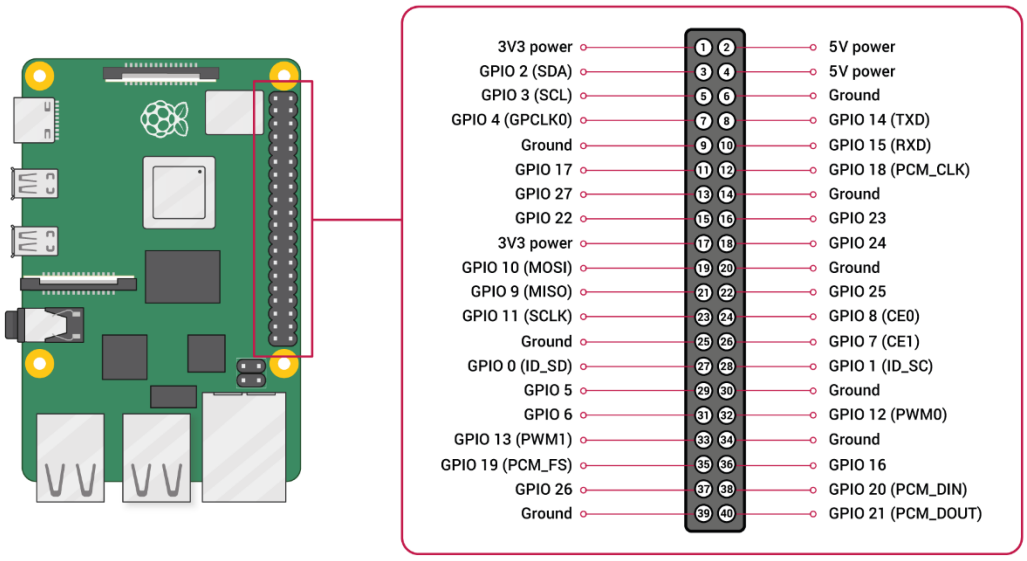


ΕΙΚΟΝΑ 3. ΕΝΕΡΓΟΠΟΙΗΣΗ ΤΗΣ ΕΠΕΚΤΑΣΗΣ GPIO

Αφού ολοκληρώσετε τα παραπάνω βήματα, το Scratch θα είναι έτοιμο να ελέγξει το GPIO του Raspberry.
Για να εξασφαλίσετε μια απρόσκοπτη εμπειρία: ο καλύτερος τρόπος για να βάλετε το περιβάλλον σε πλήρη λειτουργία, είναι να εγκαταστήσετε ένα λειτουργικό σύστημα Raspbian στο Raspberry, καθώς αυτή η κατανομή έχει τα πάντα ήδη εγκατεστημένα. Διαφορετικές εκδόσεις του Scratch είναι διαθέσιμες, ώστε οι χρήστες να μπορούν να επιλέξουν εκείνη με την οποία είναι πιο εξοικειωμένοι.

4.2.2 Κανόνες ονομασίας GPIO

Είναι σημαντικό ν' αναφερθεί ότι οι αριθμοί των ακίδων GPIO που χρησιμοποιούνται στο Scratch, ακολουθούν τους κανόνες ονομασίας που χρησιμοποιούνται απ' το ίδιο το Raspberry. Αυτό μπορεί να είναι περίπλοκο, ειδικά για τους αρχάριους, καθώς το Raspberry, στα περισσότερα μοντέλα, χρησιμοποιεί ένα βύσμα 40 ακίδων και οι ακίδες είναι αριθμημένες από το 1 έως το 40. Για να αποφευχθεί η σύγχυση, είναι καλύτερα να έχετε ένα επονομαζόμενο διάγραμμα ακίδων (pinout chart) για το Raspberry. Ένα παράδειγμα, διαθέσιμο στα επίσημα εγχειρίδια του Raspberry, παρουσιάζεται παρακάτω:



ΕΙΚΟΝΑ 4. ΤΟ ΔΙΑΓΡΑΜΜΑ ΑΚΙΔΩΝ RASPBERRY (PINOUT CHART)

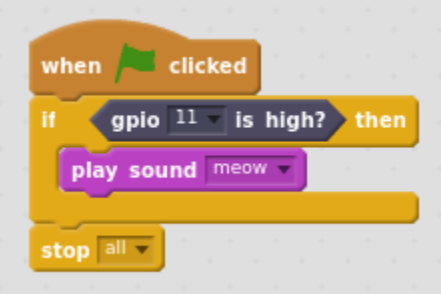
Ως εκ τούτου, αν ο κώδικας στο Scratch χρησιμοποιεί τον αριθμό 26, αυτό υποδηλώνει τη φυσική ακίδα 37 (αριστερή στήλη, δεύτερη από το τέλος). Είναι σημαντικό να εξοικειωθείτε με τη θέση των συγκεκριμένων ακίδων GPIO, ώστε να διασφαλίσετε ότι δεν υπάρχει σύγχυση μεταξύ των αριθμών που υποδηλώνουν τις φυσικές ακίδες κι εκείνων που υποδηλώνουν τις ακίδες GPIO.

4.2.3 Αλληλεπίδραση GPIO

Όπως προαναφέρθηκε, τα μπλοκ GPIO είναι πλήρως συμβατά με τα μπλοκ που παρέχονται απ' το Scratch. Ως εκ τούτου, το απλούστερο παράδειγμα για την επίδειξη του πώς συγκεκριμένες ακίδες μπορούν να ελεγχθούν από το Scratch, μια προσομοίωση ανάμματος μιας λυχνίας LED μπορεί να πραγματοποιηθεί μ' έναν πολύ βασικό τρόπο. Το επακόλουθο προϋποθέτει ότι η λυχνία LED έχει συνδεθεί στην ακίδα GPIO 11.

ΕΙΚΟΝΑ 5. ΧΡΗΣΗ ΜΙΑΣ GPIO ΑΚΙΔΑΣ ΕΞΟΔΟΥ ΣΤΟ SCRATCH

Το παραπάνω παράδειγμα χρησιμοποιεί ένα μπλοκ στοίβας, που παρέχεται από την επέκταση του GPIO. Το δυαδικό μπλοκ μπορεί επίσης να χρησιμοποιηθεί, για παράδειγμα, για να επιδράσει στην ανίχνευση μιας υψηλής κατάστασης σε οποιαδήποτε ακίδα GPIO.



ΕΙΚΟΝΑ 6. ΧΡΗΣΗ ΜΙΑΣ ΑΚΙΔΑΣ ΕΙΣΑΓΩΓΗΣ GPIO ΣΤΟ SCRATCH

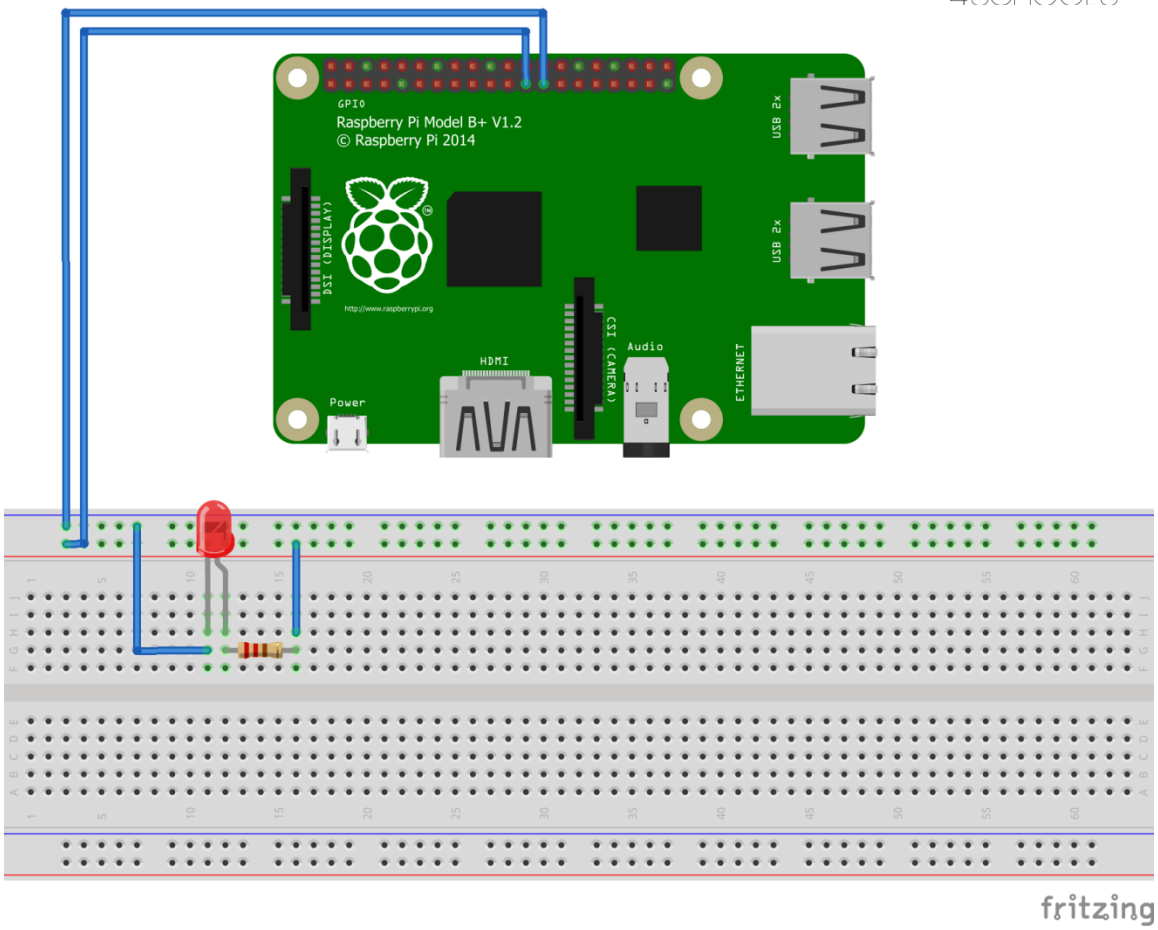
Μπορεί να φαίνεται ότι η ύπαρξη μόνο δύο επιπλέον μπλοκ είναι κάπως περιοριστική. Στην πραγματικότητα, ωστόσο, ο μικρο-ελεγκτικός προγραμματισμός έχει να κάνει με δύο καταστάσεις: υψηλή και χαμηλή. Επιπλέον, όλες οι σύγχρονες ψηφιακές συσκευές λειτουργούν με αυτόν τον τρόπο. Οι υψηλές και χαμηλές καταστάσεις μπορούν να μεταφραστούν σε λογικές τιμές του 0 και του 1 και, στην πραγματικότητα, αυτό σημαίνει ότι δεν υπάρχει τάση (0V) ή ότι υπάρχει τάση (3.3V), αντίστοιχα. Για το Raspberry, το GPIO λειτουργεί στα 3.3V, το οποίο καλό θα είναι να έχετε στο νου σας, για τη σύνδεση διαφόρων συσκευών, αισθητήρων και εξοπλισμού σε αυτές τις ακίδες.
Η επέκταση GPIO μπορεί επίσης να αντικαταστήσει υπάρχοντα έργα στο Scratch. Αντί να αλληλεπιδρούν με το χρήστη μόνο με τη χρήση της οθόνης του Scratch, κάποιες λειτουργίες μπορούν να μεταφερθούν στις φυσικές ακίδες GPIO. Για παράδειγμα, αντί για την προβολή ενός μηνύματος ή το παίξιμο ενός ήχου, η επέκταση GPIO μπορεί να χρησιμοποιηθεί για να εκπέμψει τον ήχο, να ανάψει τα LED, κ.ο.κ. Οι δυνατότητες αυτής της εγκατάστασης περιορίζονται μόνο από τη φαντασία και τη δημιουργικότητα του ατόμου που δουλεύει σε ένα συγκεκριμένο έργο, χρησιμοποιώντας το Scratch με την υποστήριξη του GPIO του Raspberry.

4.3 Πρακτικές εφαρμογές

Στις επόμενες σελίδες παρέχονται κάποια πρακτικά παραδείγματα. Αυτά τα παραδείγματα είναι σχετικά εύκολο να δημιουργηθούν και ο βασικός τους σκοπός είναι να αποδείξουν πώς μπορεί να χρησιμοποιηθεί το Scratch για τον έλεγχο των ακίδων GPIO στο Raspberry.

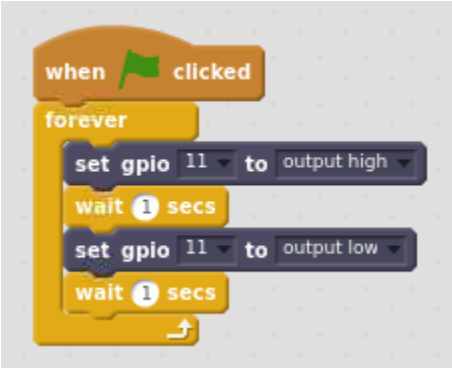
4.3.1 Λυχνία LED που αναβοσβήνει – παράδειγμα 1

Το πρώτο πρακτικό παράδειγμα θα παρουσιάσει τα βασικά του τρόπου λειτουργίας του ελεγχόμενου απ’ το Scratch GPIO. Η βασική ιδέα είναι ότι ένα πολύ απλό κύκλωμα θα χρησιμοποιηθεί για ν’ αναβοσβήσει τη λυχνία LED. Το LED θα ελέγχεται από την ακίδα GPIO νούμερο 11 (φυσική ακίδα 23). Ενώ μπορεί να χρησιμοποιηθεί οποιοδήποτε άλλο GPIO, η ακίδα GPIO 11 είναι πολύ βολική να χρησιμοποιηθεί, καθώς έχει γείωση (GND) δίπλα της (φυσική ακίδα 25).
Για να στηθεί αυτό το κύκλωμα χρειάζονται μια λυχνία LED, ένας αντιστάτης κατάλληλος για LED, μια πλακέτα δοκιμών (breadboard) και δύο καλώδια μεταγωγής.
Το Raspberry θα πρέπει να συνδεθεί με την πλακέτα δοκιμών όπως υποδεικνύεται παρακάτω.



ΕΙΚΟΝΑ 7. ΣΥΝΔΕΣΗ ΤΟΥ LED ΣΤΟ RASPBERRY

Για να κάνουμε αυτό το πολύ απλό κύκλωμα να δουλέψει, η επιλεγμένη ακίδα GPIO 11 θα πρέπει να ελέγχεται από το Scratch, το οποίο θα ρυθμίσει την κατάστασή της σε υψηλή, θα κάνει παύση για 1 δευτερόλεπτο, θα ρυθμίσει την κατάσταση σε χαμηλή και θα ξανακάνει παύση για 1 δευτερόλεπτο. Αυτές οι τέσσερις οδηγίες μπορούν να τοποθετηθούν σ' έναν αέναο βρόχο (forever loop), αλλά άλλοι τύποι βρόχων μπορούν επίσης να χρησιμοποιηθούν, όπως η επανάληψη (repeat). Ένα δείγμα του σετ των μπλοκ παρουσιάζεται παρακάτω ως πλαίσιο αναφοράς.

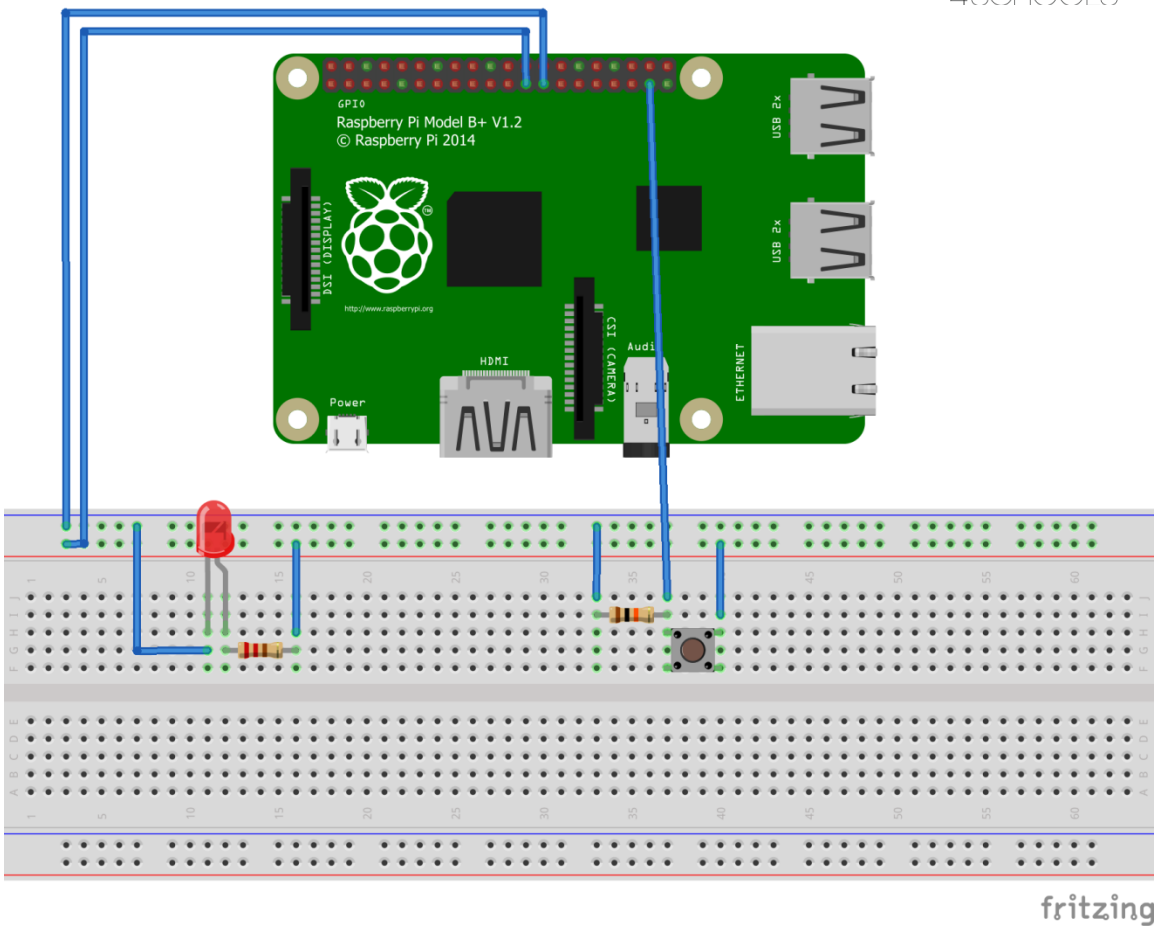


ΕΙΚΟΝΑ 8. ΕΝΑΛΛΑΓΗ ΤΗΣ ΚΑΤΑΣΤΑΣΗ ΤΗΣ ΑΚΙΔΑΣ GPIO ΣΤΟ SCRATCH

Αυτό που συμβαίνει εδώ είναι ότι το Scratch επικοινωνεί με το Raspberry, χρησιμοποιώντας την επέκταση GPIO, και δίνει εντολές που επιτρέπουν την αλλαγή της κατάστασης της ακίδας εξόδου (11) από υψηλή σε χαμηλή. Αυτό μεταφράζεται στην παρουσία τάσης στην ακίδα GPIO σε συγκεκριμένα χρονικά διαστήματα.
Αν κι αυτό το παράδειγμα είναι πολύ απλό, αποτελεί ένα σημείο εκκίνησης για τον έλεγχο του GPIO του Raspberry με τη χρήση του Scratch. Σ' αυτό το απλό κύκλωμα, η ακίδα GPIO 11 λειτουργεί ως ακίδα εξόδου (output), το οποίο σημαίνει ότι μόνο δέχεται οδηγίες για ν' αλλάξει την κατάστασή της κατάλληλα. Με την επέκταση GPIO είναι επίσης δυνατόν να «διαβάσετε» την κατάσταση της ακίδας GPIO, που στη συνέχεια θα λειτουργήσει ως ακίδα εισόδου (input). Αυτή η προσέγγιση παρουσιάζεται στο επόμενο παράδειγμα.

4.3.2 Έλεγχος των λυχνιών LED με έναν απτικό διακόπτη (tactile switch) – παράδειγμα 2

Για να παρουσιάσουμε πώς το Scratch μπορεί να «διαβάσει» την ακίδα GPIO του Raspberry ως είσοδο (input), μπορούμε να κατασκευάσουμε ένα απλό κύκλωμα. Εκτός απ' το παράδειγμα που παρουσιάστηκε παραπάνω, χρειαζόμαστε επιπλέον έναν απτικό διακόπτη, καθώς και έναν αντιστάτη 10k pull-down και επιπλέον καλώδια μεταγωγής.
Το κύκλωμα που χρησιμοποιήθηκε στο προηγούμενο παράδειγμα χρειάζεται να επεκταθεί με έναν απτικό διακόπτη, ο οποίος θα ρυθμίσει την ακίδα GPIO σε υψηλή κατάσταση όταν πατηθεί. Για να γίνει αυτό, μπορεί να χρησιμοποιηθεί η ακίδα GPIO νούμερο 26 (φυσική ακίδα 37).
Επειδή η αναμενόμενη συμπεριφορά είναι ότι η ακίδα GPIO 26 «διαβάζει» χαμηλή κατάσταση όταν ο διακόπτης αφής δεν είναι πατημένος, και υψηλή διαφορετικά, χρειάζεται να τοποθετήσουμε έναν pull-down αντιστάτη μεταξύ της γείωσης (GND) και της ακίδας GPIO 26. Χάρη σ' αυτό, κάθε φορά που δίνεται η εντολή να «διαβαστεί» η κατάσταση αυτής της ακίδας, θα «διαβάζει» χαμηλή (0V), όταν ο απτικός διακόπτης δεν είναι πατημένος. Αντίθετα, όταν ο απτικός διακόπτης είναι πατημένος, το «διάβασμα» πρέπει να είναι υψηλό. Αυτό μπορεί να επιτευχθεί συνδέοντας τον απτικό διακόπτη μεταξύ της ακίδας GPIO 26 και μιας γραμμής 3.3V, που παρέχεται από το Raspberry στις φυσικές ακίδες 1 και 17, οι οποίες είναι ήδη εφοδιασμένες με καλώδια μεταγωγής σε μια πλακέτα δοκιμών (breadboard). Το πάτημα του διακόπτη αφής θα κάνει το ρεύμα να περάσει από μια γραμμή 3.3V, θέτοντας έτσι την ακίδα GPIO 26 σε υψηλή κατάσταση. Ένα παράδειγμα κυκλώματος παρουσιάζεται παρακάτω ως πλαίσιο αναφοράς.



ΕΙΚΟΝΑ 9. Ο ΑΠΤΙΚΟΣ ΔΙΑΚΟΠΤΗΣ ΕΛΕΓΧΕΙ ΕΝΑ ΚΥΚΛΩΜΑ LED

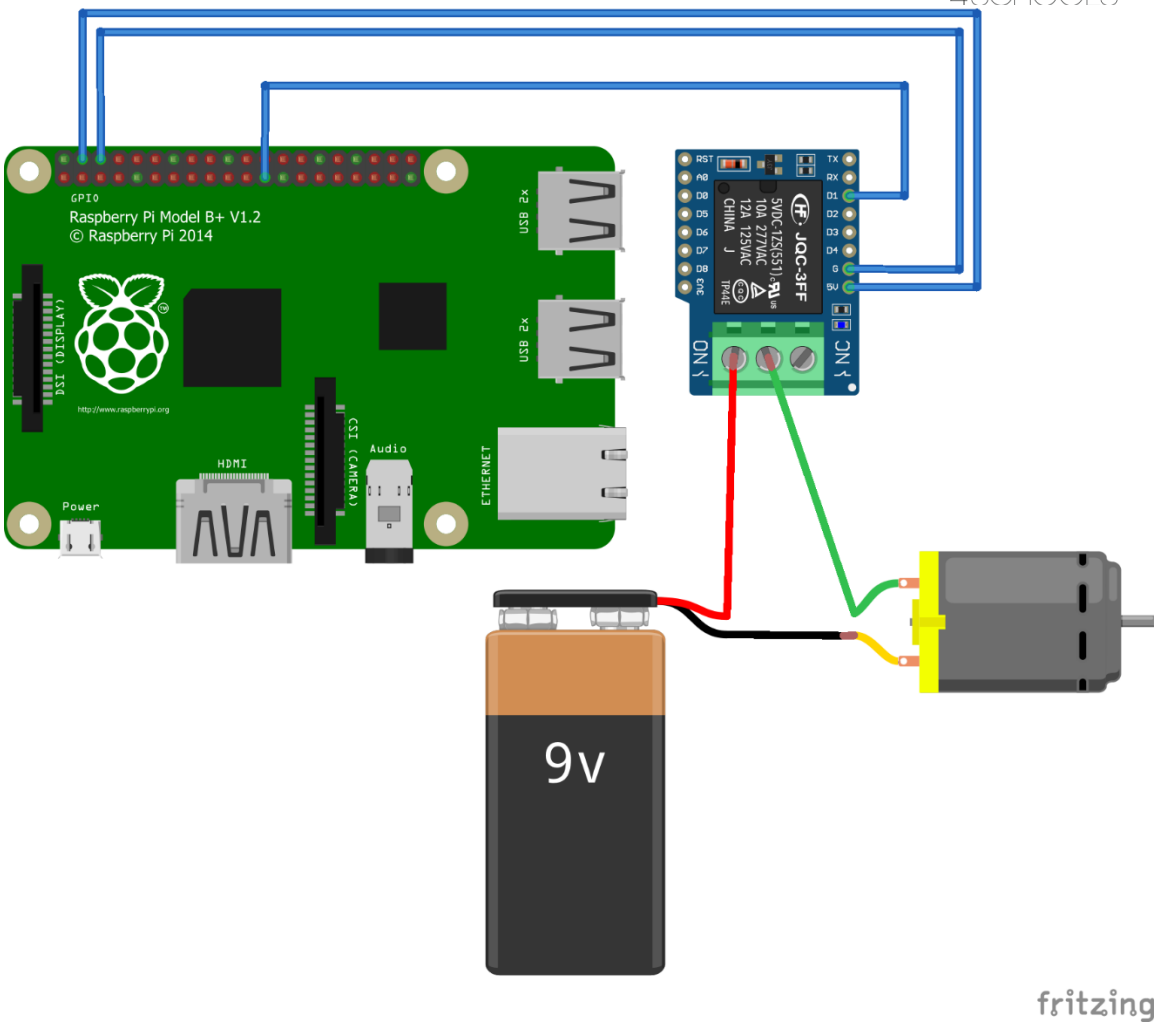
Τέλος, ο κώδικας του Scratch μπορεί να παρασχεθεί για να κάνει όλο το κύκλωμα να δουλέψει σύμφωνα με το πρόγραμμα. Για να γίνει αυτό, μπορεί να χρησιμοποιηθεί ένας αέναος βρόχος (forever loop). Μέσα σ' αυτό το βρόχο η ακίδα GPIO 26 πρέπει να «διαβαστεί». Σε όρους Scratch, ένα μπλοκ στοίβας μπορεί να χρησιμοποιηθεί ως παράμετρος στη δήλωση if (if-statement). Οι απομένουσες οδηγίες ακολουθούν την ήδη συνοπτική λογική του κυκλώματος.

ΕΙΚΟΝΑ 10. «ΔΙΑΒΑΣΜΑ» ΤΗΣ ΚΑΤΑΣΤΑΣΗΣ ΤΟΥ GPIO ΣΤΟ SCRATCH

Μόλις πατηθεί ο διακόπτης, η ακίδα GPIO 26 θα τεθεί σε υψηλή κατάσταση. Αν ανιχνευτεί μια τέτοια κατάσταση, η ακίδα GPIO 11, που χρησιμοποιήθηκε προηγουμένως, θα πρέπει να τεθεί στο υψηλό, ώστε να ανάψει η λυχνία LED. Η διαφορά μεταξύ των δύο ακίδων GPIO είναι ότι η ακίδα 26 θεωρείται ακίδα εισόδου (input) (και το Scratch «τσεκάρει» την κατάστασή της συνεχώς), ενώ η ακίδα 11 είναι μια ακίδα εξόδου (output), η οποία ελέγχει τη λυχνία LED ανάβοντάς τη (παρέχοντας τάση σε υψηλή κατάσταση) και σβήνοντάς τη (καμία τάση στη χαμηλή κατάσταση).

4.3.3 Επεκτείνοντας τη χρήση του GPIO – παράδειγμα 3

Η χρήση του Scratch για τον έλεγχο του GPIO του Raspberry μπορεί να μας οδηγήσει σε μια λανθασμένη εντύπωση ότι στη χειρότερη περίπτωση ο κώδικας δε θα δουλέψει. Στην πραγματικότητα όμως, η αλληλεπίδραση με το λειτουργικό του Raspberry χρειάζεται ιδιαίτερη προσοχή. Ένα παράδειγμα είναι ότι οι ακίδες GPIO δεν επιτρέπεται να «τραβήξουν» παραπάνω από 50mA ρεύματος. Ενώ αυτό μπορεί να είναι εντάξει όσον αφορά μια λυχνία LED ή ένα βομβητή (buzzer), οποιεσδήποτε άλλες «πεινασμένες για ρεύμα» πηγές δε θα δουλέψουν, και ίσως προκαλέσουν μόνιμη βλάβη στην πλακέτα Raspberry. Τέτοιες περιπτώσεις, ωστόσο, μπορούν να επιλυθούν με τη χρήση ενός ηλεκτρονόμου (ρελέ) και μιας εξωτερικής πηγής ενέργειας.
Ένας ηλεκτρονόμος τυπικά λειτουργεί στα 5V και αυτή η τάση παρέχεται από το Raspberry στις ακίδες 2 και 4. Την ίδια στιγμή, η ακίδα εξόδου GPIO του Raspberry ρυθμισμένη σε υψηλή κατάσταση είναι αρκετή για να πυροδοτήσει το ρελέ και να το θέσει σε κατάσταση λειτουργίας. Το εξωτερικό κύκλωμα που είναι συνδεδεμένο με το ρελέ θα κλείσει, και το ρεύμα θα ξεκινήσει να ρέει, τροφοδοτώντας κάθε συσκευή. Αυτό μπορεί να το φανταστεί κανείς στήνοντας ένα κύκλωμα, στο οποίο το Scratch ελέγχει το ρελέ με τη χρήση μιας ακίδας εξόδου GPIO, και το ρελέ στη συνέχεια τροφοδοτεί ένα μοτέρ που χρειάζεται (για παράδειγμα) 9V.
Ο ηλεκτρονόμος μπορεί να συνδεθεί στις ακίδες 4 (5V output) και 6 (GND). Ο έλεγχος/πλοήγηση του ρελέ μπορεί να συνδεθεί στην ακίδα GPIO 11, που θα χρησιμοποιηθεί ως ακίδα εξόδου. Στη συνέχεια, το εξωτερικό κύκλωμα (ένα μοτέρ 9V με τη δική του πηγή ρεύματος) πρέπει να καλωδιωθεί με τον ηλεκτρονόμο. Ένα παράδειγμα παρέχεται παρακάτω ως πλαίσιο αναφοράς.



ΕΙΚΟΝΑ 11. ΧΡΗΣΗ ΕΝΟΣ ΗΛΕΚΤΡΟΝΟΜΟΥ ΓΙΑ ΤΗΝ ΕΚΚΙΝΗΣΗ ΤΟΥ ΜΟΤΕΡ

Μέσα στο Scratch, η οδηγία που θα εκκινήσει τον ηλεκτρονόμο είναι ένα μπλοκ στοίβας, το οποίο θα θέσει την ακίδα GPIO 11 σε υψηλή κατάσταση. Αυτό μπορεί να προξενηθεί χρησιμοποιώντας ένα αντικείμενο που αναπαριστά τη γάτα. Μια ακόμη βελτίωση σε σχέση με τα προηγούμενα παραδείγματα είναι ότι θα υπάρχει μια καταχώρηση που θα παρέχεται απ' το from Scratch, η οποία θα δρα διαφορετικά ανάλογα με την τρέχουσα κατάσταση («διάβασμα») της ακίδας GPIO 11.
Γι' αυτόν το σκοπό, μπορεί να υπάρξει μια ενέργεια που θα εκτελείται κάθε φορά που θα κάνετε κλικ στο χαρακτήρα της γάτας. Αν η ακίδα εξόδου GPIO «διαβάζεται» ως υψηλή, θα τεθεί σε χαμηλή. Αν είναι χαμηλή, θα τεθεί σε υψηλή. Η ακίδα GPIO θα παραμείνει στη ρυθμισμένη κατάσταση, ενεργώντας έτσι ως ένας απλός, διφασικός διακόπτης. Επιπλέον, η γάτα θα πει «Γεια!», ώστε να δώσει επιπρόσθετη οπτική ανατροφοδότηση ότι το μπλοκ του κώδικα πράγματι εκτελέστηκε. Ένα παράδειγμα κώδικα στο Scratch παρέχεται παρακάτω ως πλαίσιο αναφοράς.

ΕΙΚΟΝΑ 11. ΤΟ ΑΝΤΙΚΕΙΜΕΝΟ ΔΡΑ ΩΣ ΔΙΦΑΣΙΚΟΣ ΔΙΑΚΟΠΤΗΣ «ΔΙΑΒΑΖΟΝΤΑΣ» ΤΗΝ ΙΔΙΑ ΤΟΥ ΤΗΝ ΚΑΤΑΣΤΑΣΗ

Σ' αυτό το παράδειγμα αποδείχθηκε ότι οι θύρες GPIO του Raspberry μπορούν να χρησιμοποιηθούν με έναν αρκετά ευέλικτο τρόπο. Αν και η ακίδα GPIO 11 χρησιμοποιείται κυρίως σαν οδηγός ακίδας εξόδου (ανοιγοκλείνοντας τον ηλεκτρονόμο), παράλληλα είναι δυνατό να κατασκευαστεί η λογική του κυκλώματος γύρω της, ελέγχοντας την τρέχουσα κατάστασή της.

4.4 Τεστ αξιολόγησης

1. Το Scratch χρειάζεται μια πρόσθετη επέκταση για να δουλέψει με το GPIO του Raspberry.
   a. **Σωστό**
   b. Λάθος
2. Η επέκταση GPIO προσθέτει τρία επιπλέον μπλοκ στο Scratch.
   a. Σωστό
   b. **Λάθος**
3. Αν η ακίδα GPIO χρησιμοποιηθεί ως έξοδος, το Scratch δεν μπορεί να «διαβάσει» την κατάστασή της.
   a. Σωστό
   b. **Λάθος**
4. Το περισσότερο ρεύμα που οποιοδήποτε συνδεδεμένο κύκλωμα τραβάει από μια θύρα GPIO είναι:
   a. 25mA
   b. 35mA
   c. **50mA**
5. Ο έλεγχος της κατάστασης μιας ακίδας GPIO μπορεί να γίνει μέσω:
   a. ενός μπλοκ στοίβας (stack block)
   b. **ενός δυαδικού (Boolean) μπλοκ**
   c. ενός μπλοκ ελέγχου
6. Η επέκταση GPIO χρησιμοποιεί αριθμούς που υποδηλώνουν:
   a. τις αριθμημένες ακίδες του Raspberry
   b. **τους καθορισμένους αριθμούς GPIO του Raspberry**

   c. τους κανόνες ονομασίας του Scratch
7. Η ρύθμιση μιας ακίδας GPIO σε υψηλή κατάσταση στο Scratch θα έχει ως αποτέλεσμα η τάση της ακίδας να είναι:
   a. 0V
   b. 1V
   c. 3.3V
8. Το Scratch είναι ικανό να ελέγξει μόνο επιλεγμένες ακίδες GPIO.
   a. Σωστό
   b. Λάθος
9. Το προτεινόμενο λειτουργικό σύστημα για εργασία με τα Scratch και GPIO είναι:
   a. Windows
   b. Raspbian
   c. macOS
10. Το μπλοκ στοίβας που παρέχεται από την επέκταση GPIO χρησιμοποιείται για:
    a. ν' αλλάξουμε την κατάσταση της ακίδας GPIO
    b. να «διαβάσουμε» την κατάσταση της ακίδας GPIO
    c. όλα τα παραπάνω

4.5 Παραπομπές

- https://www.raspberrypi.org/documentation/usage/gpio/
- https://www.raspberrypi.org/documentation/usage/gpio/scratch2/README.md
- https://www.circuits.dk/everything-about-raspberry-gpio/
- https://thepihut.com/blogs/raspberry-pi-tutorials/tutorial-tactile-switch
- https://raspberrypihq.com/use-a-push-button-with-raspberry-pi-gpio/

4.6 Επιπρόσθετες πηγές

1. Raspberry Pi GPIO: https://www.raspberrypi.org/documentation/usage/gpio/
2. Scratch Wiki: https://scratch-wiki.info/
3. Μια εναλλακτική επέκταση GPIO για το Scratch: https://raspberry-valley.azurewebsites.net/GPIO-with-Scratch/
4. Εκπαίδευση Sparkfun πάνω στα Scratch και GPIO: https://sparkfuneducation.com/how-to/scratch-gpio-control-guide.html
5. Σύντομο μάθημα για αρχάριους στο Scratch και το GPIO: https://www.magicbytes.com/coming-soon/gaming-computer-lab/tutorials-s/learn-to-code/scratch-gpio-beginner

5 Εισαγωγή στην Έκδοση Raspberry Pi του Minecraft [P4-HESO / P5-SCHOLE]

5.1 Λεξικό όρων

Όρος / Έννοια	Ορισμός / Εξήγηση
Raspberry Pi	Το Raspberry Pi είναι ένας πλήρως λειτουργικός υπολογιστής σε σχήμα πιστωτικής κάρτας, που λειτουργεί στο Raspberry Pi OS.
Minecraft	Το Minecraft είναι ένα εκπαιδευτικό παιχνίδι ανοιχτού κόσμου, όπου οι παίχτες μπορούν να χτίσουν τους δικούς τους εικονικούς κόσμους με μπλοκ που αντιπροσωπεύουν διαφορετικά υλικά.
Raspberry Pi OS	Το λειτουργικό σύστημα για το Raspberry Pi.
Python	Μια αντικειμενοστραφής γλώσσα προγραμματισμού, που θα χρησιμοποιηθεί για την αυτόματη οικοδόμηση πραγμάτων στο Minecraft.

5.2 Κυρίως περιεχόμενο

Το Minecraft Pi είναι μια έκδοση του Minecraft, με ελάχιστα χαρακτηριστικά, που αναπτύχθηκε για το Raspberry Pi. Η έκδοση Pi προορίζεται να είναι ένα εκπαιδευτικό εργαλείο για αρχάριους προγραμματιστές, επιτρέποντας στους χρήστες να απολαύσουν το παιχνίδι και να μάθουν προγραμματισμό παράλληλα. Αυτό το έγγραφο παρουσιάζει τις πιο σημαντικές και πρακτικές κατευθυντήριες γραμμές για το Minecraft Pi, όπως το πώς να ελέγξετε τον παίχτη, να χτίσετε χειροκίνητα με μπλοκ και να χρησιμοποιήσετε το περιβάλλον προγραμματισμού Python για να χειριστείτε τον κόσμο γύρω σας. Προορίζεται για εκπαιδευτικούς σκοπούς και θεωρείται ένα ταχύρρυθμο εγχειρίδιο, που περιλαμβάνει όμως τα πάντα για να μυήσει ένα νέο παίχτη στο Minecraft Pi.
Η διδακτική ενότητα αποτελείται από τα εξής μέρη:

- Εισαγωγή στις βασικές λειτουργίες του Minecraft Pi
- Στοιχεία και παίξιμο παιχνιδιού στο Minecraft Pi
- Ελέγχοντας το Minecraft Pi με τη γλώσσα προγραμματισμού Python
- Αλληλεπίδραση του Minecraft Pi με το φυσικό κόσμο μέσω του GPIO του Raspberry Pi:
    - Σύνδεση LEDs, κουμπιών και διακοπτών
    - Δημιουργία ηλεκτρικών κιτ που αλληλεπιδρούν με το Minecraft Pi
- Εισαγωγή νέων χαρτών Minecraft Pi και βοηθητικών πακέτων
- Πρακτικές εφαρμογές των Minecraft Pi και Python
- Τεστ αξιολόγησης για να ελέγξετε τις αποκτηθείσες γνώσεις σας
- Επιπλέον πηγές για να επεκτείνετε τις γνώσεις σας.

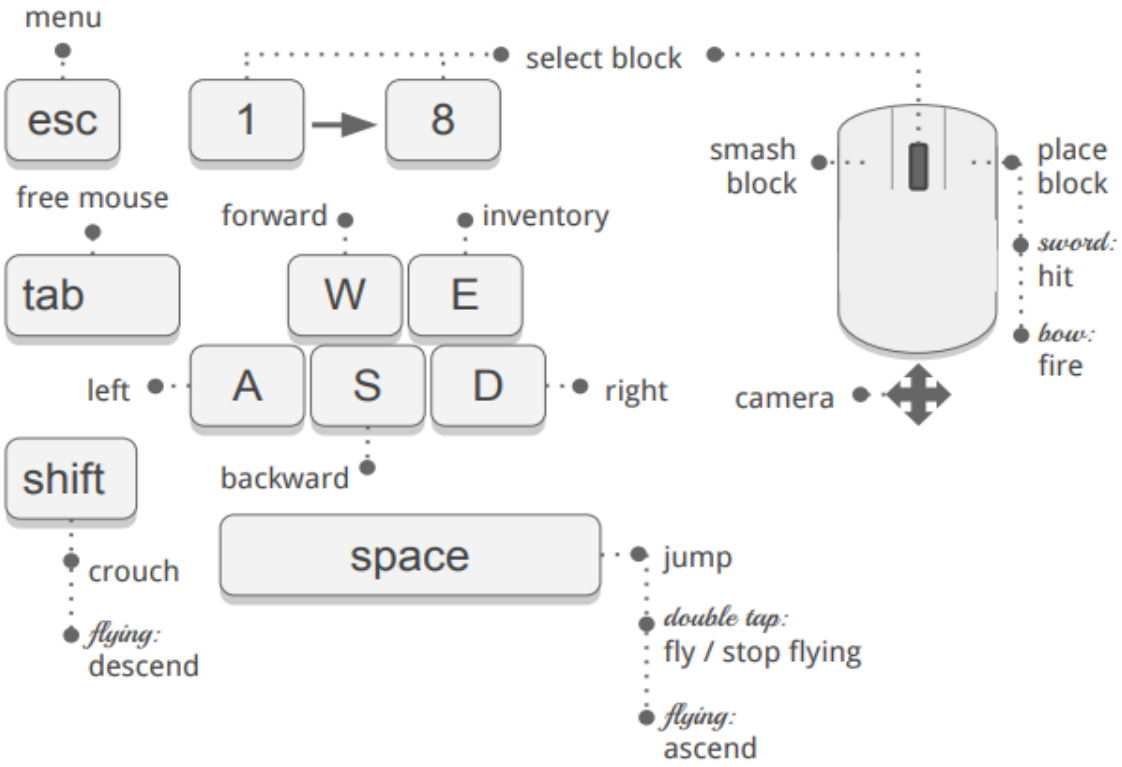
5.2.1 Εισαγωγή στις βασικές λειτουργίες του Minecraft Pi

Το Minecraft Pi είναι ένα παιχνίδι ανοιχτού κόσμου, όπου οι παίχτες χρησιμοποιούν τετράγωνα (μπλοκ) που αντιπροσωπεύουν διαφορετικά υλικά για να χτίσουν εικονικούς κόσμους. Μπορείτε να δημιουργήσετε τα πάντα, από ένα απλό σπίτι μέχρι ένα τεράστιο κάστρο, κι από ένα μικρό χωράφι μέχρι μια μεγάλη πόλη.
Το Minecraft Pi μπορεί να χειριστεί χρησιμοποιώντας κώδικες της Python, που αλληλεπιδρούν με διάφορες λειτουργίες του παιχνιδιού. Η έκδοση Raspberry Pi του Minecraft συνοδεύεται από ένα API (Application Programming Interface, στα ελληνικά Διασύνδεση Προγραμματισμού Εφαρμογών), που σας επιτρέπει να ελέγξετε το παιχνίδι χρησιμοποιώντας τη γλώσσα προγραμματισμού Python. Η Python θα χρησιμοποιηθεί για να χειριστεί τα μπλοκ και τις κατασκευές, να στείλει μηνύματα μέσα στο παιχνίδι, να αυτοματοποιήσει τις διαδικασίες οικοδόμησης και να δημιουργήσει μίνι παιχνίδια.
Το Minecraft Pi υποστηρίζει επίσης τους πολλαπλούς παίχτες, που σημαίνει ότι περισσότεροι από ένας παίκτες μπορούν να παίξουν και να αλληλεπιδράσουν ο ένας με τον άλλο στον ίδιο χάρτη. Όταν αρκετοί υπολογιστές STEMKIT είναι συνδεδεμένοι μέσω του ίδιου ασύρματου (Wi-Fi) ή ενσύρματου (Ethernet) δικτύου, ενεργοποιείται η λειτουργία πολλαπλών παιχτών, και αρκετοί χρήστες μπορούν να συνδεθούν στον ίδιο κόσμο Minecraft.
Για να «τρέξετε» το Minecraft Pi στον STEMKIT υπολογιστή σας, θα πρέπει να κάνετε διπλό κλικ στο εικονίδιο του Minecraft Pi στην επιφάνεια εργασίας σας ή να πάτε στο **Αρχικό Μενού** (το λογότυπο του Raspberry Pi στην πάνω αριστερή γωνία) → **Παιχνίδια** → **Minecraft Pi** και να κάνετε κλικ σ' αυτό (Εικόνα 1). Όταν φορτώσει το παιχνίδι, κάνετε κλικ στην **Εκκίνηση Παιχνιδιού** (**Start Game)** → **Δημιουργία Νέου** (**Create New)**.

ΕΙΚΟΝΑ 1. ΕΚΤΕΛΕΣΗ ΤΟΥ MINECRAFT PI

Οι βασικές λειτουργίες περιλαμβάνουν πλοήγηση κι έλεγχο κατά την είσοδο στον κόσμο του Minecraft. Χρησιμοποιήστε το ποντίκι για να δείτε γύρω σας, κάντε αριστερό κλικ για

να διαλύσετε τα μπλοκ και δεξί για να τα χτίσετε. Οι περισσότερες λειτουργίες ελέγχου προέρχονται από το πληκτρολόγιο, όπως φαίνεται στην εικόνα 2:



ΕΙΚΟΝΑ 2. ΣΥΝΟΨΗ ΤΩΝ ΕΛΕΓΧΩΝ ΤΟΥ MINECRAFT (ΠΗΓΗ: HTTPS://ARGHBOX.WORDPRESS.COM/2013/07/28/MINECRAFT-PI-CONTROLS/)

5.2.2 Στοιχεία και παίξιμο παιχνιδιού στο Minecraft Pi

Όταν ο χρήστης εισέρχεται στο παιχνίδι για πρώτη φορά, ξεκινάει μ' ένα σπαθί στο χέρι, το οποίο μπορεί να χρησιμοποιηθεί για την αφαίρεση των μπλοκ. Στο κάτω μέρος της οθόνης υπάρχει μια μερική προβολή της εργαλειοθήκης, με διαφορετικούς τύπους μπλοκ και εργαλείων, στα οποία μπορείτε να αποκτήσετε γρήγορη πρόσβαση κυλώντας προς τα πάνω ή προς τα κάτω τη ροδέλα του ποντικιού. Αυτά τα αντικείμενα χρησιμοποιούνται συνήθως περισσότερο σ' ένα παιχνίδι, αλλά μπορούν ν' αλλάξουν ανάλογα με το τι θέλει να χτίσει ο χρήστης και σε τι υλικά θέλει να έχει γρήγορη πρόσβαση.
Πατώντας το πλήκτρο διαστήματος, ο χαρακτήρας πηδάει. Πατήστε το δύο φορές για να αρχίσει να ανεβαίνει (να πετάει γύρω γύρω). Όταν αφήσετε το πλήκτρο διαστήματος, η ανάβαση σταματά. Για να πέσει στο έδαφος, ο παίχτης πρέπει να πατήσει δύο φορές το πλήκτρο διαστήματος ξανά.
Το πάτημα του «E», θα ανοίξει την εργαλειοθήκη που περιέχει όλους τους διαφορετικούς τύπους μπλοκ, που προσφέρονται στο Minecraft Pi. Περιηγηθείτε πάνω τους και κάντε αριστερό κλικ στο μπλοκ που θέλετε να χρησιμοποιήσετε.
Πατώντας το πλήκτρο «TAB» θα απελευθερώσετε το ποντίκι απ' το παιχνίδι, δίνοντας την ευκαιρία στο χρήστη να χρησιμοποιήσει άλλα προγράμματα του Raspberry Pi. Κατά τη γραφή κώδικα, το πλήκτρο «TAB» θα επιτρέψει στο χρήστη να πλοηγηθεί σε διάφορα

αρχεία της Python, στο μεταγλωττιστή της Python, ή στη γραμμή Εντολών, ανάλογα με το πώς θέλει ο χρήστης να προγραμματίσει.
Το πάτημα του πλήκτρου «SHIFT» κατά τη διάρκεια του παιχνιδιού θα κάνει το χαρακτήρα να σκύψει. Κατά τη διάρκεια της πτήσης το πλήκτρο «SHIFT» θα κάνει το χαρακτήρα σας να κατέβει.
Τέλος, τα πλήκτρα 1 έως 8 δίνουν στο χρήστη γρήγορη πρόσβαση σε υλικά κι εργαλεία που βρίσκονται ήδη στον κατάλογο γρήγορης πρόσβασης.
Τα μπλοκ είναι οι βασικές μονάδες δόμησης στο Minecraft. Κάθε μπλοκ είναι $1m^3$. Είναι οργανωμένα σε ένα πλέγμα και δεσμεύονται σ' αυτό, που σημαίνει ότι ένα μπλοκ δεν μπορεί να είναι σε παραπάνω από ένα κελιά. Υπάρχουν κάποιες εξαιρέσεις, όπως κρεβάτια που αποτελούνται από δύο block, που το καθένα αποτελεί το μισό του.
Τα μπλοκ αναπτύσσουν το περιβάλλον του παιχνιδιού και μπορούν να συλλεχθούν και να αξιοποιηθούν με διάφορους τρόπους. Κάποια μπλοκ, όπως το χώμα και ο ψαμμίτης, είναι αδιαφανή, ενώ άλλα, όπως το γυαλί και ο πάγος, είναι διαφανή. Κάποια άλλα, όπως οι πυρσοί, εκπέμπουν φως. Σχεδόν όλα τα μπλοκ αγνοούν τη βαρύτητα, με εξαίρεση την άμμο και το χιόνι.
Υπάρχουν διάφορα είδη μπλοκ στο Minecraft Pi. Τα φυσικά μπλοκ αναφέρονται σε μπλοκ που δημιουργήθηκαν ως μέρος του εδάφους. Τα φυτά, τα ζώα και οι μύκητες είναι οι αναπαραστάσεις των διαφόρων μορφών ζωής του Minecraft.
Κάθε τύπος μπλοκ έχει το δικό του μοναδικό κωδικό αριθμό (ID) που σχετίζεται μ' αυτό. Οι κωδικοί αριθμοί των μπλοκ χρειάζονται για το χειρισμό διαφορετικών ειδών μπλοκ κατά το γράψιμο κώδικα στην Python. Είναι χρήσιμο να έχετε μια λίστα αυτών των κωδικών αριθμών για να τους χρησιμοποιήσετε στον κώδικά σας.
Για παράδειγμα, ο κωδικός αριθμός μπλοκ για την «Πέτρα» είναι 1, για το «Γυαλί» είναι 20, κ.ο.κ. Η ολοκληρωμένη λίστα των κωδικών αριθμών των μπλοκ μπορεί να βρεθεί στον ακόλουθο σύνδεσμο: https://www.raspberrypi-spy.co.uk/2014/09/raspberry-pi-minecraft-block-id-number-reference/.

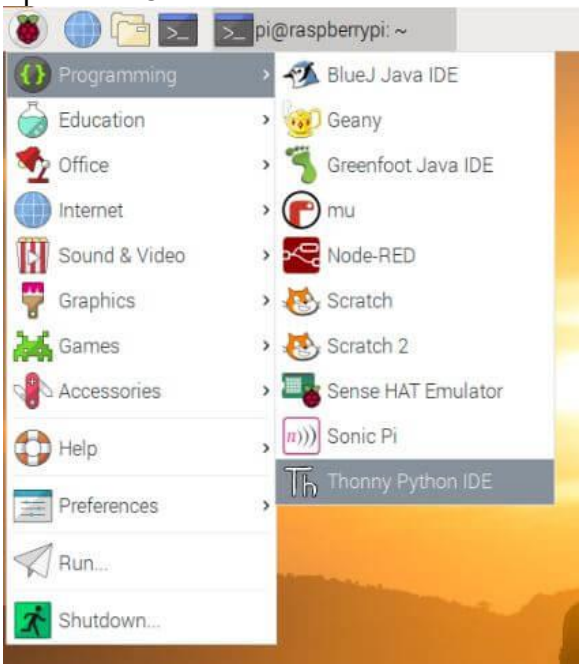
5.2.3 Ελέγχοντας το Minecraft Pi με τη γλώσσα προγραμματισμού Python

Το Minecraft Pi έχει ένα API, που επιτρέπει στο χρήστη να ελέγχει και να επικοινωνεί με το παιχνίδι χρησιμοποιώντας κώδικες της Python. Το API σας επιτρέπει να γράψετε προγράμματα που ελέγχουν, τροποποιούν και αλληλεπιδρούν με τον κόσμο του Minecraft. Ο χρήστης μπορεί να δημιουργήσει τεράστιες κατασκευές με το πάτημα ενός κουμπιού ή μια γέφυρα που εμφανίζεται αυτόματα, επιτρέποντας στο χρήστη να διασχίσει τεράστια χάσματα, ή ένα παιχνίδι ναρκαλιευτή, ένα πελώριο ρολόι πραγματικού χρόνου, ένα προγραμματίσιμο κατευθυντικό κανόνι, ή να μεταμορφώσει τα μπλοκ σε βόμβες (O'Hanlon, 2013) (https://www.stuffaboutcode.com/2013/04/minecraft-pi-edition-api-tutorial.html).
Αφού το Minecraft είναι ένας κόσμος κύβων ή μπλοκ, με σχετικό μέγεθος 1m x 1m x 1m, κάθε μπλοκ έχει τη δικιά του ξεχωριστή θέση στον κόσμο σε συντεταγμένες x, y, z (το **x** είναι μπρος/πίσω, το **z** είναι αριστερά/δεξιά, και το **y** είναι πάνω/κάτω).
Χρησιμοποιώντας κώδικες της Python μπορείτε, μεταξύ άλλων, να κάνετε τα ακόλουθα:

- Εντοπισμός της θέσης του παίκτη.
- Αλλαγή ή ορισμός της θέσης του παίκτη.
- Λήψη του τύπου μπλοκ πάνω στο οποίο στέκεται ο χρήστης ή είναι κοντά του
- Αλλαγή ενός τύπου μπλοκ με έναν άλλο.
- Αλλαγή οπτικής γωνίας κάμερας.

- Αποστολή μηνυμάτων στον παίκτη.

Για να ανοίξετε το περιβάλλον προγραμματισμού της Python, πηγαίνετε στο Κεντρικό Μενού (Main Menu) → Προγραμματισμός (Programming), και κάνετε κλικ στο Thonny Python, όπως φαίνεται στην εικόνα 3.



ΕΙΚΟΝΑ 3. ΕΝΤΟΠΙΣΜΟΣ ΤΟΥ THONNY PYTHON

Στις πρώτες γραμμές του κώδικά μας, θα μάθουμε πώς να συνδέουμε το Minecraft Pi με τον κώδικα. Πρώτα απ' όλα, αποθηκεύστε το αρχείο σας χρησιμοποιώντας το όνομα *first.py*. Θυμηθείτε να αποθηκεύετε πάντα τα αρχεία σας και να χρησιμοποιείτε την επέκταση αρχείου *.py*, ώστε το πρόγραμμά σας να μεταφράζετε σε αρχείο της Python.
Η εικόνα 4 δείχνει τις γραμμές κώδικα που απαιτούνται, ώστε να συνδέσετε το Minecraft Pi με την Python.

first.py

```
1 from mcpi.minecraft import minecraft
2
3 mc = Minecraft.create()
```

ΕΙΚΟΝΑ 4. ΣΥΝΔΕΣΗ ΤΗΣ PYTHON ΜΕ ΤΟ MINECRAFT PI

Κατ' αρχάς, κάνουμε εισαγωγή (import) του Minecraft. Έπειτα, αποθηκεύουμε το αντικείμενο παιχνιδιού Minecraft σε μια μεταβλητή που ονομάζεται «mc», που θα χρησιμοποιηθεί αργότερα για να καλέσει εντολές στο πρόγραμμά σας.
Το πρώτο ολοκληρωμένο πρόγραμμά μας θα επικοινωνεί ένα μήνυμα στο χαρακτήρα μας στο Minecraft. Δημιουργήστε ένα νέο αρχείο και αποθηκεύστε το με το όνομα *hello.py*.

Η εικόνα 5 δείχνει τον κώδικα που χρειάζεται να γράψετε:

hello.py

```
1  from mcpi.minecraft import minecraft
2
3  mc = Minecraft.create()
4
5  mc.postToChat("Hello World!")
```

ΕΙΚΟΝΑ 5. ΑΠΟΣΤΟΛΗ ΕΝΟΣ ΜΗΝΥΜΑΤΟΣ ΣΤΟΝ ΠΑΙΚΤΗ

Όταν γράψετε τον κώδικα, κάντε κλικ στην Αποθήκευση και πατήστε το πλήκτρο F5 για να εκτελέσετε το σενάριό σας. Ένα μήνυμα εμφανίζεται στο Minecraft.

Εντοπισμός θέσης του παίκτη:

Ο εντοπισμός της θέσης του παίκτη είναι απαραίτητος πριν από την οικοδόμηση οποιασδήποτε κατασκευής. Για να τον επιτύχουμε, χρησιμοποιούμε την εντολή `mc.player.getPos()`. Υπάρχουν δύο τρόποι για τον εντοπισμό της θέσης του παίκτη:

1. Χρησιμοποιούμε μια μεταβλητή που ονομάζεται «pos» και αποθηκεύουμε τις συντεταγμένες του παίκτη, όπως φαίνεται στην εικόνα 6:

pos.py

```
1  from mcpi.minecraft import minecraft
2
3  mc = Minecraft.create()
4
5  pos = mc.player.getPos()
```

ΕΙΚΟΝΑ 6. ΑΠΟΘΗΚΕΥΣΗ ΤΗΣ ΘΕΣΗΣ ΤΟΥ ΠΑΙΚΤΗ ΣΤΗ ΜΕΤΑΒΛΗΤΗ POS

2. Αποθηκεύουμε τη θέση του παίκτη σε συντεταγμένες ***xyz***, όπως φαίνεται στην εικόνα 7 (το **x** είναι μπρος/πίσω, το **z** είναι αριστερά/δεξιά, το **y** είναι πάνω/κάτω):

pos.py *

```
1  from mcpi.minecraft import minecraft
2
3  mc = Minecraft.create()
4
5  x, y, z = mc.player.getPos()
```

ΕΙΚΟΝΑ 7 ΑΠΟΘΗΚΕΥΣΗ ΘΕΣΗΣ ΠΑΙΚΤΗ ΣΕ ΣΥΝΤΕΤΑΓΜΕΝΕΣ XYZ

Τηλεμεταφορά του παίκτη μας:

Το να έχουμε τις συντεταγμένες του παίκτη σημαίνει επίσης ότι μπορούμε να τις χειριστούμε, δηλαδή μπορούμε να τηλεμεταφέρουμε τον παίκτη μας σε διάφορα μέρη σ' έναν κόσμο του Minecraft.
Για να το επιτύχουμε αυτό, χρησιμοποιούμε την εντολή `mc.player.setPos()`. Η εικόνα 8 δείχνει πώς να χρησιμοποιήσουμε αυτή την εντολή για να τηλεμεταφέρουμε τον παίκτη μας 200 διαστήματα ψηλά στον αέρα κι έπειτα να τον δούμε να πέφτει στην αρχική του θέση στο έδαφος.

tel.py

```
1  from mcpi.minecraft import minecraft
2
3  mc = Minecraft.create()
4
5  x, y, z = mc.player.getPos()
6
7  mc.player.setPos(x, y+200, z)
```

ΕΙΚΟΝΑ 8 ΤΗΛΕΜΕΤΑΦΟΡΑ ΤΟΥ ΠΑΙΚΤΗ ΜΑΣ 200 ΔΙΑΣΤΗΜΑΤΑ ΨΗΛΑ ΣΤΟΝ ΑΕΡΑ

Διαχείριση των μπλοκ:
Εκτός του εντοπισμού και του χειρισμού της θέσης του παίκτη μας, μπορούμε να χρησιμοποιήσουμε τη θέση του παίκτη μας για να αλληλεπιδράσουμε με τα μπλοκ. Ένα απ' τα πράγματα που μπορούμε να κάνουμε, είναι να γνωρίζουμε ακριβώς σε ποιο μπλοκ στέκεται πάνω ο παίκτης μας. Για να το πετύχουμε αυτό, πρώτα βρίσκουμε τη θέση του πλακιδίου χρησιμοποιώντας την εντολή `mc.getTilePos()`, κι έπειτα χρησιμοποιούμε την εντολή `mc.getBlock(x,y,z)`, όπως φαίνεται στην εικόνα 9.

tile.py

```
1  from mcpi.minecraft import minecraft
2
3  mc = Minecraft.create()
4
5  x, y, z = mc.getTilePos()
6
7  blockBelowPlayerType = mc.getBlock(x, y, z)
```

ΕΙΚΟΝΑ 9. ΕΝΤΟΠΙΣΜΟΣ ΘΕΣΗΣ ΠΛΑΚΙΔΙΟΥ ΚΙ ΑΠΟΘΗΚΕΥΣΗ ΤΟΥ ΣΤΙΣ ΣΥΝΤΕΤΑΓΜΕΝΕΣ

Έπειτα μπορούμε να δημιουργήσουμε μπλοκ γύρω απ' τον παίκτη μας, χρησιμοποιώντας την εντολή `mc.setBlock(x,y,z,blockType,blockData)`. Οι συντεταγμένες *xyz* αναφέρονται στην τοποθεσία στην οποία δημιουργούμε ένα μπλοκ, το `blockType` αναφέρεται στους διαφορετικούς κωδικούς αριθμούς των τύπων μπλοκ, και το `blockData` αναφέρεται στις επιπλέον ιδιότητες που έχουν κάποια μπλοκ (π.χ., διαφορετικά χρώματα). Η εικόνα 10 δείχνει πώς να δημιουργήσετε ένα μπλοκ από πέτρα (το `blockType` είναι 1) δίπλα στον παίκτη σας.

block.py

```
1  from mcpi.minecraft import minecraft
2
3  mc = Minecraft.create()
4
5  x, y, z = mc.player.getPos()
6
7  mc.setBlock(x+1, y, z, 1)
```

ΕΙΚΟΝΑ 10. ΔΗΜΙΟΥΡΓΙΑ ΕΝΟΣ ΠΕΤΡΙΝΟΥ ΜΠΛΟΚ ΔΙΠΛΑ ΣΤΟΝ ΠΑΙΚΤΗ

Όπως προαναφέρθηκε, ορισμένα μπλοκ έχουν επιπλέον ιδιότητες. Για παράδειγμα:

- Μαλλί (Wool) (ID-35) □ 0: λευκό, 1: πορτοκαλί, 2: φούξια, 3: ανοιχτό μπλε, 4: κίτρινο, κ.λπ.
- Ξύλο (Wood) (ID-17) □ 0: βελανιδιά, 1: έλατο, 2: σημύδα, κ.λπ.
- Ψηλή χλόη (Tall grass) (ID-31) □ 0: θάμνος, 1: γρασίδι, 2: φτέρη
- Πυρσός (Torch) (ID-50) □ 0: κατεύθυνση ανατολική, 1: δυτική, 2: βόρεια, 3: νότια

Η ολοκληρωμένη λίστα των τύπων μπλοκ και των ιδιοτήτων τους είναι διαθέσιμη στον ακόλουθο σύνδεσμο: https://minecraft.gamepedia.com/Block.
Τέλος, το API επιτρέπει τη δημιουργία πολλαπλών μπλοκ μαζί, ώστε να μπορούμε να δημιουργήσουμε διάφορες δομές με το πάτημα ενός κουμπιού. Για να το επιτύχουμε, χρησιμοποιούμε την εντολή `setBlocks(x1,y1,z1,x2,y2,z2,blockType,blockData)`. Λειτουργεί προσδιορίζοντας δύο σετ συντεταγμένων, τις οποίες χρησιμοποιεί το API για να συμπληρώσει το κενό μεταξύ τους μ' ένα συγκεκριμένο τύπο μπλοκ. Η εικόνα 11 δείχνει πώς να φτιάξετε ένα μπλοκ διαστάσεων 10x10x10, φτιαγμένο από χρυσάφι.

goldblock.py

```
1  from mcpi.minecraft import minecraft
2
3  mc = Minecraft.create()
4
5  gold = 41
6
7  x, y, z = mc.player.getPos()
8
9  mc.setBlocks(x+1, y, z+1, x+11, y+11, z+11, gold)
```

ΕΙΚΟΝΑ 11. ΔΗΜΙΟΥΡΓΙΑ ΕΝΟΣ 10X10X10 ΧΡΥΣΟΥ ΜΠΛΟΚ

Ειδικά μπλοκ:
Τα ειδικά μπλοκ μέσα στο Minecraft Pi αναφέρονται σε μπλοκ που μπορούν να αλληλεπιδράσουν με το περιβάλλον ενός κόσμου του Minecraft. Αυτά τα μπλοκ κυμαίνονται από απλούς μηχανισμούς μέχρι ρέουσα λάβα. Η ολοκληρωμένη λίστα των ειδικών μπλοκ είναι διαθέσιμη στον ακόλουθο σύνδεσμο: https://minecraft.gamepedia.com/Block. Δύο είναι τα ειδικά μπλοκ που χρησιμοποιούνται πιο συχνά στο Minecraft: τα μπλοκ TNT και Λάβα.

Τα μπλοκ TNT μπορούν να χρησιμοποιηθούν απ' τον παίκτη για να δημιουργήσει ελεγχόμενες εκρήξεις που καταστρέφουν κατασκευές, βουνά και χωράφια. Ο παίκτης μπορεί να τοποθετήσει ένα μπλοκ TNT όπως οποιοδήποτε άλλο μπλοκ, αλλά όταν κάνουμε δεξί κλικ μερικές φορές αυτό το μπλοκ TNT πυροδοτείται, δίνοντας ένα περιθώριο 4 δευτερολέπτων στον παίκτη να κινηθεί μακριά, πριν να εκραγεί. Ο κωδικός του αριθμός είναι 46. Η εικόνα 12 δείχνει πώς να δημιουργήσετε μπλοκ TNT χρησιμοποιώντας τον κώδικα της Python (παρατηρείστε ότι το τελευταίο ψηφίο πρέπει πάντα να είναι "1", για να ανατιναχθεί το μπλοκ TNT).

tnt.py

```
1  from mcpi.minecraft import minecraft
2
3  mc = Minecraft.create()
4
5  tnt = 46
6
7  mc.setBlock(x, y, z, tnt, 1)
```

ΕΙΚΟΝΑ 12. ΔΗΜΙΟΥΡΓΙΑ ΤΩΝ ΜΠΛΟΚ TNT ΜΕ ΤΗ ΧΡΗΣΗ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ ΤΗΣ PYTHON

Η Λάβα είναι ένα ρευστό μπλοκ που εκπέμπει φως, προκαλεί ζημιά ανάλογη της πυρκαγιάς και απλώνεται σε μια περιοχή 3x3 από πάνω της και 5x5 από κάτω της. Ο κωδικός της αριθμός είναι 10. Η λάβα μπορεί να κάψει εύφλεκτες κατασκευές, όπως το γρασίδι και το ξύλο, αλλά και τον παίκτη, οπότε πρέπει να είμαστε προσεκτικοί να μην πατήσουμε πάνω της. Όταν η λάβα κρυώσει, γίνεται βράχος. Η εικόνα 13 δείχνει πώς να δημιουργήσετε Λάβα χρησιμοποιώντας τον κώδικα της Python:

lava.py

```
1  from mcpi.minecraft import minecraft
2
3  mc = Minecraft.create()
4
5  lava = 10
6
7  mc.setBlock(x+3, y, z+3, lava)
```

ΕΙΚΟΝΑ 13. ΔΗΜΙΟΥΡΓΙΑ ΜΠΛΟΚ ΛΑΒΑΣ ΜΕ ΤΗ ΧΡΗΣΗ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ ΤΗΣ PYTHON

Εισαγωγή προγραμματιστικών πακέτων (modules) του Minecraft Pi:
Για να εξασφαλίσουμε ότι τα προγράμματά μας «τρέχουν» ομαλά, χωρίς να «κρασάρουν» το παιχνίδι, χρειάζεται να εισάγουμε κάποια προγραμματιστικά πακέτα (modules) του Minecraft στην αρχή του κώδικά μας στην Python. Αυτά τα προγραμματιστικά πακέτα μας επιτρέπουν επίσης την πρόσβαση σε ιδιότητες (properties) και παραμέτρους (parameters)

που είναι απαραίτητες για το χειρισμό ενός κόσμου στο Minecraft. Η εικόνα 14 δείχνει πού και πώς πρέπει να εισαχθούν αυτά τα προγραμματιστικά πακέτα:

modules.py

```
1  from mcpi.minecraft import minecraft
2  from mcpi.block import block
3  from time import sleep
4
5  mc = Minecraft.create()
6
7  # rest of program
8  # ...
9  # ...
10 # ...
```

ΕΙΚΟΝΑ 14. ΑΠΑΡΑΙΤΗΤΑ ΠΡΟΓΡΑΜΜΑΤΙΣΤΙΚΑ ΠΑΚΕΤΑ (MODULES) ΣΤΟΝ ΚΩΔΙΚΑ PYTHON

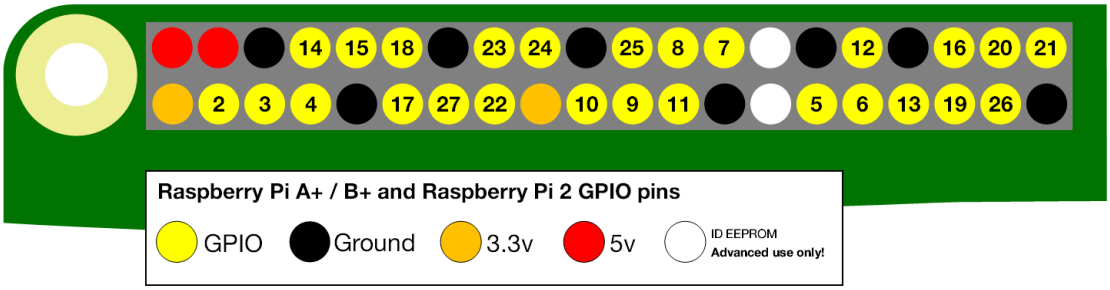
Ανακεφαλαίωση των εντολών στην Python:

- `postToChat("our message")` – επικοινωνία με το χρήστη στο παιχνίδι,
- `player.getPos()` – εντοπισμός ακριβούς τοποθεσίας ενός παίκτη,
- `player.setPos(x, y, z)` – ρύθμιση (αλλαγή) της θέσης του παίκτη,
- `player.getTilePos()` – εντοπισμός της θέσης του μπλοκ στο οποίο βρίσκεται ο παίκτης,
- `getBlock(x, y, z, blockType, blockData)` – λήψη του τύπου του μπλοκ για μια συγκεκριμένη θέση,
- `setBlock(x, y, z, blockType, blockData)` – ορισμός (αλλαγή) ενός μπλοκ σ' ένα συγκεκριμένο τύπο μπλοκ,
- `setBlocks(x1, y1, z1, x2, y2, z2, blockType, blockData)` - ορισμός πολλών μπλοκ ταυτοχρόνως, παρέχοντας 2 σύνολα συντεταγμένων x, y, z.

5.2.4 Αλληλεπίδραση με τον υλικό κόσμο μέσω του GPIO

Αυτή η ενότητα περιγράφει μερικά σύντομα μαθήματα, ώστε να μπορέσετε να κάνετε το Minecraft να αλληλεπιδράσει με τον υλικό κόσμο μέσω του GPIO. Θα μάθετε να προγραμματίζετε κουμπιά να αλληλεπιδρούν με το Minecraft Pi παιχνίδι σας, να αυτοματοποιείτε διαδικασίες, να χειρίζεστε τα μπλοκ και να δημιουργείτε μίνι παιχνίδια, και, τέλος, να προγραμματίζετε λυχνίες LED να προσομοιώνουν γεγονότα που συμβαίνουν στο Minecraft.

Μάθημα 1: Σύνδεση ενός πιεζόμενου κουμπιού
Σ' αυτό το μάθημα θα μάθετε πώς να συνδέετε ένα πιεζόμενο κουμπί μέσω του GPIO του Raspberry Pi. Θα εξοικειωθείτε με τις ακίδες GPIO, με το ποιες ακίδες να χρησιμοποιήσετε, καθώς και την αρίθμησή τους. Η εικόνα 15 δείχνει τα καθιερωμένα ονόματα των ακίδων Broadcom (BCM). Η αρίθμηση δεν είναι σε αριθμητική σειρά, οπότε να είστε προσεκτικοί/ές.

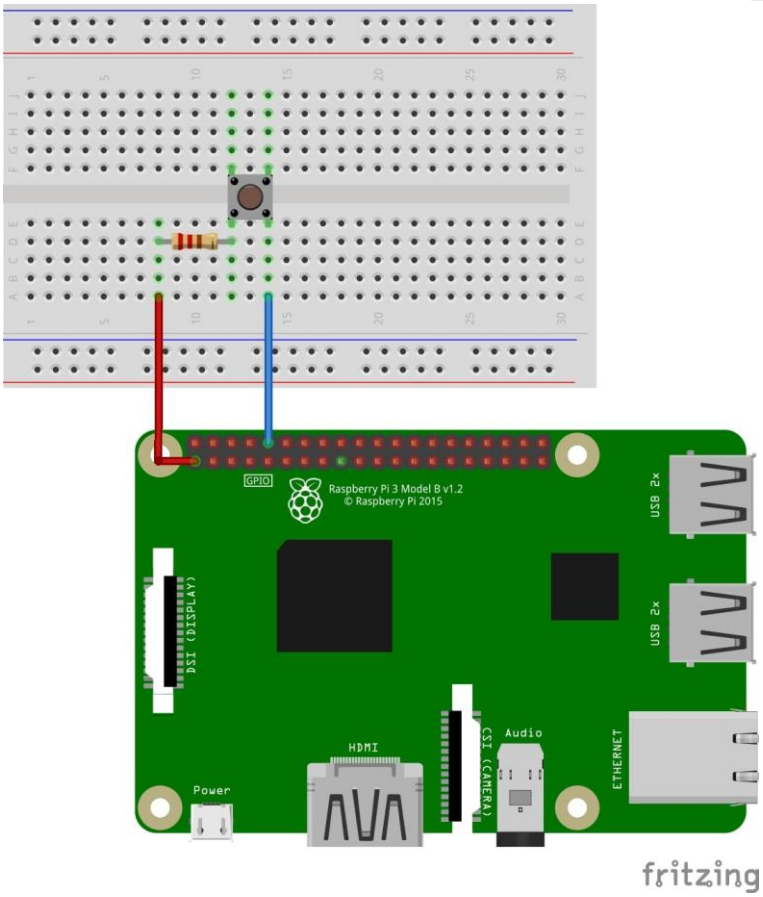


ΕΙΚΟΝΑ 15. ΑΡΙΘΜΗΣΗ ΑΚΙΔΩΝ GPIO

Τα ακόλουθα υλικά είναι απαραίτητα γι' αυτό το μάθημα:

- 1 x πλακέτα δοκιμών (breadboard)
- 1 x πιεζόμενο κουμπί (push button)
- 1 x αντιστάτης 220 Ohm
- 2 x καλώδια μεταγωγής θηλυκό σε αρσενικό

Η συνδεσιμότητα είναι πολύ απλή και παρουσιάζεται στην εικόνα 16. Προσέξτε ότι 1 καλώδιο μεταγωγής είναι συνδεδεμένο με μία ακίδα 3.3V (κόκκινο) και το άλλο είναι συνδεδεμένο με μία ακίδα εισόδου/εξόδου GPIO (αριθμός ακίδας 16).



ΕΙΚΟΝΑ 16. ΣΥΝΔΕΣΗ ΕΝΟΣ ΠΙΕΖΟΜΕΝΟΥ ΚΟΥΜΠΙΟΥ ΣΤΟ RASPBERRY GPIO (ΠΗΓΗ: RASPBERRYPIHQ.COM)

Τώρα που το κύκλωμά σας είναι έτοιμο, μπορείτε να μπείτε σ' ένα νέο αρχείο Python, για να τεστάρετε τη λειτουργικότητα του κουμπιού. Η εικόνα 17 δείχνει τον απαραίτητο κώδικα στην Python, ώστε να τεστάρετε το κουμπί σας.

button_test.py *

```
1  import RPi.GPIO as GPIO
2  import time
3
4  GPIO.setmode(GPIO.BCM) #tell the Pi what headers to use
5  GPIO.setup(15 GPIO.IN) #tell the Pi pin 15 is an input
6
7  while True:
8      if GPIO.input(15) == True: #look for button press
9          print "Button works!" #log result
10         time.sleep(0.5) #wait 0.5 seconds
```

ΕΙΚΟΝΑ 17. ΚΩΔΙΚΑΣ PYTHON ΓΙΑ ΕΛΕΓΧΟ ΛΕΙΤΟΥΡΓΙΚΟΤΗΤΑΣ ΤΟΥ ΚΟΥΜΠΙΟΥ

Πατήστε «Αποθήκευση» κι έπειτα το F5 για να εκτελέσετε τον κώδικα. Τώρα, κάθε φορά που πατάτε το κουμπί, θα πρέπει να εμφανίζεται το μήνυμα «Το κουμπί λειτουργεί!» (“Button works!”) στο παράθυρο τερματικού της Python. Για να σταματήσετε την εκτέλεση του κώδικα, πιέστε *Ctrl+C*.

Μάθημα 2: Σούπερ Εξόρυξη (super mining) στο Minecraft
Σ’ αυτό το δεύτερο μάθημα, θα χρησιμοποιήσετε το προηγούμενο κύκλωμα για να αλληλεπιδράσετε με το παιχνίδι σας στο Minecraft Pi. Θα δημιουργήσετε ένα πρόγραμμα που καταστρέφει ένα μπλοκ από μπλοκ στον κόσμο του Minecraft, κάθε φορά που πατάτε το κουμπί. Δημιουργήστε ένα Νέο Αρχείο κι Αποθηκεύστε το ως *mining.py*.
Ακολουθήστε τον κώδικα της εικόνας 18 και δείτε τι συμβαίνει στον κόσμο του Minecraft όταν εκτελέσετε τον κώδικα (F5) και πατήσετε το κουμπί. Πάρτε υπόψιν σας ότι μπορείτε να χειριστείτε τους τύπους μπλοκ και τις συντεταγμένες όπως επιθυμείτε, αλλά μην το παρακάνετε, γιατί ίσως το Raspberry Pi θα ζοριστεί.

mining.py

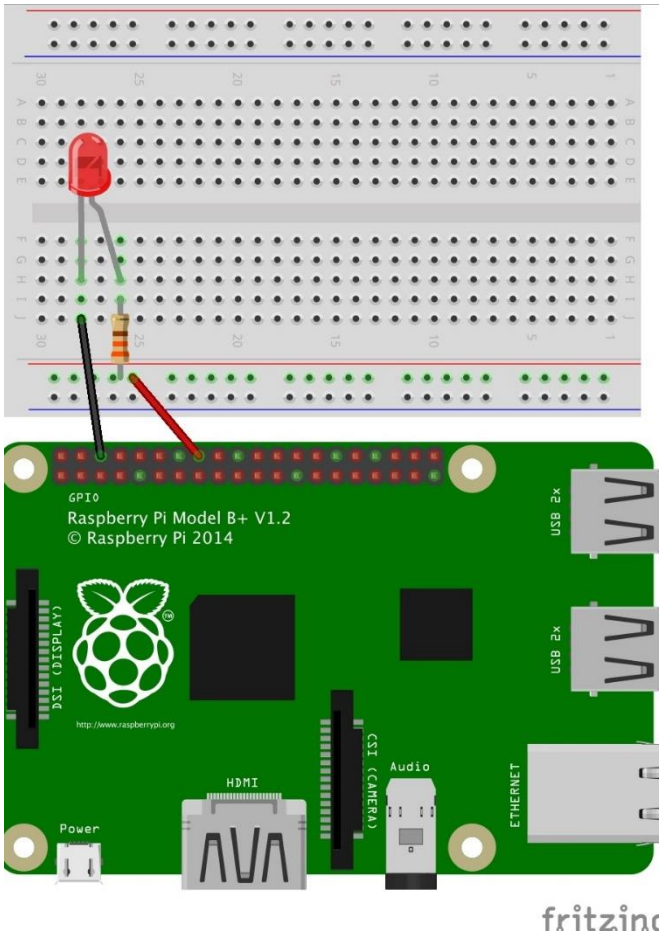
```
1  import RPi.GPIO as GPIO
2  import sleep
3  from mcpi.minecraft import Minecraft
4
5  mc = Minecraft.create() #connect Minecraft Pi with Python
6
7  GPIO.setmode(GPIO.BCM) #tell the Pi what headers to use
8  GPIO.setup(15, GPIO.IN) #tell the Pi pin 15 is an input
9
10 while True:
11     if GPIO.input(15) == True #look for button press
12     x, y, z = mc.player.getPos() #read the player's position
13
14     mc.setBlocks(x, y, z, x+10, y+10, z+10, 0) #mine 10 blocks
15     mc.setBlocks(x, y, z, x-10, y-10, z-10, 0) #mine 10 blocks
16
17     time.sleep(0.5) #wait 0.5 seconds
```

ΕΙΚΟΝΑ 18. ΚΩΔΙΚΑΣ ΤΗΣ PYTHON ΓΙΑ ΔΟΚΙΜΗ ΤΗΣ ΛΕΙΤΟΥΡΓΙΚΟΤΗΤΑΣ ΤΟΥ ΚΟΥΜΠΙΟΥ

Μάθημα 3: Δημιουργήστε έναν ανιχνευτή διαμαντιών (diamond detector)
Στο τρίτο μάθημα θα δημιουργήσετε ένα πρόγραμμα που ανιχνεύει μπλοκ διαμαντιών στο Minecraft. Πρώτα θα χρειαστεί να δημιουργήσετε ένα νέο κύκλωμα, χρησιμοποιώντας τα ακόλουθα υλικά:

- 1 x πλακέτα δοκιμών (breadboard)
- 1 x λυχνία LED (οποιουδήποτε χρώματος)
- 1 x αντιστάτη 220 Ohm
- 2 x καλώδια μεταγωγής θηλυκό σε αρσενικό

Η συνδεσιμότητα είναι πολύ απλή, όπως φαίνεται στην εικόνα 19. Προσέξτε ότι ένα καλώδιο μεταγωγής είναι συνδεδεμένο με μία ακίδα γείωσης (μαύρο) και το άλλο είναι συνδεδεμένο με μια ακίδα εισόδου/εξόδου GPIO (αριθμός ακίδας 23).



ΕΙΚΟΝΑ 19. ΣΥΝΔΕΣΗ ΜΙΑΣ ΛΥΧΝΙΑΣ LED ΣΤΟ GPIO. (ΠΗΓΗ: RASPBERRYPIHQ.COM)

Όταν κατασκευάσετε το κύκλωμα, μπορείτε να ανοίξετε την Python, να δημιουργήσετε ένα Νέο Αρχείο και να το Αποθηκεύσετε ως *detector.py*. Ακολουθήστε τον κώδικα της εικόνας 20, εκτελέστε τον (F5) και κινηθείτε γύρω-γύρω στον κόσμο του Minecraft.

detector.py

```python
import RPi.GPIO as GPIO
import time
from mcpi.minecraft import Minecraft
from mcpi.blocks import Blocks

mc = Minecraft.create() #connect Minecraft with Python

led_pin = 23 #store the GPIO pin number in variable

GPIO.setmode(GPIO.BCM) #tell the Pi what headers to use
GPIO.setup(23, GPIO.OUT) #tell the Pi pin 23 is an output

while True: #repeat indefinitely
    x, y, z = mc.playerPos()
    for i in range(10): #check every block until block 10
        if mc.getBlock(x, y-i, z) == 56:
            GPIO.output(led_pin, True) #turn LED on
            time.sleep(0.5) #wait
            GPIO.output(led_pin, False) #turn LED off
            time.sleep(0.5) #wait
```

ΕΙΚΟΝΑ 20. ΚΩΔΙΚΑΣ ΤΗΣ PYTHON ΓΙΑ ΤΗ ΔΗΜΙΟΥΡΓΙΑ ΕΝΟΣ ΑΝΙΧΝΕΥΤΗ ΔΙΑΜΑΝΤΙΩΝ

5.2.5 Εισαγωγή νέων χαρτών και βοηθητικών πακέτων

Το Minecraft Pi προσφέρει τη δυνατότητα να εισάγετε νέους χάρτες και βοηθητικών πακέτων, για να εξερευνήσετε νέους κόσμους και να βελτιώσετε την εμπειρία σας στο παιχνίδι. Η πρόσθεση νέων χαρτών είναι μια εύκολη διαδικασία:

1. Πρώτα θα πρέπει να βρείτε έναν κόσμο που σας αρέσει και να τον κατεβάσετε στο Raspberry.

2. Όταν βρείτε έναν κόσμο που σας αρέσει, πρέπει να τον κατεβάσετε ως αρχείο .zip. Για να το επιτύχετε αυτό, απλώς κάντε κλικ στο «download» κι ακολουθήστε τα βήματα.

3. Όταν το αρχείο σας κατέβει, πρέπει να εξάγετε το αρχείο .zip. Κάντε δεξί κλικ κι έπειτα κάντε κλικ στην εξαγωγή (extract).

4. Στο Raspberry Pi OS, ανοίξτε το **σύστημα αρχείων** (***filesystem)***. Επιλέξτε «***view»*** και στη συνέχεια **προβολή κρυφών αρχείων** (***show hidden files)***.

5. Βρείτε το **φάκελο** (***folder)*** του Minecraft, κάντε διπλό κλικ. Έπειτα διπλό κλικ ξανά στο φάκελο με το όνομα **«παιχνίδια»** (***"games")*** και στη συνέχεια ξανά διπλό κλικ

στο φάκελο με το όνομα ***"com.mojang"***. Εκεί θα βρείτε ένα φάκελο με την ονομασία ***"minecraftWorlds"***. Κάντε διπλό κλικ και είστε έτοιμοι να προσθέσετε νέους κόσμους. Απλώς σύρετε (drag) κι αποθέστε (drop) τα αποσυμπιεσμένα αρχεία των κόσμων σας στο φάκελο αυτό.

6. ***Χρήσιμη συμβουλή***: ενώ βρίσκεστε στο ευρετήριο κόσμων του Minecraft, μπορείτε να μετονομάσετε τους κόσμους που ήδη έχετε δημιουργήσει στο παιχνίδι, κάνοντας απλώς ***δεξί κλικ*** πάνω τους και στη συνέχεια πατώντας ***«Μετονομασία» (rename)***. Αυτός είναι ο μόνος τρόπος να μετονομάσετε τους κόσμους σας.

Θα βρείτε μια τεράστια ποικιλία χαρτών στους ακόλουθους συνδέσμους:

- https://thebraithwaites.co.uk/minecraft-pi-edition-maps-texture-packs-survival-and-more/
- https://www.minecraftforum.net/forums/minecraft-pocket-edition/mcpe-maps?page=2

Όσον αφορά τα βοηθητικά πακέτα, η διαδικασία είναι παρόμοια με την παραπάνω, αλλά χρειάζεται ένα διαφορετικό ευρετήριο.
Για παράδειγμα, μπορείτε να κατεβάσετε ένα πακέτο υφών (texture pack) από εδώ. Ακολουθήστε τις οδηγίες στο σύνδεσμο. Πρέπει να βρείτε το φάκελο "***Assets***" στο ευρετήριο του Minecraft και να τοποθετήσετε εκεί τα αποσυμπιεσμένα αρχεία. Φορτώστε ξανά το παιχνίδι και το πακέτο υφών θα είναι έτοιμο για χρήση.
Μπορείτε να βρείτε περισσότερα βοηθητικά πακέτα (resource packages), όπως επίσης και χάρτες, στους ακόλουθους συνδέσμους:

- https://www.planetminecraft.com/collection/10099/resource-packs/
- https://www.planetminecraft.com/texture_pack/the-pi-pack---132---037/
- https://thebraithwaites.co.uk/minecraft-pi-edition-maps-texture-packs-survival-and-more/
- https://www.minecraftforum.net/forums/minecraft-pocket-edition/mcpe-maps?page=2

5.3 Πρακτικές εφαρμογές

Για να δοκιμάσετε τις γνώσεις σας και να εξασκηθείτε περαιτέρω, σας προτείνουμε μερικές πρακτικές εφαρμογές. Αυτές συνιστάται να γίνουν μετά από κάθε μάθημα και περιλαμβάνουν τα ακόλουθα:

- Παράδειγμα 1: Παγιδέψτε τον παίκτη μεταξύ των μπλοκ!
- Παράδειγμα 2: Χτίστε ένα σπίτι!
- Παράδειγμα 3: Χτίστε ένα σπίτι με μια παραλλαγή!
- Παράδειγμα 4: Ένα Μεγάλο Μπλοκ από μπλοκ!
- Παράδειγμα 5: Δημιουργήστε ένα ηφαίστειο! (για προχωρημένους)

5.3.1 Παράδειγμα 1: Παγιδέψτε τον παίκτη μεταξύ των μπλοκ!

Ο παρακάτω κώδικας εντοπίζει τη θέση του παίκτη στο πλακίδιο, κι έπειτα καλεί τη λειτουργία `getBlock()` του Minecraft API για να βρει τον τύπο του μπλοκ στο οποίο στέκεται ο παίκτης (αφαιρώντας 1 από τη συντεταγμένη y), πριν να χρησιμοποιήσει το `setBlock()` για τη δημιουργία μπλοκ του ίδιου τύπου μ' αυτό στο οποίο στέκεται ο παίκτης γύρω του (https://www.stuffaboutcode.com/2013/04/minecraft-pi-edition-api-tutorial.html). Για παράδειγμα, αν ο παίκτης βρίσκεται πάνω σε ΠΕΤΡΑ (STONE), τότε μπλοκ ΠΕΤΡΑΣ θα εμφανιστούν γύρω του.

trap.py *

```
1  from mcpi.minecraft import minecraft
2  from mcpi.block import block
3  from time import sleep
4
5  mc = Minecraft.create()
6
7  playerTilePos = mc.player.getTilePos()
8  blockBelowPlayerType = mc.getBlock(playerTilePos.x, playerTilePos.y - 1, playerTilePos.z)
9  mc.setBlock(playerTilePos.x + 1, playerTilePos.y + 1, playerTilePos.z, blockBelowPlayerType)
10 mc.setBlock(playerTilePos.x, playerTilePos.y + 1, playerTilePos.z + 1, blockBelowPlayerType)
11 mc.setBlock(playerTilePos.x - 1, playerTilePos.y + 1, playerTilePos.z, blockBelowPlayerType)
12 mc.setBlock(playerTilePos.x, playerTilePos.y + 1, playerTilePos.z - 1, blockBelowPlayerType)
13 mc.postToChat("Trapped you")
14 time.sleep(5)
```

Χρησιμοποιήστε τον παραπάνω κώδικα, μαζί με ό,τι έχετε μάθει μέχρι τώρα, και γράψτε τον κώδικά σας. Έπειτα εκτελέστε τον κώδικα στο περιβάλλον προγραμματισμού της Python, ενώ εκτελείτε το Minecraft, και δείτε τι συμβαίνει. Είναι ο παίκτης σας παγιδευμένος μεταξύ μπλοκ του ίδιου τύπου με αυτό πάνω στο οποίο στέκεται; Αν όχι, τι πάει στραβά; Μπορείτε να βρείτε το λάθος; Αν όλα λειτουργούν σύμφωνα με την περιγραφή, τότε το επόμενο πράγμα που πρέπει να κάνετε είναι να ελευθερώσετε τον παίκτη σας. Προσπαθήστε να το κάνετε αυτό μόνοι/-ες σας!
**Βοήθεια: η αφαίρεση μπλοκ γίνεται χρησιμοποιώντας το* `setBlock()`*, αλλά αντί να κάνουμε το μπλοκ στερεό, όπως ΞΥΛΟ (WOOD) ή ΠΕΤΡΑ (STONE), το ορίζουμε σε ΑΕΡΑ (AIR).*

5.3.2 Παράδειγμα 2: Χτίστε ένα σπίτι!

Θέλετε να χτίσετε ένα σπίτι, αλλά δε θέλετε να ξοδέψετε ώρες χτίζοντας τα μπλοκ ένα-ένα. Ο ακόλουθος κώδικας θα δημιουργήσει ένα απλό κτίριο, που μοιάζει με σπίτι, από την αρχή. Ο κώδικας είναι αρκετά βασικός κι εξηγείται παρακάτω. Διάφορες γραμμές σχολίων έχουν εισαχθεί για τη διευκόλυνσή σας.

house.py *

```
1  from mcpi.minecraft import minecraft
2  from mcpi.block import block
3  from time import sleep
4
5  mc = Minecraft.create()
6
7  x, y, z = mc.player.getPos()
8
9  # build walls
10 mc.setBlocks(x+2, y-1, z+2, x+7, y+3, z+8, 5)
11 # remove interior
12 mc.setBlocks(x+3, y, z+3, x+6, y+2, z+7, 0)
13 # make doorway
14 mc.setBlocks(x+2, y, z+5, x+2, y+1, z+5, 0)
15 # make window 1
16 mc.setBlocks(x+4, y+1, z+8, x+5, y+1, z+8, 102)
17 # make window 2
18 mc.setBlocks(x+4, y+1, z+2, x+5, y+1, z+2, 102)
```

Πολύ απλό, σωστά; χρησιμοποιήστε τον παραπάνω κώδικα, μαζί με όσα έχετε μάθει μέχρι τώρα, και γράψτε τον κώδικά σας. Έπειτα εκτελέστε τον στο περιβάλλον προγραμματισμού της Python, ενώ παράλληλα εκτελείτε το Minecraft, και δείτε τι συμβαίνει. Τι θα λέγατε να προσθέσετε ένα δεύτερο όροφο και μια σκεπή; Χρησιμοποιήστε τη φαντασία και τις ικανότητες προγραμματισμού σας, μαζί με την εντολή `mc.setBlocks()` , για να τροποποιήσετε τον κώδικα και δείτε τι συμβαίνει!

5.3.3 Παράδειγμα 3: Χτίστε ένα σπίτι με μια παραλλαγή!

Μάθατε πώς να χτίσετε ένα σπίτι εκτελώντας τον κώδικά σας, αλλά τι θα λέγατε να χτίσετε όσα σπίτια θέλετε και όποτε τα θέλετε με το πάτημα ενός κουμπιού;
Μπορείτε να αναθέσετε σε κουμπί ηλεκτρονικού παιχνιδιού arcade να χτίζει σπίτια κάθε φορά που το πατάτε. Πώς;
Πρώτα απ' όλα, πρέπει να συνδέσετε ένα κουμπί στην πλακέτα GPIO, όπως μάθατε *(σιγουρευτείτε ότι συνδέετε τις σωστές ακίδες στην πλακέτα GPIO. Αυτό το παράδειγμα χρησιμοποιεί την GPIO24 ως ακίδα εισόδου, αλλά μπορείτε να χρησιμοποιήσετε οποιαδήποτε ανταποκρίνεται καλύτερα στις ανάγκες σας και να αλλάξετε τον κώδικα ανάλογα)*.
Έπειτα, ακολουθήστε τον παρακάτω κώδικα (Διάφορες γραμμές σχολίων έχουν εισαχθεί για τη διευκόλυνσή σας):

house_twist.py *

```python
#connect Python with Raspberry Pi GPIO board
import RPi.GPIO as GPIO

from mcpi.minecraft import minecraft
from mcpi.block import block
from time import sleep

mc = Minecraft.create()

GPIO.setmode(GPIO.BCM) #set mode for GPIO
GPIO.setup(24, GPIO.IN) #set input pin to GPIO24

while True:
    if GPIO.input(24) == True:
        x,y,z = mc.player.getPos() #get players position
        mc.setBlocks(x+2,y-1,z+2,x+7,y+3,z+8, 5) #make shell
        mc.setBlocks(x+3,y,z+3,x+6,y+2,z+7, 0) #remove inside
        mc.setBlocks(x+2,y,z+5,x+2,y+1,z+5, 0) #make doorway
        mc.setBlocks(x+4,y+1,z+8,x+5,y+1,z+8, 102) #make window 1
        mc.setBlocks(x+4,y+1,z+2,x+5,y+1,z+2, 102) #make window 2
        mc.setBlocks(x+7,y+1,z+4,x+7,y+1,z+6, 102) #make window 3
        print ("House is built")
        time.sleep(0.1) #wait 0.1 sec
```

Εκτελέστε τον κώδικά σας κι έπειτα πατήστε το καθορισμένο κουμπί. Ένα σπίτι θα πρέπει να εμφανιστεί δίπλα σας. Θέλετε να χτίσετε περισσότερα; Πατήστε το κουμπί πολλές φορές καθώς κινείστε στον κόσμο του Minecraft. Χρησιμοποιήστε τη φαντασία και τις ικανότητες προγραμματισμού σας και επεξεργαστείτε τις συντεταγμένες `mc.setBlocks()` , για να προσθέσετε περισσότερους ορόφους, να επεκτείνετε την έκταση της κατασκευής, να δημιουργήσετε περισσότερες πόρτες και παράθυρα, κ.λπ. Μπορείτε να προγραμματίσετε ένα κουμπί να χτίσει μια πολυκατοικία;

5.3.4 Παράδειγμα 4: Ένα Μεγάλο Μπλοκ από μπλοκ!

Άλλη μία άσκηση με τη χρήση της εντολής `mc.setBlocks()` . Μπορείτε να δημιουργήσετε τεράστια μπλοκ ενός συγκεκριμένου τύπου μπλοκ για περαιτέρω χρήση. Προσπαθήστε να χρησιμοποιήσετε τον ακόλουθο κώδικα στο σενάριό σας και δείτε τι γίνεται όταν «τρέξετε» το πρόγραμμά σας:

```
tnt = 46
mc.setBlocks(x+1, y+1, z+1, x+11, y+11, z+11, 46, 1)
```

Χρησιμοποιήστε τον παραπάνω κώδικα, μαζί με όσα έχετε μάθει μέχρι τώρα και γράψτε τον κώδικά σας. Έπειτα εκτελέστε τον κώδικα στο περιβάλλον προγραμματισμού της Python, ενώ παράλληλα εκτελείτε το Minecraft, και παρατηρείστε τι συμβαίνει. Λειτουργεί; Αν όχι, τι πάει λάθος; Αν όλα λειτουργούν σύμφωνα με την περιγραφή, τότε το επόμενο πράγμα που πρέπει να κάνετε είναι να αλλάξετε τους τύπους και τα σχήματα των μπλοκ και να χτίσετε τις μοναδικές σας κατασκευές. Προσπαθήστε να το κάνετε μόνοι/ες σας!

5.3.5 Παράδειγμα 5: Δημιουργήστε ένα Ηφαίστειο! (για προχωρημένους)

Ο ακόλουθος κώδικας θα σας δείξει πώς να δημιουργήσετε ένα ενεργό ηφαίστειο στον κόσμο του Minecraft από την αρχή. Επειδή αυτή η άσκηση είναι προχωρημένη σε σχέση με τις προηγούμενες, δίνεται ολόκληρος ο κώδικας παρακάτω (https://learnlearn.uk/raspberrypi/2018/02/10/minecraft-python-volcano-tutorial/). Διάφορες γραμμές σχολίων έχουν εισαχθεί για τη διευκόλυνσή σας.

volcano.py

```
# import Minecraft
import mcpi.minecraft as Minecraft
# import block library
import mcpi.block as block
# import random and time modules
import random, time

mc = Minecraft.create()
mc.postToChat("Minecraft Volcano!")

mc.setBlocks(-100, 0, -100, 100, 50, 100, block.AIR)

height = 20
center = 20,0,0  #x,y,z

for i in range(height):     # Create the base!
    size = height - i
    mc.setBlocks(center[0] - size, center[1] + i, center[2] - size, center[0] + size, center[1] + i, center[2] + size, block.STONE)

for i in range(height * height):
    randomx = random.randint(center[0] - height, center[0] + height)
    randomz = random.randint(center[2] - height, center[2] + height)
    mc.setBlock(randomx,center[1] + height + 10, randomz, block.GRAVEL)

while True:
    mc.setBlock(center[0],center[1]+height, center[2], block.LAVA_FLOWING)
    time.sleep(1)
```

Χρησιμοποιήστε τον παραπάνω κώδικα, μαζί με όσα έχετε μάθει μέχρι τώρα, και γράψτε τον κώδικά σας. Έπειτα εκτελέστε τον στο περιβάλλον προγραμματισμού της Python, ενώ παράλληλα εκτελείτε το Minecraft, και παρατηρήστε τι συμβαίνει. Λειτουργεί; Αν όχι, τι πάει λάθος; Αν όλα λειτουργούν βάσει της περιγραφής, τότε το επόμενο πράγμα που πρέπει να κάνετε είναι να κάνετε το ηφαίστειό σας να εκτοξεύσει νερό αντί για λάβα. Αν αυτό ήταν πολύ εύκολο, προσπαθήστε να δημιουργήσετε ένα κυκλικό ηφαίστειο αντί για ένα τετράγωνο.

5.4 Τεστ αξιολόγησης

1. Το Minecraft Pi υποστηρίζει πλήρως τα χαρακτηριστικά του παιχνιδιού Minecraft.
    1. Ναι
    2. **Όχι**
2. Μπορώ να «τρέξω» το Minecraft:
    1. Κάνοντας διπλό κλικ στο εικονίδιο στην επιφάνεια εργασίας
    2. Κάνοντας πλοήγηση στο κεντρικό μενού
    3. Χρησιμοποιώντας τη γραμμή εντολών για να «τρέξω» το πρόγραμμα
    4. **Όλα τα παραπάνω**
3. Το Minecraft Pi έχει ένα API για τον έλεγχο του παιχνιδιού με τη χρήση του περιβάλλοντος προγραμματισμού της Python κι ενός αριθμού σεναρίων.
    1. **Ναι**

2. Όχι

4. Ένα μπλοκ στο Minecraft είναι $1m^3$.
   1. Ναι
   2. Όχι
   3. **Όχι πάντα**
5. Τι κάνει η ακόλουθη γραμμή κώδικα: *from mcpi.minecraft import Minecraft ;*
   1. Ο κώδικας είναι λάθος
   2. **Συνδέει το περιβάλλον προγραμματισμού Python στο Minecraft**
   3. Αλλάζει την οπτική γωνία της κάμερας στο παιχνίδι
   4. Τίποτα απ' τα παραπάνω
6. Μπορούμε να τηλεμεταφέρουμε τον παίκτη μας με τη χρήση της εντολής *setPos()* .
   1. **Ναι**
   2. Όχι
7. Τι σημαίνει η ακόλουθη γραμμή κώδικα: *gold = 41 ;*
   1. Ορίζουμε την τιμή ενός αντικειμένου στις 41 μονάδες χρυσού
   2. Ορίζουμε 41 μπλοκ χρυσού να χρησιμοποιηθούν στο παιχνίδι
   3. **Το 41 αντιπροσωπεύει τον κωδικό αριθμό (ID) του χρυσού, αποθηκευμένο σε μια μεταβλητή με το όνομα «χρυσός»**
   4. Τίποτα απ' τα παραπάνω
8. Τι σημαίνει το τελευταίο ψηφίο στον ακόλουθο κώδικα: mc.setBlock(x, y, z, wool, **2**) ;
   1. **Επιπλέον ιδιότητα του τύπου μπλοκ μαλλί (wool)**
   2. Ζητούμε 2 μπλοκ μαλλιού
   3. Αντιπροσωπεύει κάποιο είδος συντεταγμένων
   4. Τίποτα απ' τα παραπάνω
9. Η γραμμή *GPIO.setup(23, GPIO.IN)* λέει στο Pi ότι αυτή η ακίδα 23 χρησιμοποιείται ως έξοδος.
   1. **Ναι**
   2. όχι
10. Για να δημιουργήσω ένα κύκλωμα για τη χρήση ενός κουμπιού με το Raspberry Pi, χρειάζομαι μια πλακέτα δοκιμών (breadboard), δύο καλώδια μεταγωγής και το κουμπί.
    1. Ναι
    2. **Όχι**

5.5 Παραπομπές

- Richardson, C., (2013), Minecraft Pi book, retrieved from https://arghbox.files.wordpress.com/2013/06/minecraftbook.pdf
- Minecraft controls, retrieved from https://arghbox.wordpress.com/2013/07/28/minecraft-pi-controls/
- Block ID numbers, retrieved from https://www.raspberrypi-spy.co.uk/2014/09/raspberry-pi-minecraft-block-id-number-reference/
- O'Hanlon, (2013), Minecraft: Pi Edition - API Tutoria, retrieved from https://www.stuffaboutcode.com/2013/04/minecraft-pi-edition-api-tutorial.html

- Special blocks in Minecraft, retrieved from https://minecraft.gamepedia.com/Block
- Minecraft Wiki, retrieved from https://minecraft.gamepedia.com/Pi_Edition

5.6 Επιπρόσθετες πηγές

Αν θέλετε περισσότερες λεπτομέρειες για το Minecraft Pi, μπορείτε να επισκεφτείτε τις παρακάτω πηγές:

- Minecraft API: https://www.stuffaboutcode.com/p/minecraft-api-reference.html
- Raspberry Pi: https://www.stuffaboutcode.com/p/raspberry-pi.html
- Minecraft Wiki: https://minecraft.gamepedia.com/Pi_Edition
- Manhattan project in Minecraft: https://www.stuffaboutcode.com/2013/04/minecraft-pi-edition-manhattan-stroll.html
- Massive Analogue Clock: https://www.stuffaboutcode.com/2013/02/raspberry-pi-minecraft-analogue-clock.html
- Planetary gravity simulation: https://www.stuffaboutcode.com/2013/03/raspberry-pi-minecraft-planetary.html
- Coding Tips: http://www.laschina.org/wp-content/uploads/2017/09/Minecraft-Coding-Tips.pdf
- Raspberry Pi projects: https://projects.raspberrypi.org/en/projects?software%5B%5D=python&hardware%5B%5D=raspberry-pi
- Minecraft Pi Handbook: http://repository.erasmusplus.website/RETROSTEM/Material/Minecraft%20Pi%20-%20Handbook.pdf

5.7 Συμπεράσματα

Βασικά συμπεράσματα για την **Ενότητα 3 – Εισαγωγή στο Minecraft Pi**:

Αν ακολουθήσατε αυτό το βοήθημα με το Raspberry Pi, αναμένεται να:

- Έχετε πρόσβαση στο Minecraft Pi και να δημιουργήσετε ένα νέο κόσμο.
- Πλοηγήστε στο Minecraft Pi με τη χρήση των ελέγχων κίνησης στο πληκτρολόγιό σας.
- Ξέρετε πώς να τοποθετήσετε και να καταστρέψετε ένα μπλοκ, και να πλοηγήστε μεταξύ των διαφορετικών τύπων μπλοκ στο ευρετήριο του παιχνιδιού.
- Συνδέετε την Python στο Minecraft Pi.
- Χρησιμοποιείτε το περιβάλλον προγραμματισμού της Python.
- Χειρίζεστε τα μπλοκ με τη χρήση κώδικα και σεναρίων της Python.
- Κατανοείτε επαρκώς τις υπόλοιπες λειτουργίες του Minecraft Pi.
- Κάνετε το Minecraft να αλληλεπιδρά με τον εξωτερικό κόσμο με τη χρήση κουμπιών και λυχνιών LED.
- Εισάγεται νέους κόσμους και να ανακαλύπτετε βοηθητικά πακέτα (resource packages) συμβατά με το Minecraft Pi.

6 Προγραμματισμός του Raspberry Pi GPIO με τη χρήση Python [P2-AKNOW]

6.1 Λεξικό όρων

Όρος / Έννοια	Ορισμός / Εξήγηση
Raspberry Pi GPIO	Το Raspberry Pi GPIO είναι η σειρά των ακίδων κατά μήκος της πάνω άκρης της πλακέτας. Μια κεφαλή 40 ακίδων βρίσκεται σε όλες τις σύγχρονες πλακέτες Raspberry Pi. Κατά βάση, η λειτουργικότητα του Raspberry Pi προέρχεται απ' αυτές τις ακίδες, που μπορούν να διαμορφωθούν και να ελεγχθούν με τη χρήση μιας γλώσσας προγραμματισμού. Καθεμία από τις ακίδες GPIO μπορεί να ορισθεί σε λογισμικό ως ακίδα εισόδου ή εξόδου και να χρησιμοποιηθεί για ένα ευρύ φάσμα στόχων, όπως ο έλεγχος λυχνιών LED, βομβητών (buzzer), μηχανών, σερβομηχανισμών, ή η αλληλεπίδραση με αισθητήρες, η επικοινωνία με άλλες συσκευές, κ.λπ.
Python	Η Python είναι μια διερμηνευόμενη, αντικειμενοστραφής, υψηλού επιπέδου γλώσσα προγραμματισμού. Η Python έχει απλό, εύκολο να μαθευτεί συντακτικό, που δίνει έμφαση στην αναγνωσιμότητα και, ως εκ τούτου, ελαττώνει το συνολικό χρόνο που χρειάζεται για να τη μάθετε και να αναπτύξετε και να διατηρήσετε ένα πρόγραμμα. Η Python υποστηρίζει προγραμματιστικά πακέτα, κάτι που ενθαρρύνει την αρθρωτότητα του προγράμματος και την επαναχρησιμοποίηση του κώδικα. Ο μεταγλωττιστής της Python και η εκτενής καθιερωμένη βιβλιοθήκη είναι διαθέσιμα σε source ή δυαδική μορφή, χωρίς χρέωση, για όλες τις γνωστές πλατφόρμες και μπορεί να διανεμηθεί ελεύθερα.

6.2 Κυρίως περιεχόμενο

Η ενότητα 4 – Προγραμματισμός του Raspberry Pi GPIO με τη χρήση της Python μας εισάγει στη χρήση των ακίδων Γενικής Χρήσης Εισόδου Εξόδου (General-Purpose Input Output pins - GPIO) στον υπολογιστή Raspberry Pi. Το GPIO είναι ένα ισχυρό χαρακτηριστικό του Raspberry Pi και μπορεί να βρεθεί στην πάνω άκρη της πλακέτας. Μια κεφαλή 40 ακίδων βρίσκεται σε όλους τους σύγχρονους υπολογιστές Raspberry Pi και το μεγαλύτερο μέρος της λειτουργικότητάς της προέρχεται από αυτές τις ακίδες. Μετά την ολοκλήρωση της ενότητας 4, ο μαθητής αναμένεται να ξέρει πώς να ελέγχει και ν' αλληλεπιδρά με το Raspberry Pi GPIO χρησιμοποιώντας τη γλώσσα προγραμματισμού Python.
Η ενότητα 4 αποτελείται από τα εξής μέρη:

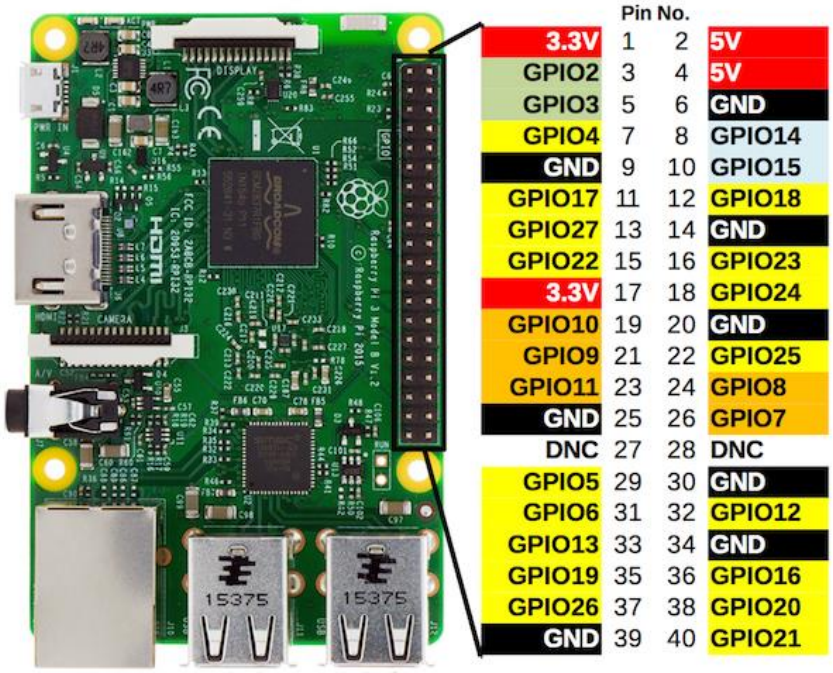
- Εισαγωγή στο Raspberry Pi GPIO
- Εισαγωγή στη γλώσσα προγραμματισμού Python
- Εισαγωγή στα ηλεκτρονικά κυκλώματα:
    - Αναβοσβήστε μια λυχνία LED με τη χρήση ακίδων Raspberry Pi GPIO και Python
    - Ενεργοποίηση ενός βομβητή (buzzer) μ' ένα πιεζόμενο κουμπί

- Σύνδεση και χρήση ενός αισθητήρα μέσω του GPIO
- Δημιουργία ενός συναγερμού εγγύτητας ή ενός φαναριού.

- Πρακτικές εφαρμογές της χρήσης του GPIO με τον προγραμματισμό Python
- Τεστ αξιολόγησης για να ελέγξετε τις αποκτηθείσες γνώσεις σας
- Επιπρόσθετες πηγές για να επεκτείνετε τις γνώσεις σας

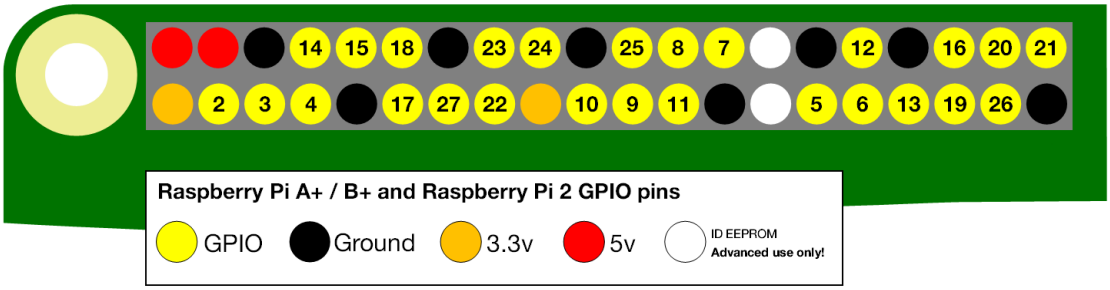
6.2.1 Εισαγωγή στις ακίδες Raspberry Pi GPIO

Το Raspberry Pi GPIO είναι η σειρά των ακίδων κατά μήκος της πάνω άκρης της πλακέτας. Μια κεφαλή 40 ακίδων βρίσκεται σε όλα τα σύγχρονα μοντέλα του Raspberry Pi. Το μεγαλύτερο μέρος της λειτουργικότητας του Raspberry Pi προέρχεται από το GPIO, το οποίο μπορεί να διαμορφωθεί και να ελεγχθεί με τη χρήση μιας γλώσσας προγραμματισμού. Η εικόνα 1 δείχνει πού βρίσκονται οι ακίδες GPIO σε ένα Raspberry Pi.



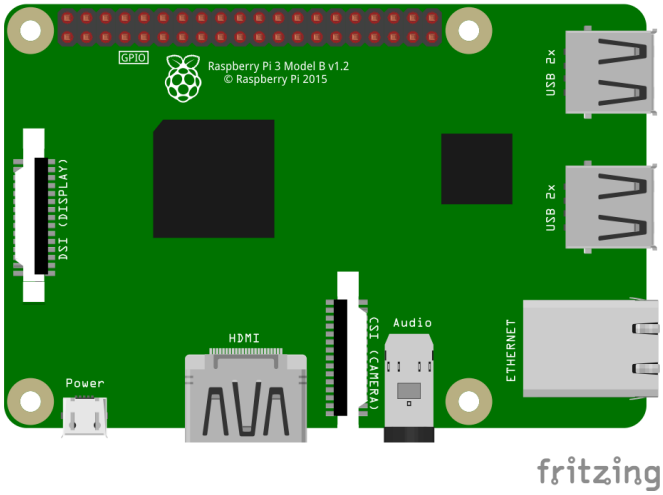
ΕΙΚΟΝΑ 21. ΑΚΙΔΕΣ RASPBERRY PI GPIO

Καθεμία από τις ακίδες GPIO μπορεί να ορισθεί σε λογισμικό ως ακίδα εισόδου ή εξόδου και να χρησιμοποιηθεί για ένα ευρύ φάσμα στόχων, όπως ο έλεγχος λυχνιών LED, βομβητών (buzzer), μηχανών, σερβομηχανισμών, ή η αλληλεπίδραση με αισθητήρες, η επικοινωνία με άλλες συσκευές, κ.λπ.
Ο χρήστης χρειάζεται να εξοικειωθεί με τις ακίδες GPIO, με το ποιες από αυτές μπορούν να χρησιμοποιηθούν, την ονομασία και την αρίθμησή τους, όπως και με το ποιες ιδιότητες κατέχουν. Η εικόνα 2 δείχνει τα καθιερωμένα ονόματα των ακίδων Broadcom (BCM). Προσέξτε ότι η αρίθμηση των ακίδων GPIO δεν είναι σε αριθμητική σειρά.



ΕΙΚΟΝΑ 22. ΚΑΘΙΕΡΩΜΕΝΑ ΟΝΟΜΑΤΑ ΑΚΙΔΩΝ BROADCOM (BCM)

Για να τις συγκρίνετε με το Raspberry Pi, προσανατολίστε τις ακίδες ώστε να βρίσκονται στην πάνω αριστερή πλευρά της πλακέτας, όπως φαίνεται στην εικόνα 3.



ΕΙΚΟΝΑ 23. ΣΧΗΜΑΤΙΚΗ ΚΑΤΟΨΗ ΤΟΥ ΥΠΟΛΟΓΙΣΤΗ RASPBERRY PI

Όλες αυτές οι χρωματισμένες με κίτρινο ακίδες είναι τυπικές ακίδες GPIO. Οι μαύρες είναι οι ακίδες γείωσης και οι υπόλοιπες είναι οι ακίδες τάσης στα 5V ή 3.3V. Η διάταξη των ακίδων δεν αλλάζει ποτέ, ανεξάρτητα της έκδοσης του Raspberry Pi model B που έχετε, αρχίζοντας από την κορυφή προς τα κάτω.

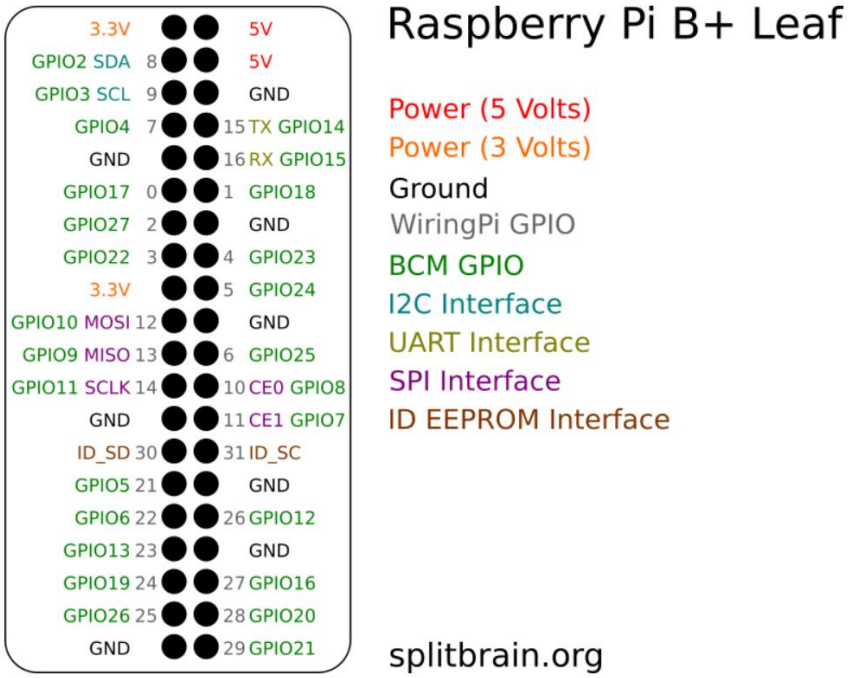
Ακίδες Τάσης και Γείωσης (Voltage and Ground pins): Δύο ακίδες 5V και δύο ακίδες 3.3V υπάρχουν στην πλακέτα, καθώς και ένας αριθμός ακίδων γείωσης (0V), που είναι μη διαμορφώσιμες. Οι ακίδες Γείωσης πρέπει να θεωρούνται από τις πιο σημαντικές ακίδες, επειδή οποιοδήποτε κύκλωμα που δημιουργείται πρέπει να συνδέεται σε μια απ' αυτές, διαφορετικά μπορεί να βλάψει την πλακέτα ή να έχει απρόσμενη συμπεριφορά. Οι εναπομένουσες ακίδες είναι όλες γενικής χρήσης 3.3V, που σημαίνει ότι οι έξοδοι είναι ρυθμισμένοι στα 3.3V και οι είσοδοι αναμένονται να είναι στα 3.3V.

Έξοδοι (Outputs): Μια ακίδα GPIO ορισμένη ως ακίδα εξόδου μπορεί να ρυθμιστεί σε υψηλή (3.3V) ή χαμηλή (0V).

Είσοδοι (Inputs): Μια ακίδα GPIO ορισμένη ως ακίδα εισόδου μπορεί να «διαβαστεί» ως υψηλή (3.3V) ή χαμηλή (0V).

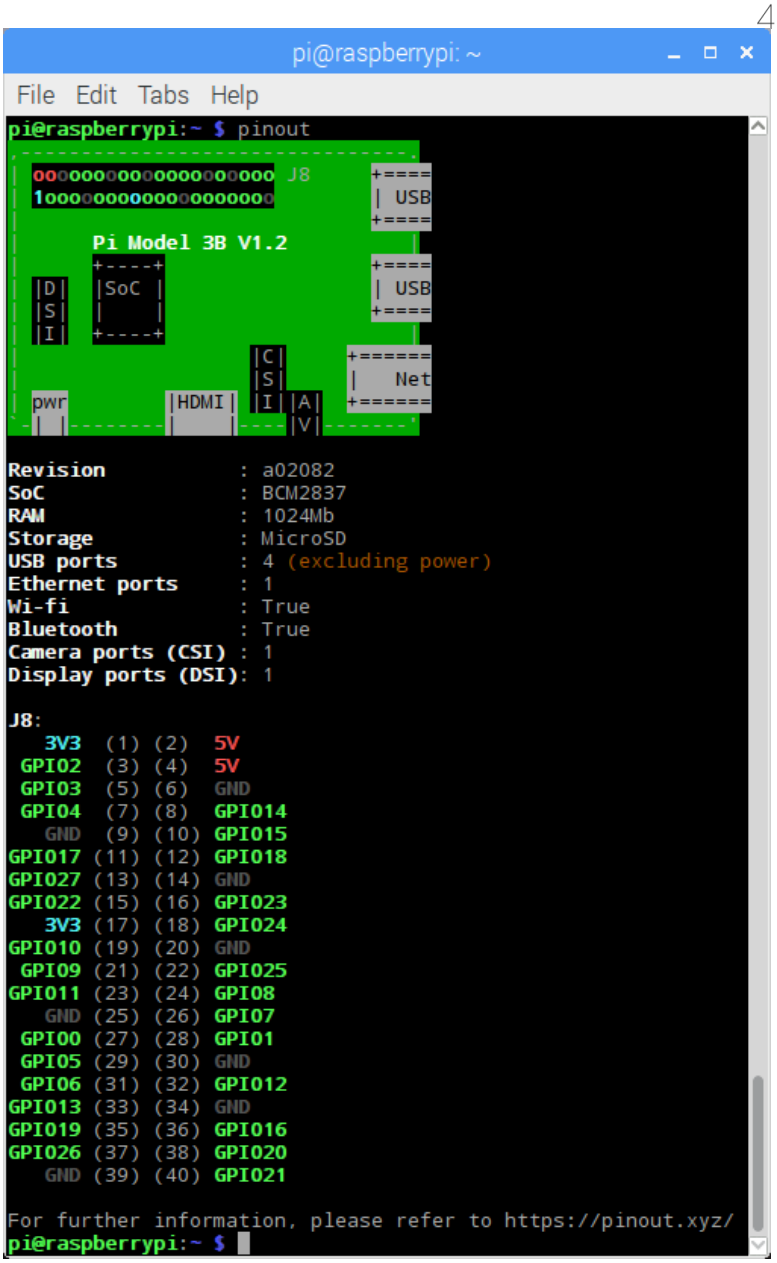
Οι ακίδες GPIO έχουν βασικά δύο καταστάσεις: ΥΨΗΛΗ (HIGH) ή ΧΑΜΗΛΗ (LOW), και, συνδυάζοντας αυτές τις δυαδικές επιλογές, πολλά περισσότερα αποτελέσματα μπορούν

να δημιουργηθούν με κυκλώματα που αλληλεπιδρούν με τον υλικό κόσμο με προγράμματα. Σε επόμενο κεφάλαιο, υψηλά και χαμηλά σήματα θα χρησιμοποιηθούν για να κάνουμε το Raspberry Pi να αλληλεπιδράσει με ηλεκτρονικά εξαρτήματα και αισθητήρες.
Είναι σημαντικό να ξεχωρίζουμε τις ακίδες GPIO. Κάποιοι άνθρωποι χρησιμοποιούν ετικέτες ακίδων, όπως για παράδειγμα το εκτυπώσιμο Raspberry Leaf (εικόνα 4).



ΕΙΚΟΝΑ 24. RASPBERRY PI B+ LEAF

Επιπλέον, μπορείτε να έχετε πρόσβαση σε μια εύχρηστη πηγή στο Raspberry Pi ανοίγοντας ένα παράθυρο τερματικού και εκτελώντας την εντολή `pinout` (Εικόνα 5). Αυτό το εργαλείο είναι προ-εγκατεστημένο στο εικονίδιο επιφάνειας εργασίας του Raspberry Pi OS και παρέχεται από τη βιβλιοθήκη GPIO Zero Python.



ΕΙΚΟΝΑ 25. ΤΟ ΕΡΓΑΛΕΙΟ PINOUT ΤΟΥ RASPBERRY PI

Είναι δυνατόν να ελέγξετε τις ακίδες GPIO χρησιμοποιώντας μια σειρά από γλώσσες κι εργαλεία προγραμματισμού. Στην ενότητα 4, θα μάθετε να χρησιμοποιείτε τη γλώσσα προγραμματισμού Python για να αλληλεπιδράτε με το GPIO του Raspberry Pi.

6.2.2 Εισαγωγή στη γλώσσα προγραμματισμού Python

Η Python είναι μια γενικής χρήσεως, αντικειμενοστραφής γλώσσα προγραμματισμού, εύκολη στην εκμάθηση, το διάβασμα και το γράψιμο, και με το Raspberry Pi, σας επιτρέπει να συνδέσετε το πρόγραμμά σας με τον υλικό κόσμο. Η Python έχει ένα πολύ ξεκάθαρο συντακτικό, εύκολο να το διαβάσετε, και χρησιμοποιείd τυπικές αγγλικές λέξεις-κλειδιά.

ΕΙΚΟΝΑ 26. ΤΟ ΛΟΓΟΤΥΠΟ ΤΗΣ PYTHON

Το Raspberry Pi έχει ένα προ-εγκατεστημένο περιβάλλον ανάπτυξης Python 3, που ονομάζεται Thonny. Μπορείτε να ανοίξετε το Thonny από το μενού της εφαρμογής στην επιφάνεια εργασίας του Raspberry Pi. Το Thonny είναι εξοπλισμένο με ένα REPL (Read-Evaluate-Print-Loop) το οποίο είναι ένα prompt στο οποίο μπορείτε να εισάγεται μια εντολή Python. Το Thonny διαθέτει ενσωματωμένη επισήμανση σύνταξης και μερική υποστήριξη αυτόματης συμπλήρωσης που διευκολύνει τη ζωή σας κατά την πληκτρολόγηση κώδικα.

Η Python υποστηρίζει τα εξής:

- Εκτύπωση (Print)
- Ενδοπαραγραφοποίηση (Indentation)
- Μεταβλητές (Variables)
- Σχόλια (Comments)
- Λίστες (Lists)
- Επαναλήψεις (Iterations)
- Εύρος (Range)
- Μήκος (Length)
- Δηλώσεις If (If statements)

```
print("Hello world")
```

ΕΙΚΟΝΑ 27. ΑΠΛΟ ΠΑΡΑΔΕΙΓΜΑ ΣΥΝΤΑΚΤΙΚΟΥ PYTHON

Λεπτομερείς πληροφορίες, για το πώς μπορεί να χρησιμοποιηθεί το καθένα απ' τα παραπάνω, είναι διαθέσιμες στους ακόλουθους συνδέσμους:

1. https://www.raspberrypi.org/documentation/usage/python/README.md
2. https://www.raspberrypi.org/documentation/usage/gpio/python/README.md

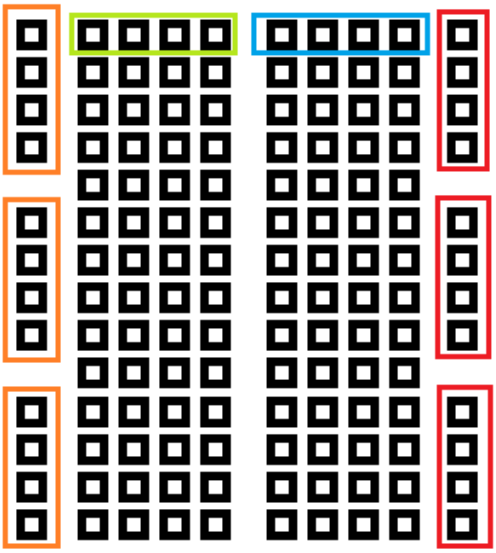
6.2.3 Εισαγωγή στα ηλεκτρονικά κυκλώματα

Μια σειρά από σύντομα μαθήματα παρουσιάζονται σ' αυτή την ενότητα. Εκτός από το Raspberry Pi, είναι απαραίτητα τα εξής εξαρτήματα:

- 10 x καλώδια μεταγωγής Αρσενικό σε Θηλυκό
- 1 x Πλακέτα δοκιμών/κυκλωμάτων (Breadboard)

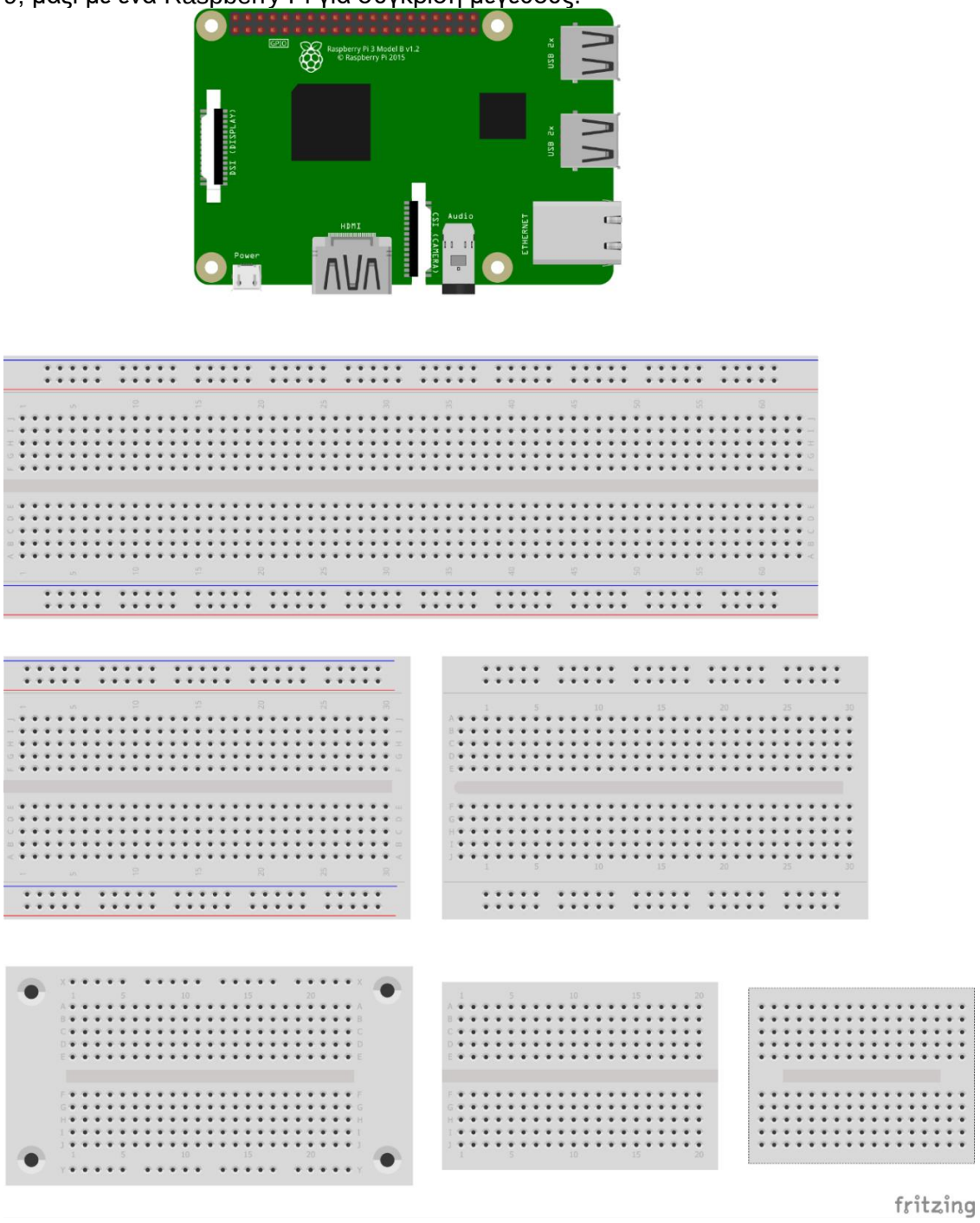
- 3 x λυχνίες LED
- 6 x Αντιστάτες (μεταξύ 300 και 1K Ohm). Θα χρειαστούμε 3x 1K Ohms για τον αισθητήρα απόστασης, και έναν αντιστάτη 300 με 1K Ohm για κάθε λυχνία LED
- 1 x Υπερηχητικό Αισθητήρα Απόστασης HC-SR04
- 1 x βομβητή (buzzer)

Ενώ η σύνδεση απλών εξαρτημάτων στις ακίδες GPIO είναι ασφαλής, είναι σημαντικό να είστε προσεκτικοί στο πώς συνδέετε τα καλώδια. Οι λυχνίες LED θα πρέπει να έχουν αντιστάσεις για τον περιορισμό του ρεύματος που τις διαπερνά. Η χρησιμοποιήσετε εξαρτήματα των 5V για εξαρτήματα των 3V3. Μη συνδέετε κινητήρες απευθείας στις ακίδες GPIO.
Οι λυχνίες LED καταναλώνουν διαφορετικά ποσά ενέργειας ανάλογα με το φως, κάτι που μπορεί να βλάψει το Raspberry Pi, αν είναι συνδεδεμένο χωρίς αντιστάτη που να ελαττώνει το ρεύμα. Πρέπει να ξεκινήσετε με έναν αντιστάτη μεγάλης αντίστασης και σταδιακά να την ελαττώσετε, μέχρι να είστε ικανοποιημένοι/ες με την ένταση του φωτός LED. Οι περισσότερες λυχνίες LED λειτουργούν μια χαρά με αντίσταση 300-1K Ohm. Όσο περισσότερη αντίσταση χρησιμοποιείτε, τόσο πιο αμυδρό θα είναι το φως του LED. Αν συνδέσετε μια λυχνία LED με μια αντίσταση 300 Ohm και δε δείτε καθόλου φως, τότε είτε η λάμπα είναι χαλασμένη είτε δεν έχει συνδεθεί σωστά. Δοκιμάστε ένα άλλο LED κι ελέγξτε αν είναι συνδεδεμένο σωστά, δηλαδή ότι ο μακρύτερος ακροδέκτης του είναι το θετικό άκρο και πρέπει να είναι συνδεδεμένο σε μια ακίδα GPIO, και ο κοντύτερος ακροδέκτης είναι το αρνητικό άκρο και πρέπει να συνδεθεί με τη γείωση του κυκλώματος.
Καταρχάς, αν δεν είστε εξοικειωμένοι με την πλακέτα δοκιμών, η βασική ιδέα είναι να σας βοηθήσουμε να δοκιμάσετε γρήγορα ένα κύκλωμα, να κάνετε πρωτότυπα, καθώς και να μάθετε σχετικά με την εργασία με κυκλώματα χωρίς να κάνετε καμία συγκόλληση (soldering). Μια τυπική πλακέτα δοκιμών μοιάζει κάπως έτσι:



ΕΙΚΟΝΑ28. ΣΧΗΜΑΤΙΚΗ ΚΑΤΟΨΗ ΜΙΑΣ ΤΥΠΙΚΗΣ ΠΛΑΚΕΤΑΣ ΔΟΚΙΜΩΝ (BREADBOARD)

Όπως φαίνεται στην εικόνα 8, οι σειρές κατά μήκος των άκρων είναι συνδεδεμένες κάθετα, και τα κεντρικά τμήματα είναι συνδεδεμένα οριζόντια. Αν υπάρχει μεγαλύτερο κενό, όπως στη μέση αυτής της πλακέτας, αυτό δε συνδέεται. Αν θέλετε να συνδεθούν αυτά τα τμήματα, πρέπει να τα συνδέσετε μ' ένα καλώδιο μεταγωγής.
Υπάρχουν πλακέτες δοκιμών σε διάφορα μεγέθη, που μπορούν να χρησιμοποιηθούν ανάλογα με το μέγεθος του κυκλώματος. Οι πιο συνηθισμένες παρουσιάζονται στην εικόνα 9, μαζί με ένα Raspberry Pi για σύγκριση μεγέθους.

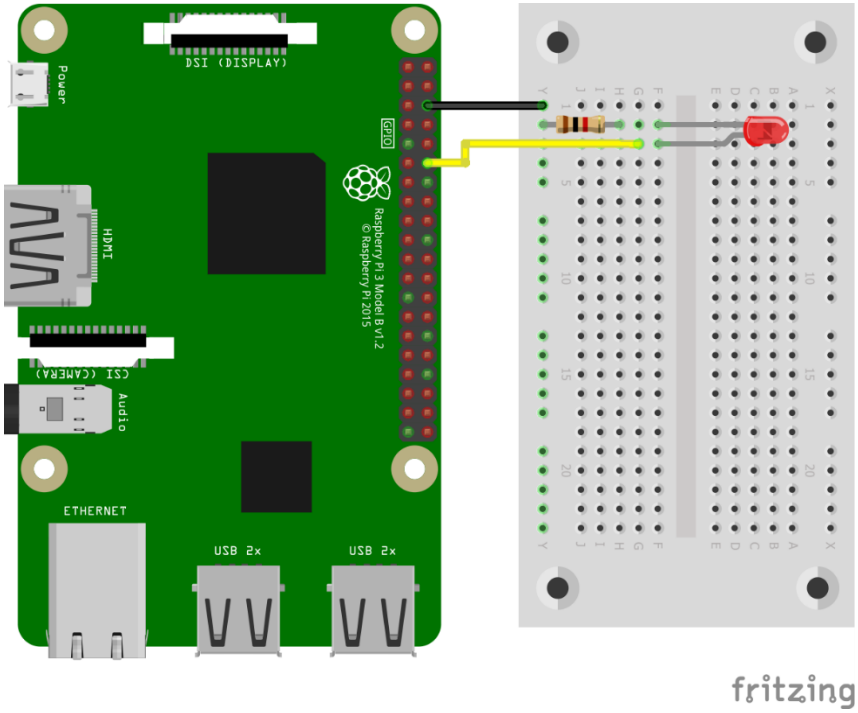


ΕΙΚΟΝΑ 29. ΣΧΗΜΑΤΙΚΗ ΚΑΤΟΨΗ ΔΙΑΦΟΡΕΤΙΚΩΝ ΠΛΑΚΕΤΩΝ ΚΥΚΛΩΜΑΤΩΝ (BREADBOARDS)

Εκτός απ' τα εξαρτήματα που ζητήθηκαν στην αρχή αυτής της ενότητας, θα χρειαστείτε επίσης ένα πληκτρολόγιο που μπορεί να συνδεθεί στις θύρες USB του Raspberry Pi σας.

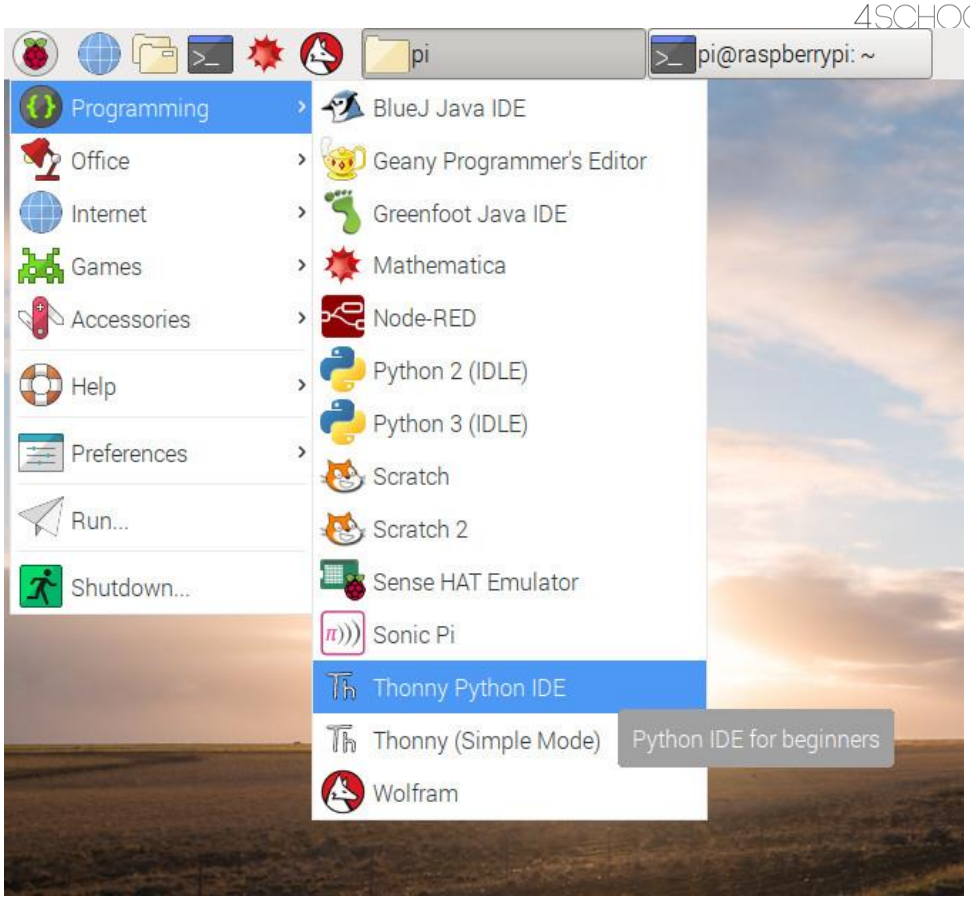
6.2.3.1 Αναβοσβήστε μια λυχνία LED

Πριν να προχωρήσετε, απενεργοποιήστε το Raspberry Pi και βγάλτε το από την πρίζα. Γι' αυτό το κύκλωμα θα χρειαστείτε μια πλακέτα δοκιμών, μια λυχνία LED, έναν αντιστάτη 220 Ohm και δύο καλώδια μεταγωγής θηλυκό σε αρσενικό. Το ολοκληρωμένο κύκλωμα παρουσιάζεται στο σχηματικό διάγραμμα της εικόνας 10.



ΕΙΚΟΝΑ 30. ΣΧΗΜΑΤΙΚΗ ΑΠΕΙΚΟΝΙΣΗ ΤΟΥ ΚΥΚΛΩΜΑΤΟΣ LED

Τώρα μπορείτε να ανοίξετε το Raspberry Pi, και θα ξεκινήσετε να αναπτύσσετε έναν απλό κώδικα στην Python. Στις περισσότερες περιπτώσεις η βιβλιοθήκη GPIO Python είναι ήδη εγκατεστημένη στο λειτουργικό σύστημα του Raspberry Pi. Αν όχι, τότε πρέπει να την εγκαταστήσετε. Για να το κάνετε αυτό, ανοίξτε ένα παράθυρο τερματικού (terminal) και πληκτρολογήστε τη γραμμή εντολής: `$ sudo apt-get install python-rpi.gpio` Από το πάνω αριστερά μενού της επιφάνειας εργασίας επιλέξτε «Προγραμματισμός» (Programming) κι έπειτα το Thonny Python (δείτε το στιγμιότυπο οθόνης που ακολουθεί).



ΕΙΚΟΝΑ 31. ΕΝΤΟΠΙΣΜΟΣ ΤΟΥ THONNY PYTHON

Πρώτα πρέπει να δημιουργήσετε ένα νέο αρχείο Python, κάνοντας κλικ στο **File (Αρχείο) → New File (Νέο Αρχείο)** κι αποθηκεύστε το με το όνομα `led.py`.
Στη συνέχεια, είστε έτοιμοι να αρχίσετε να πληκτρολογείτε τις πρώτες γραμμές του προγράμματός σας:

```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
```

Η αρίθμηση BCM είναι ο αριθμός ακίδας που το μικροκύκλωμα (τσιπ) Broadcom θεωρεί ότι είναι. Θα χρησιμοποιήσετε την αρίθμηση BCM, η οποία επίσης παρουσιάστηκε σε προηγούμενες εικόνες. Είναι χρήσιμο να την εκτυπώσετε και να την έχετε πάντα κάπου πρόχειρη, καθώς θα χρειαστεί να ελέγξετε ότι οι συνδέσεις στο κύκλωμά σας είναι σωστές κι ότι ανταποκρίνονται στον κώδικα που προγραμματίζετε.
Τώρα, συνεχίζοντας με τον κώδικά σας, ορίζετε:

```
GPIO.setup(18, GPIO.OUT)
```

Εδώ, ρυθμίζετε την ακίδα να είναι μια ακίδα εξόδου πληροφοριών. Μπορείτε επίσης να ρυθμίσετε την ακίδα να λαμβάνει πληροφορίες αντ' αυτού.

```
GPIO.output(18, GPIO.HIGH)
time.sleep(3)
```

Εδώ, λέτε στην ακίδα 18 να εξάγει ένα υψηλό σήμα, κι έπειτα να σταματήσει για 3 δευτερόλεπτα.

```
GPIO.output(18, GPIO.LOW)
GPIO.cleanup()
```

Τέλος, αλλάζετε την έξοδο της ακίδας σε ΧΑΜΗΛΟ σήμα, κι έπειτα χρησιμοποιείτε την εντολή `GPIO.cleanup()` για να επαναφέρετε τις καταστάσεις της ακίδας στις προκαθορισμένες, πριν τελειώσετε με το πρόγραμμα. Αυτό γίνεται ούτως ώστε το επόμενο πρόγραμμα που θα εκτελεστεί να έρθει σε μία κατάσταση που αναμένεται. Είναι αρκετά εύκολο να ξεχάσουμε και ν' αφήσουμε μερικές ακίδες σε διαφορετική κατάσταση και να ανακαλύψουμε αργότερα ότι συμπεριφέρονται περίεργα.
Το ολοκληρωμένο πρόγραμμα ακολουθεί παρακάτω. Επιπλέον γραμμές σχολίων έχουν εισαχθεί για τη διευκόλυνσή σας.

```
################################################

# first import libraries and set GPIO numbering mode
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)

# setup pin of LED as output and turn it on
GPIO.setup(18, GPIO.OUT)
GPIO.output(18, GPIO.HIGH)
# sleep for 3 seconds and then turn off LED
time.sleep(3)
GPIO.output(18, GPIO.LOW)

# cleanup gpio before exiting
GPIO.cleanup()
################################################
```

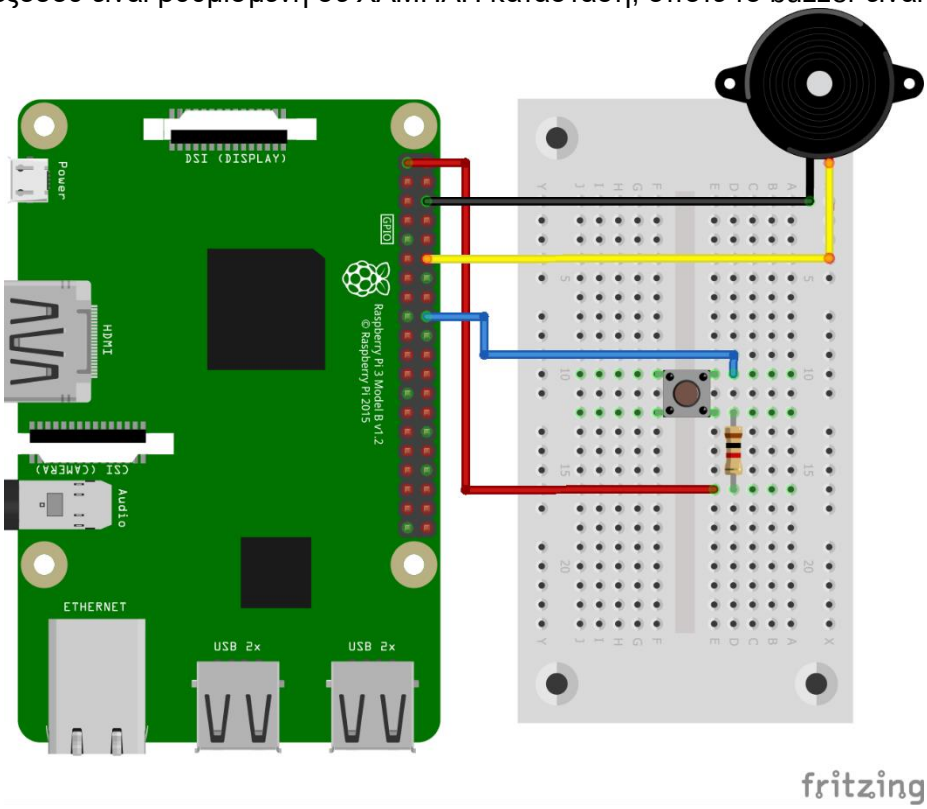
Τώρα είστε έτοιμοι/ες να «τρέξετε» το πρόγραμμά σας. Πρώτα αποθηκεύστε το πρόγραμμά σας κι έπειτα πιέστε το F5 για να εκτελέσετε τον κώδικα.
Όταν «τρέξει» το πρόγραμμα, η λυχνία LED θα πρέπει να ανάψει για τρία δευτερόλεπτα κι έπειτα να σβήσει. Μπορείτε να προσπαθήσετε να προσθέσετε περισσότερα LED στο κύκλωμά σας και να κωδικοποιήσετε ή να προσθέσετε ένα βομβητή (buzzer) που θα κάνει ήχο όταν ανάβουν τα LED. Μπορείτε επίσης να προσπαθήσετε να εξασκηθείτε σε απλές δομές της Python, όπως οι βρόχοι (loops) που θα κάνουν τα LED να αναβοσβήνουν αρκετές φορές.

6.2.3.2 Ενεργοποιήστε ένα βομβητή (buzzer)

Πριν να προχωρήσετε μ' αυτό το μάθημα, πρώτα θα προσπαθήσετε να χρησιμοποιήσετε μια ακίδα GPIO pin ως είσοδο. Θα χρειαστείτε ένα πιεζόμενο κουμπί κι ένα κύκλωμα, όπως φαίνεται στο σχηματικό διάγραμμα παρακάτω (εικόνα 12). Σ' αυτήν την περίπτωση, θα χρησιμοποιήσετε ένα βομβητή (buzzer) αντί για μια λυχνία LED, ο οποίος θα συνδεθεί με την ακίδα GPIO 18. Στο πρόγραμμά σας θα ρυθμίσετε αυτή την ακίδα ως έξοδο. Επίσης, θα συνδέσετε ένα πιεζόμενο κουμπί στην ακίδα GPIO 24, την οποία θα ρυθμίσετε ως είσοδο. Η άλλη άκρη του κουμπιού θα πρέπει να είναι συνδεδεμένη με την ακίδα τάσης 3.3V του Raspberry Pi μέσω ενός αντιστάτη. Έτσι, όταν πιέσετε το κουμπί, η ακίδα GPIO 24 θα είναι σε ΥΨΗΛΗ κατάσταση, δηλαδή σε 3.3V.

Όταν τελειώσετε με το κύκλωμά σας, μπορείτε να ανοίξετε το Raspberry Pi και ν' αρχίσετε να πληκτρολογείτε το πρόγραμμά σας σ' ένα νέο αρχείο Python (`button.py`). Ο ολοκληρωμένος κώδικας παρουσιάζεται παρακάτω, μαζί με μερικές γραμμές σχολίων για επεξηγήσεις.

Στο πρόγραμμά σας, πρώτα εισάγετε τα προγραμματιστικά πακέτα της Python που χρειάζεστε, κι έπειτα ρυθμίζετε τις ακίδες GPIO για το βομβητή και το κουμπί ως έξοδο κι είσοδο αντίστοιχα. Στη συνέχεια, απλώς για δοκιμή, ανάβετε το βομβητή για 3 δευτερόλεπτα. Στο τέλος, αρχίζετε μια λούπα που «τρέχει» συνεχώς. Στη λούπα ελέγχετε αν η ακίδα εισόδου είναι σε ΥΨΗΛΗ κατάσταση. Αν ναι, τότε αυτό σημαίνει ότι το κουμπί είναι πατημένο, κι έτσι, θέλετε να θέσετε την ακίδα εξόδου σε ΥΨΗΛΗ για να ανοίξει το buzzer. Διαφορετικά, η ακίδα εξόδου είναι ρυθμισμένη σε ΧΑΜΗΛΗ κατάσταση, οπότε το buzzer είναι κλειστό.



ΕΙΚΟΝΑ 32. ΣΧΗΜΑΤΙΚΟ ΔΙΑΓΡΑΜΜΑ ΚΥΚΛΩΜΑΤΟΣ ΜΕ ΠΙΕΖΟΜΕΝΟ ΚΟΥΜΠΙ ΚΑΙ ΒΟΜΒΗΤΗ ΣΥΝΔΕΔΕΜΕΝΑ ΣΤΙΣ ΑΚΙΔΕΣ GPIO

```
################################################
# first import modules and GPIO numbering mode
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.cleanup()
# pin for buzzer (or led)
pin_out = 18
GPIO.setup(pin_out, GPIO.OUT)
# pin for push button
pin_button = 24
GPIO.setup(pin_button, GPIO.IN)
```

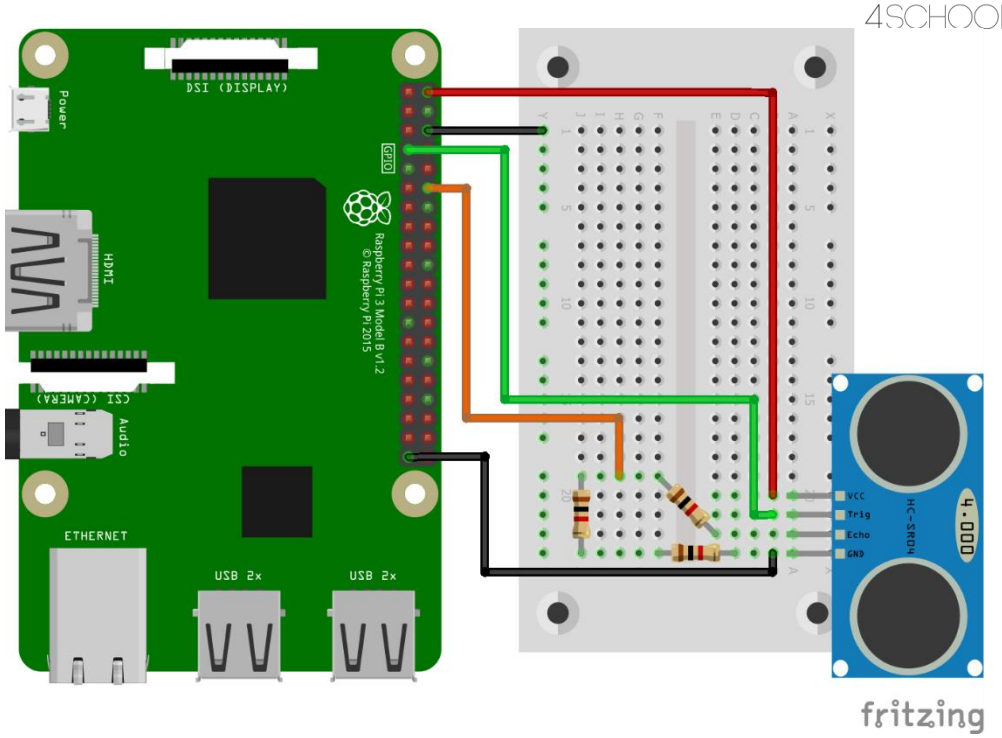
```
# turn on buzzer for 3 sec
GPIO.output(pin_out, GPIO.HIGH)
time.sleep(3)
GPIO.output(pin_out, GPIO.LOW)
# do this loop forever! Press Ctrl-C to stop it
# When button is pressed then buzzer is on
while True:
    if GPIO.input(pin_button) == GPIO.HIGH:
        print("button pushed")
        GPIO.output(pin_out, GPIO.HIGH)
    else :
        GPIO.output(pin_out, GPIO.LOW)
##############################################
```

6.2.3.3 Συνδέστε έναν αισθητήρα

Σ' αυτό το μάθημα, θα χρησιμοποιήσετε έναν αισθητήρα, τον υπερηχητικό αισθητήρα απόστασης HC-SR04, μαζί με την είσοδο GPIO.
Ο αισθητήρας απόστασης HC-SR04 μετράει την απόσταση εκπέμποντας έναν ηχητικό παλμό και χρονομετρώντας πόσο χρόνο παίρνει στην ηχώ να επιστρέψει. Χρησιμοποιώντας τη γνωστή σταθερά που είναι η ταχύτητα του ήχου, μπορούμε να προσδιορίσουμε με μαθηματική ακρίβεια την απόσταση οποιουδήποτε αντικειμένου μπροστά απ' αυτόν τον αισθητήρα, απλώς μετρώντας πόσος χρόνος πέρασε απ' τη στιγμή που εκπέμφθηκαν τα κύματα ήχου, προσέκρουσαν στο αντικείμενο μπροστά απ' τον αισθητήρα, αναπήδησαν πίσω, κι επέστρεψαν στον αισθητήρα.
Πριν να προχωρήσετε, απενεργοποιήστε το Raspberry Pi και βγάλτε το απ' την πρίζα. Για το κύκλωμα αυτού του μαθήματος θα χρειαστείτε:

- 1 Πλακέτα δοκιμών (Breadboard)
- 4x καλώδια μεταγωγής αρσενικό σε θηλυκό
- 1x 1K Ohm και 1x 2K Ohm αντιστάσεις ή 3x 1K Ohm αντιστάσεις
- 1x υπερηχητικό αισθητήρα απόστασης HC-SR04

Ο αισθητήρας απόστασης συνοδεύεται από 4 ακίδες: τροφοδοσία (power - VCC), σκανδαλισμός (trigger - TRIG), αντήχηση (echo - ECHO), και γείωση (ground - GND). Η ακίδα τροφοδοσίας θα συνδεθεί με την ακίδα 5V του Raspberry Pi, η ακίδα σκανδαλισμού θα αντιστοιχηθεί με την ακίδα GPIO ως έξοδος, η ακίδα αντήχησης θα αντιστοιχηθεί με μια ακίδα GPIO ως είσοδος, και, τέλος, η ακίδα γείωσης θα συνδεθεί με μια ακίδα γείωσης στο Raspberry Pi.
Η εικόνα 13 απεικονίζει το τελικό κύκλωμα.



ΕΙΚΟΝΑ 33. ΣΧΗΜΑΤΙΚΟ ΔΙΑΓΡΑΜΜΑ ΚΥΚΛΩΜΑΤΟΣ ΜΕ ΤΟΝ ΑΙΣΘΗΤΗΡΑ ΑΠΟΣΤΑΣΗΣ ΣΥΝΔΕΔΕΜΕΝΟ ΣΤΙΣ ΑΚΙΔΕΣ GPIO

Τώρα μπορείτε να ανοίξετε το Raspberry Pi, να δημιουργήσετε ένα νέο αρχείο Python και να ξεκινήσετε ένα νέο πρόγραμμα (`sensor.py`).

Όπως και στο προηγούμενο μάθημα, πρώτα εισάγετε τις ίδιες βιβλιοθήκες και την αρχική λειτουργία.

```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
```

Έπειτα ορίζετε τις ακίδες TRIG και ECHO ως έξοδο και είσοδο αντίστοιχα:

```
TRIG = 4
ECHO = 18
GPIO.setup(TRIG,GPIO.OUT)
GPIO.setup(ECHO,GPIO.IN)
```

Εδώ, ορίζετε τις ακίδες TRIG και ECHO ως τους αριθμούς ακίδων Broadcom που σκοπεύετε να χρησιμοποιήσετε γι' αυτό το κομμάτι του αισθητήρα. Αυτό γίνεται επειδή χρειάζεται να αναφέρετε και τις δύο ακίδες πολλές φορές.

Έπειτα, εκπέμπετε έναν υπερηχητικό παλμό απ' τον αισθητήρα με τα εξής:

```
GPIO.output(TRIG, True)
time.sleep(0.00001)
GPIO.output(TRIG, False)
```

Χρειάζεται να μετρήσετε το χρόνο που περνάει μέχρι να ανιχνευτεί η αντήχηση του παλμού απ' τον αισθητήρα:

```
while GPIO.input(ECHO) == False:
    tstart = time.time()
```

Μετά απ' αυτό, εκπέμπετε ένα σήμα κι έπειτα μετράτε το χρονικό διάστημα:

```
while GPIO.input(ECHO) == True:
    tend = time.time()
```

```
sig_time = tend-tstart
```

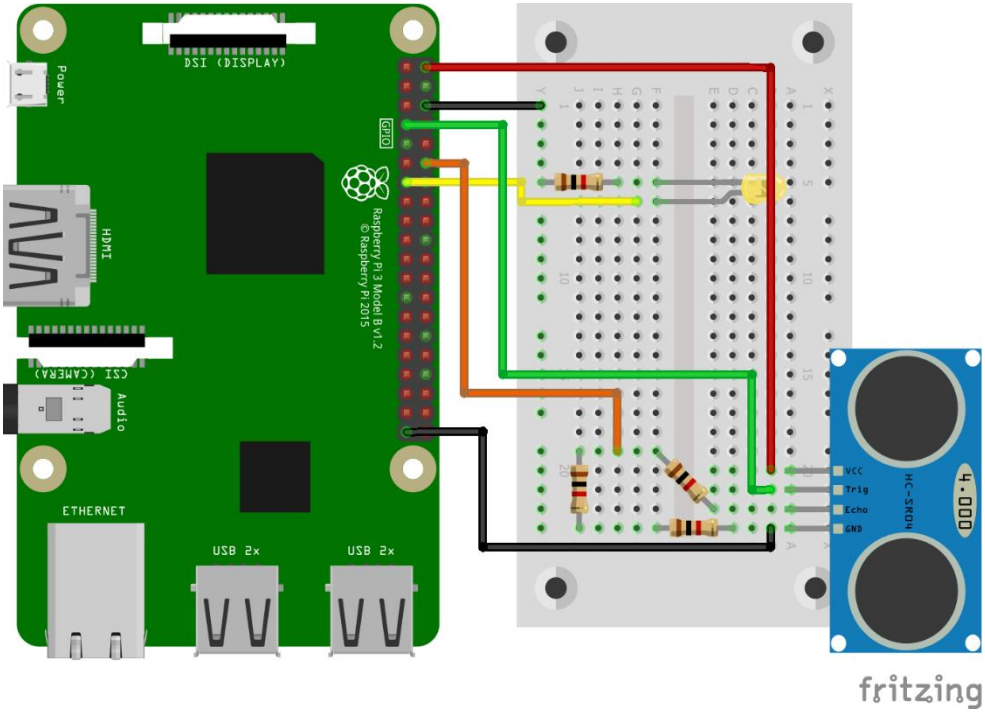
Όταν επιτέλους λάβετε ένα χρόνο εισόδου, μπορείτε να αφαιρέσετε τον τελικό χρόνο απ' τον αρχικό χρόνο και να υπολογίσετε την απόσταση πολλαπλασιάζοντας με την ταχύτητα του ήχου. Η ταχύτητα του ήχου σε ξηρό αέρα στους 20 °C είναι 343.26 m/sec.
Προσέξτε επίσης ότι πρέπει να πολλαπλασιάσετε με ½, διότι ο ηχητικός παλμός στην πραγματικότητα διανύει την απόσταση μέχρι το αντικείμενο ή το εμπόδιο δύο φορές (δηλ. εκπέμπεται απ' τον αισθητήρα, ταξιδεύει μέχρι να ανακλαστεί σ' ένα αντικείμενο και ταξιδεύει πίσω στον αισθητήρα για ν' ανιχνευθεί):

```
distance = sig_time * 34326.* 0.5
print("signal(sec), distance(cm):", sig_time, distance)
GPIO.cleanup()
```

Μπορείτε να τυπώσετε την απόσταση, κι έπειτα να επαναφέρετε τις ακίδες.
Το αποτέλεσμά σας πρέπει να είναι κάπως έτσι: `signal(sec), distance(cm): 0.001 17.163`
Προσέξτε επίσης, ότι, μετά από περίπου 20 μοίρες κλίσης, η απόστασή σας ίσως είναι πολύ στρεβλωμένη, καθώς η αρχική ανάκλαση μπορεί να αστοχήσει και ο παλμός να χρειαστεί αρκετές ανακλάσεις σε αντικείμενα μέχρι να επιστρέψει στον αισθητήρα.
Το ολοκληρωμένο πρόγραμμα κώδικα γι' αυτό το μάθημα ακολουθεί παρακάτω. Μερικές γραμμές σχολίων έχουν εισαχθεί για διευκόλυνση και σαφήνεια.

```
################################################
# first import libraries and set GPIO numbering mode
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)

# define TRIG and ECHO pins and set them up as output and input
TRIG = 4
ECHO = 18
GPIO.setup(TRIG,GPIO.OUT)
GPIO.setup(ECHO,GPIO.IN)
# emit a burst of ultrasound
GPIO.output(TRIG, True)
time.sleep(0.00001)
GPIO.output(TRIG, False)

# measure time interval
while GPIO.input(ECHO) == False:
    tstart = time.time()
while GPIO.input(ECHO) == True:
    tend = time.time()
sig_time = tend-tstart

# calculate distance and print value
distance = sig_time * 34326.* 0.5
print("signal(sec), distance(cm):", sig_time, distance)

# cleanup gpio before exiting
GPIO.cleanup()
################################################
```

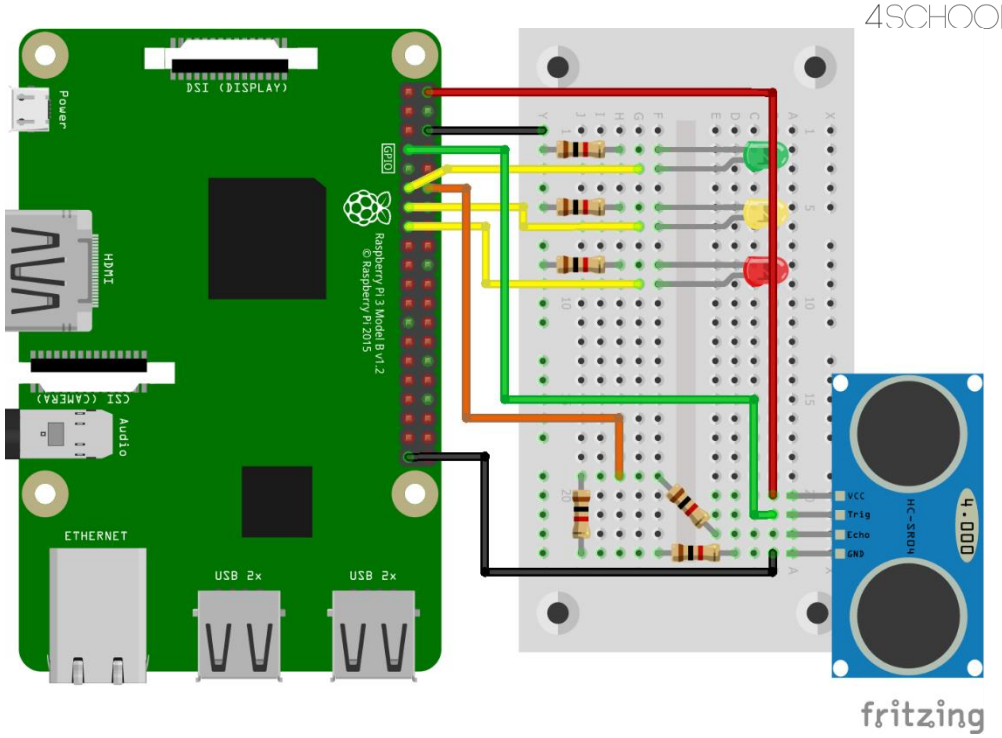
Μπορείτε να συνεχίσετε συνδυάζοντας κυκλώματα και κώδικες απ' αυτό και το προηγούμενο μάθημα, ώστε να προσθέσετε ένα ή περισσότερα LED (π.χ. εικόνα 14 παρακάτω).



ΕΙΚΟΝΑ 34. ΣΧΗΜΑΤΙΚΟ ΔΙΑΓΡΑΜΜΑ ΚΥΚΛΩΜΑΤΟΣ ΜΕ LED ΚΑΙ ΑΙΣΘΗΤΗΡΑ ΑΠΟΣΤΑΣΗΣ ΣΥΝΔΕΔΕΜΕΝΑ ΣΕ ΑΚΙΔΕΣ GPIO

6.2.3.4 Δημιουργήστε ένα συναγερμό εγγύτητας

Σ' αυτό το μάθημα, θα συνδυάσουμε ό,τι έχουμε μάθει μέχρι τώρα για να δημιουργήσουμε ένα συναγερμό εγγύτητας ή ένα φως φρένων, όπως αυτά που έχουν τα αυτοκίνητα για να βοηθήσουν τους οδηγούς στο παρκάρισμα.
Η κεντρική ιδέα ενός συναγερμού εγγύτητας ή ενός φωτός φρένων είναι να δείχνει πράσινο όταν υπάρχει ακόμα αρκετός χώρος, να γίνεται κίτρινο καθώς η απόσταση μικραίνει, κι έπειτα κόκκινο όταν η απόσταση είναι πια ελάχιστη, δηλ. όταν το όχημα θα πρέπει να σταματήσει. Θα κατασκευάσουμε αυτό το σύστημα συνδυάζοντας τα κυκλώματα και τον κώδικα που αναπτύξαμε στα προηγούμενα μαθήματα.
Καταρχάς, απλώς αφήνουμε το κύκλωμα αισθητήρα απόστασης όπως είναι και προσθέτουμε τα κυκλώματα για τις τρεις λυχνίες LED. Η εικόνα 15 απεικονίζει το ολοκληρωμένο κύκλωμα.



ΕΙΚΟΝΑ 35. ΣΧΗΜΑΤΙΚΟ ΔΙΑΓΡΑΜΜΑ ΚΥΚΛΩΜΑΤΟΣ ΜΕ ΑΙΣΘΗΤΗΡΑ ΑΠΟΣΤΑΣΗΣ ΚΑΙ ΠΟΛΛΑΠΛΑ LED ΣΥΝΔΕΔΕΜΕΝΑ ΣΕ ΑΚΙΔΕΣ GPIO

Αφού τα συνδέσουμε όλα, συστήνεται να ελέγξουμε ξανά το κύκλωμα πριν να ανοίξουμε το Raspberry Pi. Στη συνέχεια, πρέπει να τροποποιήσετε τον προηγούμενο κώδικα αισθητήρα απόστασης, να εισάγετε ένα συνεχόμενο βρόχο και να μετατρέψετε ορισμένα κομμάτια του κώδικα σε λειτουργίες. Μπορείτε να ξεκινήσετε αντιγράφοντας κι επικολλώντας τον υπάρχοντα κώδικα απ' το προηγούμενο μάθημα σε ένα νέο αρχείο Python (`proximity_alarm.py`). Ο νέος κώδικας θα πρέπει να μοιάζει κάπως έτσι:

```
import RPi.GPIO as GPIO
import time
GPIO.setwarnings(False)
# doing this first, since using a while True.
GPIO.cleanup()
GPIO.setmode(GPIO.BCM)
TRIG = 4
ECHO = 18
GPIO.setup(TRIG,GPIO.OUT)
GPIO.setup(ECHO,GPIO.IN)

def get_distance():
    GPIO.output(TRIG, True)
    time.sleep(0.00001)
    GPIO.output(TRIG, False)
    while GPIO.input(ECHO) == False:
        tstart = time.time()
    while GPIO.input(ECHO) == True:
        tend = time.time()
```

```
    sig_time = tend-tstart
    # Distance in cm
    distance = sig_time * 34326.* 0.5
    #print("signal(sec), distance(cm):", sig_time, distance)
    return distance

while True:
    distance = get_distance()
    time.sleep(0.05)
```

Προσέξτε ότι έχετε μεταφέρει το μεγαλύτερο μέρος του κώδικα στη συνάρτηση `get_distance()` κι έπειτα την καλούμε σ' ένα συνεχόμενο βρόχο while. Επίσης, παρατηρείστε ότι πρέπει να συμπεριλάβετε κάποιο χρόνο αναμονής (sleep time) μεταξύ των εκτελέσεων της συνάρτησης, διαφορετικά ίσως ο αισθητήρας να μη συμπεριφέρεται φυσιολογικά. Αν δεν τον αφήσετε να «ξεκουραστεί» για λίγο, θα υπερφορτωθεί και θα κλείσει. Τώρα χρειάζεστε απλώς έναν απλό κώδικα που ανάβει τα φώτα LED. Για παράδειγμα, ένα πράσινο φως, με την προϋπόθεση ότι συνδέσατε την πρίζα για το πράσινο φως στην ακίδα BCM 17:

```
GREEN = 17
GPIO.setup(GREEN,GPIO.OUT)
def green_light():
    GPIO.output(GREEN, GPIO.HIGH)
```

Θα μπορούσατε να αντιγράψετε και να επικολλήσετε το παραπάνω και για τις άλλες λυχνίες LED, όμως παρατηρείστε ότι τίποτα δεν τις σβήνει αν αλλάξουμε φώτα (χρώματα), οπότε στη συνάρτηση `green_light()` χρειάζεστε κάτι σαν αυτό:

```
def green_light():
    GPIO.output(GREEN, GPIO.HIGH)
    GPIO.output(YELLOW, GPIO.LOW)
    GPIO.output(RED, GPIO.LOW)
```

Τώρα μπορείτε να κάνετε απλώς αντιγραφή κι επικόλληση για όλα τα φώτα:

```
GREEN = 17
YELLOW = 27
RED = 22
GPIO.setup(GREEN,GPIO.OUT)
GPIO.setup(YELLOW,GPIO.OUT)
GPIO.setup(RED,GPIO.OUT)
def green_light():
    GPIO.output(GREEN, GPIO.HIGH)
    GPIO.output(YELLOW, GPIO.LOW)
    GPIO.output(RED, GPIO.LOW)
def yellow_light():
    GPIO.output(GREEN, GPIO.LOW)
    GPIO.output(YELLOW, GPIO.HIGH)
    GPIO.output(RED, GPIO.LOW)
def red_light():
    GPIO.output(GREEN, GPIO.LOW)
```

```
    GPIO.output(YELLOW, GPIO.LOW)
    GPIO.output(RED, GPIO.HIGH)
```

Έπειτα ο βρόχος while θα μοιάζει κάπως έτσι:

```
while True:
    distance = get_distance()
    time.sleep(0.05)
    print(distance)
    if distance >= 30:
        green_light()
    elif 30 > distance > 10:
        yellow_light()
    elif distance <= 10:
        red_light()
```

Επομένως, αν η απόσταση είναι μεγαλύτερη ή ίση με 30εκ., ανάβει το πράσινο φως. Αν είναι μεταξύ 10 και 30εκ., ανάβει το κίτρινο φως, και τέλος το κόκκινο φως ανάβει για απόσταση μικρότερη ή ίση των 10 εκ.
Ο ολοκληρωμένος κώδικας παρουσιάζεται παρακάτω. Αρκετές γραμμές σχολίων έχουν προστεθεί για διευκόλυνση και σαφήνεια.

```
################################################
# first import libraries and set GPIO numbering mode
import RPi.GPIO as GPIO
import time
GPIO.setwarnings(False)
# doing this first, since we're using a while True.
GPIO.cleanup()
GPIO.setmode(GPIO.BCM)

# define TRIG and ECHO pins and set them up as output and input
TRIG = 4
ECHO = 18
GPIO.setup(TRIG,GPIO.OUT)
GPIO.setup(ECHO,GPIO.IN)
# function to get distance value
def get_distance():
    # emit a burst of ultrasound
    GPIO.output(TRIG, True)
    time.sleep(0.00001)
    GPIO.output(TRIG, False)
    # measure time interval
    while GPIO.input(ECHO) == False:
        tstart = time.time()
    while GPIO.input(ECHO) == True:
        tend = time.time()
    sig_time = tend-tstart
    # Distance in cm
    distance = sig_time * 34326.* 0.5
    #print("signal(sec), distance(cm):", sig_time, distance)
```

```
    return distance

# define pins for LEDs and setup them as outputs
GREEN = 17
YELLOW = 27
RED = 22
GPIO.setup(GREEN,GPIO.OUT)
GPIO.setup(YELLOW,GPIO.OUT)
GPIO.setup(RED,GPIO.OUT)
# function to turn on green, turn off yellow and red
def green_light():
    GPIO.output(GREEN, GPIO.HIGH)
    GPIO.output(YELLOW, GPIO.LOW)
    GPIO.output(RED, GPIO.LOW)
# function to turn on yellow, turn off green and red def yel-
low_light():
    GPIO.output(GREEN, GPIO.LOW)
    GPIO.output(YELLOW, GPIO.HIGH)
    GPIO.output(RED, GPIO.LOW)
# function to turn on red, turn off yellow and green
def red_light():
    GPIO.output(GREEN, GPIO.LOW)
    GPIO.output(YELLOW, GPIO.LOW)
    GPIO.output(RED, GPIO.HIGH)

# loop that runs forever
while True:
    distance = get_distance()
    time.sleep(0.05)
    print(distance)
    # check distance and turn on light accordingly
    if distance >= 30:
        green_light()
    elif 30 > distance > 10:
        yellow_light()
    elif distance <= 10:
        red_light()
##############################################
```

Όταν «τρέξετε» αυτό το πρόγραμμα, η συσκευή σας σε κατάσταση λειτουργίας θα υποδεικνύει ότι υπάρχει αρκετός χώρος (πράσινο φως), ότι πλησιάζουμε (κίτρινο φως), ότι πρέπει να σταματήσουμε (κόκκινο φως).

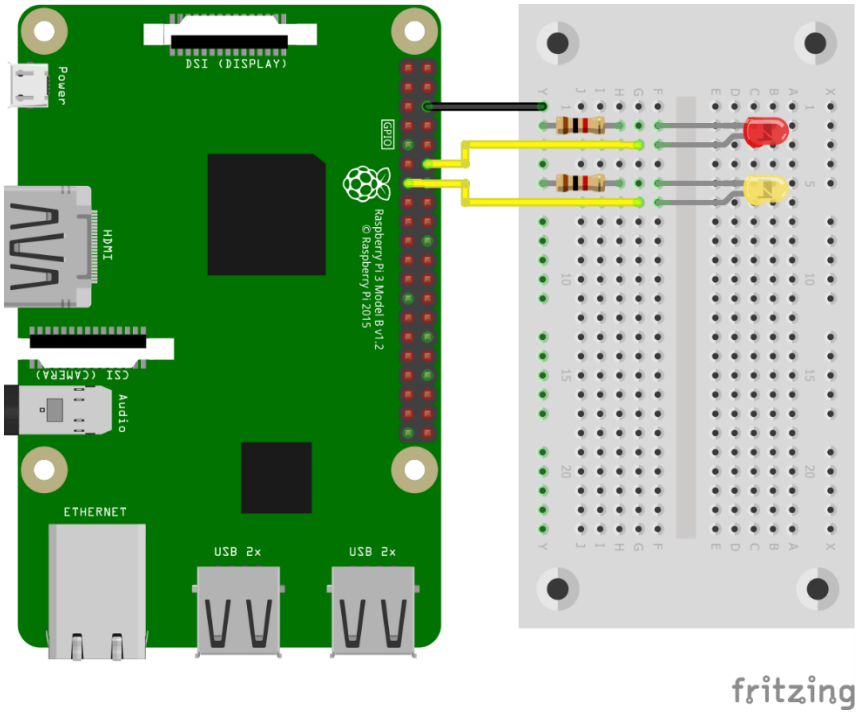
6.3 Πρακτικές εφαρμογές

Για να ελέγξετε τις γνώσεις σας και να εξασκηθείτε περαιτέρω, σας προτείνουμε μερικές πρακτικές εφαρμογές. Αυτές συνίστανται να γίνουν μετά από κάθε μάθημα και περιλαμβάνουν τα εξής:

- Παράδειγμα 1: Χρησιμοποιήστε ένα Βομβητή (Buzzer) και Πολλαπλές λυχνίες LED με ακίδες GPIO του Raspberry Pi
- Παράδειγμα 2: Μετρήστε την Ταχύτητα του Ήχου
- Παράδειγμα 3: Φτιάξτε ένα Συναγερμό Εγγύτητας με Φως και Ήχο

6.3.1 Παράδειγμα 1: Χρησιμοποιήστε ένα Βομβητή (Buzzer) και Πολλαπλές λυχνίες LED με ακίδες GPIO του Raspberry Pi

Σ' αυτήν την άσκηση, προσπαθήστε να τροποποιήσετε το κύκλωμα και τον κώδικα που αναπτύξατε στο πρώτο μάθημα ούτως ώστε να συνδέσετε και να χρησιμοποιήσετε πολλαπλές λυχνίες LED, αλλά και να τις προγραμματίσετε να αναβοσβήνουν με κάποιο μοτίβο (π.χ. όταν η μία είναι αναμμένη η άλλη να είναι σβηστή ή να αναβοσβήνουν σε διαφορετικά χρονικά διαστήματα). Γι' αυτή την άσκηση, φτιάξτε ένα κύκλωμα σαν αυτό που φαίνεται στην εικόνα 16. Μπορείτε επίσης να συνδέσετε ένα βομβητή να βγάζει έναν ήχο.



ΕΙΚΟΝΑ 36. ΣΧΗΜΑΤΙΚΟ ΔΙΑΓΡΑΜΜΑ ΚΥΚΛΩΜΑΤΟΣ ΜΕ ΠΟΛΛΑΠΛΑ LED ΣΥΝΔΕΔΕΜΕΝΑ ΣΕ ΑΚΙΔΕΣ GPIO

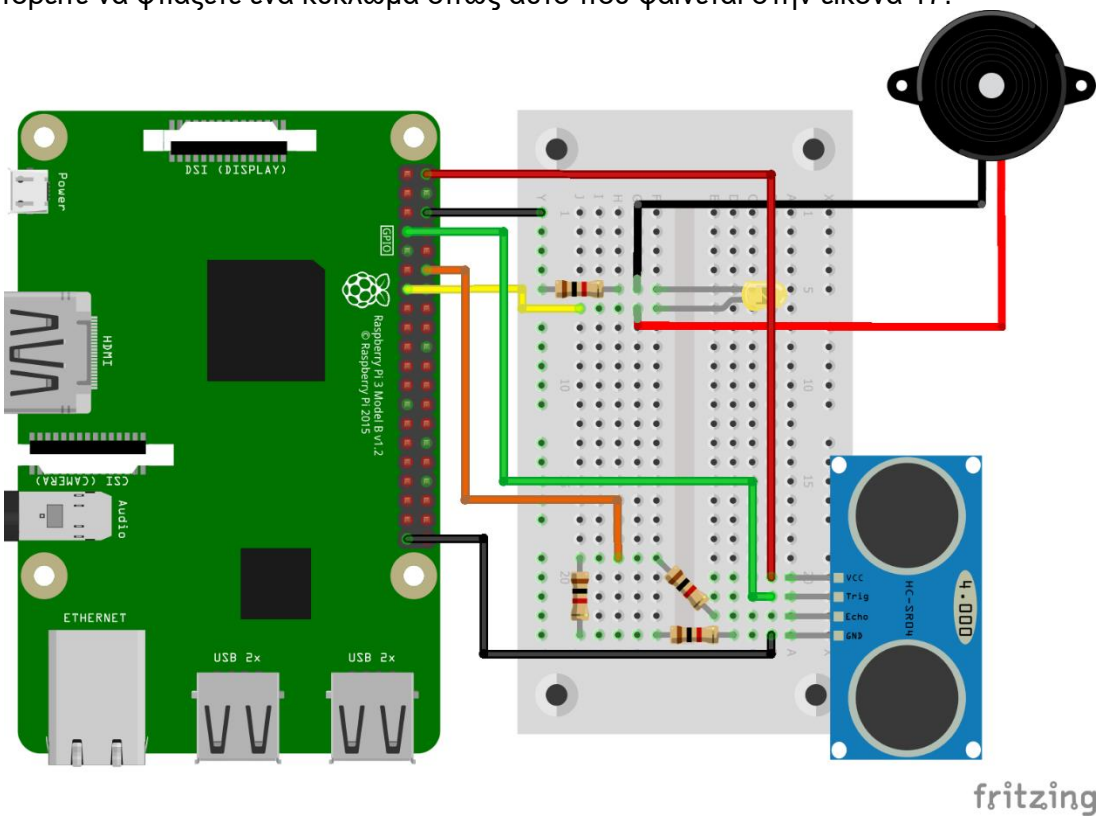
6.3.2 Παράδειγμα 2: Μετρήστε την Ταχύτητα του Ήχου

Σ' αυτήν την άσκηση, προσπαθήστε να μετρήσετε την ταχύτητα του ήχου χρησιμοποιώντας τον αισθητήρα και το κύκλωμα του δεύτερου μαθήματος. Θα χρειαστεί να τροποποιήσετε τον κώδικα, ώστε να τυπώνει το χρόνο του σήματος αντί για την υπολογιζόμενη απόσταση. Μετρήστε αρκετές γνωστές αποστάσεις μεταξύ του χεριού σας, του εμποδίου ή του αντικειμένου και του αισθητήρα. Καταγράψτε ζεύγη τιμών αποστάσεων (π.χ. που

μετρήσατε με ένα χάρακα) και μετρήσεων χρόνου, κι έπειτα υπολογίστε μόνοι/ες σας την ταχύτητα του ήχου.

6.3.3 Παράδειγμα 3: Φτιάξτε ένα Συναγερμό Εγγύτητας με Φως και Ήχο

Σ' αυτήν την άσκηση, προσπαθήστε να προσθέσετε ένα βομβητή (buzzer) στο συναγερμό σας απ' το Μάθημα 3, ώστε να βγάζει ήχο ανάλογα με την απόσταση. Γι' αυτή την άσκηση μπορείτε να φτιάξετε ένα κύκλωμα όπως αυτό που φαίνεται στην εικόνα 17.



ΕΙΚΟΝΑ 37. ΣΧΗΜΑΤΙΚΟ ΔΙΑΓΡΑΜΜΑ ΚΥΚΛΩΜΑΤΟΣ ΜΕ ΑΙΣΘΗΤΗΡΑ ΑΠΟΣΤΑΣΗΣ, ΛΥΧΝΙΕΣ LED ΚΑΙ ΒΟΜΒΗΤΗ ΣΥΝΔΕΔΕΜΕΝΑ ΣΕ ΑΚΙΔΕΣ GPIO

6.4 Τεστ αξιολόγησης

1. Μπορούμε να ρυθμίσουμε μια ακίδα GPIO ως:
    a. Είσοδο
    b. Έξοδο
    c. Είσοδο ή Έξοδο
2. Πρέπει να απενεργοποιήσετε το Raspberry Pi πριν να χρησιμοποιήσετε τις ακίδες GPIO του.
    a. Σωστό
    b. Λάθος
3. Σε τι κατάσταση μπορεί να βρίσκεται μια ακίδα GPIO;
    a. Υψηλή
    b. Υψηλή ή Χαμηλή
    c. Χαμηλή
4. Το GPIO του Raspberry Pi έχει ακίδες τάσης 3.3V και 10V.
    a. Σωστό
    b. Λάθος
5. Μια απ' τις πιο σημαντικές ακίδες της κεφαλίδας 40 ακίδων είναι:
    a. Οι ακίδες γείωσης
    b. Οι ακίδες τάσης
    c. Οι ακίδες GPIO
6. Με τη χρήση της γλώσσας προγραμματισμού Python και τις ακίδες GPIO μπορείτε να αλληλεπιδράσετε με τον υλικό κόσμο.
    a. Σωστό
    b. Λάθος
7. Με τη χρήση της Python στο Raspberry Pi μπορούμε να προγραμματίσουμε τις:
    a. ακίδες γείωσης
    b. ακίδες τάσης
    c. ακίδες GPIO
8. Για τη σύνδεση του GPIO με μια πλακέτα δοκιμών χρειαζόμαστε καλώδια μεταγωγής αρσενικού σε θηλυκό.
    a. Σωστό
    b. Λάθος
9. Η Python είναι:
    a. Μια γλώσσα προγραμματισμού γενικής χρήσης
    b. Μια γλώσσα προγραμματισμού μόνο για το Raspberry Pi
    c. Ένα φίδι με πόδια
10. Το Raspberry Pi είναι ένας πλήρως λειτουργικός υπολογιστής σε μέγεθος πιστωτικής κάρτας.
    a. Σωστό
    b. Λάθος

6.5 Παραπομπές

- https://www.raspberrypi.org/documentation/usage/gpio/
- https://github.com/splitbrain/rpibplusleaf
- https://www.raspberrypi.org/documentation/usage/gpio/python/README.md
- https://www.raspberrypi.org/documentation/usage/python/README.md
- https://pythonprogramming.net/gpio-raspberry-pi-tutorials/

6.6 Επιπλέον πηγές

- Raspberry Pi GPIO: https://www.raspberrypi.org/documentation/usage/gpio/
- Εμπράγματος Προγραμματισμός (Physical Computing) με την Python: https://projects.raspberrypi.org/en/projects/physical-computing
- Ξεκινώντας με τη γλώσσα προγραμματισμού Python και το Raspberry Pi: https://raspberrypihq.com/getting-started-with-python-programming-and-the-raspberry-pi/
- Το Σύντομο Μάθημα για τηνPython: https://docs.python.org/3/tutorial/
- Το περιοδικό MagPi: https://www.raspberrypi.org/magpi/

6.7 Συμπεράσματα

Βασικά συμπεράσματα για την **Ενότητα 4 – Προγραμματισμός του Raspberry Pi με τη χρήση της Python**:
Αφού διαβάσετε αυτό το κεφάλαιο, αναμένεται να έχετε μάθει, καταλάβει και εξασκηθεί στα ακόλουθα μαθησιακά στοιχεία:

- Τι είναι μια ακίδα Raspberry Pi GPIO και τη βασική της λειτουργικότητα.
- Πώς να φτιάχνετε βασικά κυκλώματα με λυχνίες LED, βομβητές (buzzer), αισθητήρες, καθώς και πώς να τα συνδέετε αυτά στις ακίδες Raspberry Pi GPIO.
- Πώς να εγκαταστήσετε, να ελέγχετε και να προγραμματίζετε τις ακίδες GPIO με τη χρήση της γλώσσας προγραμματισμού Python.
- Εξοικείωση με και αξιοποίηση των βασικών δομών και συντακτικού της γλώσσας προγραμματισμού Python, συμπεριλαμβανομένων των λειτουργιών, των μεταβλητών, των βρόχων for, των βρόχων while, και των δηλώσεων if.

7 Εμπράγματος Προγραμματισμός (Physical Computing) [P1-ECAM]

7.1 Λεξικό όρων

Όρος / Έννοια	Ορισμός / Εξήγηση
Εμπράγματος Προγραμματισμός (Physical computing)	Ο εμπράγματος προγραμματισμός σημαίνει αλληλεπίδραση με αντικείμενα πραγματικού κόσμου προγραμματίζοντάς τα από έναν υπολογιστή. Στα παραδείγματα περιλαμβάνεται ο προγραμματισμός ενός LED για φλας, η ανάγνωση περιβαλλοντικών δεδομένων από έναν αισθητήρα ή ακόμη και ο έλεγχος ρομποτικών αντικειμένων. Εφαρμογές όπως αυτές βρίσκονται γύρω μας στην καθημερινή ζωή, από σήματα κυκλοφορίας και εμπόδια εισιτηρίων έως αυτοκίνητα χωρίς οδηγό και γραμμές συναρμολόγησης. Πίσω από καθεμία από αυτές τις εφαρμογές υπάρχουν αλγόριθμοι και προγράμματα που διέπουν τη συμπεριφορά τους. Ο Εμπράγματος Προγραμματισμός συνδυάζει υλικό και λογισμικό για να δημιουργήσει κάτι χρήσιμο ή παραγωγικό, ή απλά για διασκέδαση.
Εφαρμογή (Application)	Μια εφαρμογή έχει σχεδιαστεί για να εκτελεί μια συγκεκριμένη εργασία διαφορετική από αυτήν που σχετίζεται με τη λειτουργία του ίδιου του υπολογιστή. Έχει σχεδιαστεί για να βοηθά τους ανθρώπους να εκτελούν μια δραστηριότητα.
API	Το API είναι το αρκτικόλεξο για τη διεπαφή προγραμματισμού εφαρμογών, το οποίο είναι ένας διαμεσολαβητής λογισμικού που επιτρέπει σε δύο εφαρμογές να μιλούν μεταξύ τους.
Περιφερειακή συσκευή (Peripheral)	Οποιαδήποτε συσκευή είναι συνδεδεμένη με υπολογιστή αλλά όχι μέρος αυτού.
Προγραμματιστής (Programmer)	Ο προγραμματιστής, ο προγραμματιστής συσκευών, ο προγραμματιστής τσιπ, ο καυστήρας συσκευών ή ο συγγραφέας PROM είναι ένα κομμάτι ηλεκτρονικού εξοπλισμού που τακτοποιεί γραπτό λογισμικό για διαμόρφωση.

7.2 Κυρίως περιεχόμενο

Σε αυτό το κεφάλαιο περιγράφονται τα καθοριστικά χαρακτηριστικά του Εμπράγματου Προγραμματισμού. Θα προσδιοριστούν οι βασικές ικανότητες που θα αποκτηθούν με τον Εμπράγματο Προγραμματισμό.

7.2.1 Εισαγωγή στον Εμπράγματο Προγραμματισμό

«Εμπράγματος Προγραμματισμός: Αίσθηση και έλεγχος του Υλικού κόσμου με υπολογιστές» (T.Igoe, Tisch NYU).

Πρόκειται για ένα πρόσφατο κίνημα που απαιτεί δεξιότητες στην Ηλεκτρονική, τον Προγραμματισμό, την Επεξεργασία Δεδομένων, τη Φυσική, ακόμα και στο Σχεδιασμό. Ο στόχος του Εμπράγματου Προγραμματισμού είναι να δημιουργήσει νέες συσκευές σε αλληλεπίδραση με τον πραγματικό κόσμο.
Σ' αυτό το κεφάλαιο θα παρουσιαστεί η έννοια του Εμπράγματου Προγραμματισμού και οι συμμετέχοντες θα διδαχθούν τις θεωρητικές και πρακτικές διαστάσεις του.
Αυτό το κεφάλαιο είναι δομημένο με παραδείγματα, όπως η ανάπτυξη ενός αυτόνομου ευκίνητου ρομπότ, ικανό να κινηθεί χάρη στους διαφορετικούς εγκατεστημένους αισθητήρες και σύμφωνα με κάποιους κανόνες.

Ορισμός σύμφωνα με τη Wikipedia, την ελεύθερη εγκυκλοπαίδεια:
«Ο Εμπράγματος Προγραμματισμός περιλαμβάνει διαδραστικά συστήματα, που μπορούν να διαισθανθούν και να ανταποκριθούν με το περιβάλλον γύρω τους. Ενώ αυτός ο ορισμός είναι αρκετά ευρύς, ώστε να περιλαμβάνει συστήματα, όπως τα έξυπνα συστήματα ελέγχου κυκλοφορίας αυτοκινήτων ή τις διαδικασίες αυτοματισμού εργοστασίων, δε χρησιμοποιείται συχνά για να τα περιγράψει. Με την ευρύτερη έννοια, ο εμπράγματος προγραμματισμός είναι ένα δημιουργικό πλαίσιο για την κατανόηση της σχέσης των ανθρώπων με τον ψηφιακό κόσμο. Στην πράξη, ο όρος αυτός συχνότερα περιγράφει χειροποίητα έργα τέχνης, σχεδιασμού ή DIY χόμπι, τα οποία χρησιμοποιούν αισθητήρες και μικροελεγκτές για τη μετάφραση αναλογικής εισόδου σε ένα σύστημα λογισμικού, ή/και τον έλεγχο ηλεκτρομηχανικών συσκευών, όπως κινητήρων, σερβομηχανισμών, φωτισμού ή άλλου υλικού».

Τι είναι ο Εμπράγματος Προγραμματισμός;
Ο Εμπράγματος Προγραμματισμός είναι μια προσέγγιση για την εκμάθηση του πώς επικοινωνούν οι άνθρωποι μέσω των υπολογιστών, ξεκινώντας αναλογιζόμενος του πώς εκφράζονται οι άνθρωποι σωματικά. Πολλές από τις αρχικές οδηγίες σχεδιασμού διεπαφής υπολογιστή παίρνουν δεδομένο το υλικό υπολογιστή — δηλαδή, ότι υπάρχει ένα πληκτρολόγιο, μια οθόνη, ίσως ηχεία, κι ένα ποντίκι — και συγκεντρώνονται στην εκμάθηση του απαραίτητου λογισμικού για το σχεδιασμό αυτών των ορίων. Στον εμπράγματο προγραμματισμό, θεωρούμε το ανθρώπινο σώμα ως δεδομένο κι επιχειρούμε να σχεδιάσουμε εντός των ορίων της έκφρασής του.

Αυτό σημαίνει ότι πρέπει να μάθουμε πώς μετατρέπει ένας υπολογιστής τις αλλαγές της εκπεμπόμενης απ' τα σώματά μας ενέργειας, υπό τη μορφή της θερμότητας, του φωτός, του ήχου κ.λπ., σε μεταβαλλόμενα ηλεκτρονικά σήματα που μπορεί να ερμηνεύσει. Μαθαίνουμε για τους αισθητήρες που το κάνουν αυτό και για πολύ απλούς υπολογιστές, που ονομάζονται μικρο-ελεγκτές, οι οποίοι «διαβάζουν» τους αισθητήρες και μετατρέπουν τα αποτελέσματά τους σε δεδομένα. Τέλος, μαθαίνουμε πώς οι μικρο-ελεγκτές επικοινωνούν με άλλους υπολογιστές.

Ο εμπράγματος προγραμματισμός συνεπάγεται τη δημιουργία ή τη χρήση συσκευών που αλληλεπιδρούν με τον κόσμο γύρω τους. Ένας φυσικός υπολογιστής διαισθάνεται το περιβάλλον του, επεξεργάζεται αυτήν την πληροφορία, κι έπειτα εκτελεί κάποια ενέργεια. Αυτός ο κύκλος «αίσθησης - σκέψης – πράξης» μπορεί επίσης να χρησιμοποιηθεί για να ορίσει ένα ρομπότ. Στις Τεχνολογίες BirdBrain χρησιμοποιούμε τους όρους «εμπράγματος προγραμματισμός» και «ρομποτική» χωρίς διάκριση.
Η δημιουργία ενός ρομπότ περιλαμβάνει μηχανική ή κατασκευή. Έπειτα, χρειάζεται επίσης να γραφτεί ένα πρόγραμμα υπολογιστή για να βάλει το ρομπότ να κάνει κάτι ενδιαφέ-

ρον! Τα ρομπότ δε μοιάζουν πάντα σαν αυτά που δείχνουν στις ταινίες. Κάποια είναι ανθρωποειδή ή ρομποτικά αυτοκίνητα, αλλά μπορεί κανείς να φτιάξει και ρομποτικά κατοικίδια ή κήπους. Δεν υπάρχουν όρια! Τα ρομπότ παρέχουν στους μαθητές έναν τρόπο να δουν τον κώδικά τους να λειτουργεί στον κόσμο.

Ο εμπράγματος προγραμματισμός καλύπτει το σχεδιασμό και την υλοποίηση διαδραστικών αντικειμένων και εγκαταστάσεων, κι επιτρέπει στους μαθητές ν' αναπτύξουν ξεκάθαρα, χειροπιαστά προϊόντα του πραγματικού κόσμου, που προκύπτουν απ' την φαντασία τους. Μ' αυτόν τον τρόπο, η κονστρουκτιβιστική μάθηση καλλιεργείται σε επίπεδο που επιτρέπει στους μαθητές να αποκτήσουν απτική εμπειρία και, κατ' αυτόν τον τρόπο, συγκεκριμενοποιεί το εικονικό.

Ο εμπράγματος προγραμματισμός υιοθετεί μια ενεργή προσέγγιση, που σημαίνει ότι περνάει κανείς πολύ χρόνο φτιάχνοντας κυκλώματα, συγκολλώντας, γράφοντας προγράμματα, φτιάχνοντας κατασκευές που συγκρατούν αισθητήρες και συστήματα ελέγχου, καθώς και ανακαλύπτοντας πώς να συσχετίσει όλα αυτά όσο το δυνατόν καλύτερα με την έκφραση ενός ατόμου.

Τι χρειάζεται να ξέρετε πριν παρακολουθήσετε ένα μάθημα εμπράγματου προγραμματισμού;

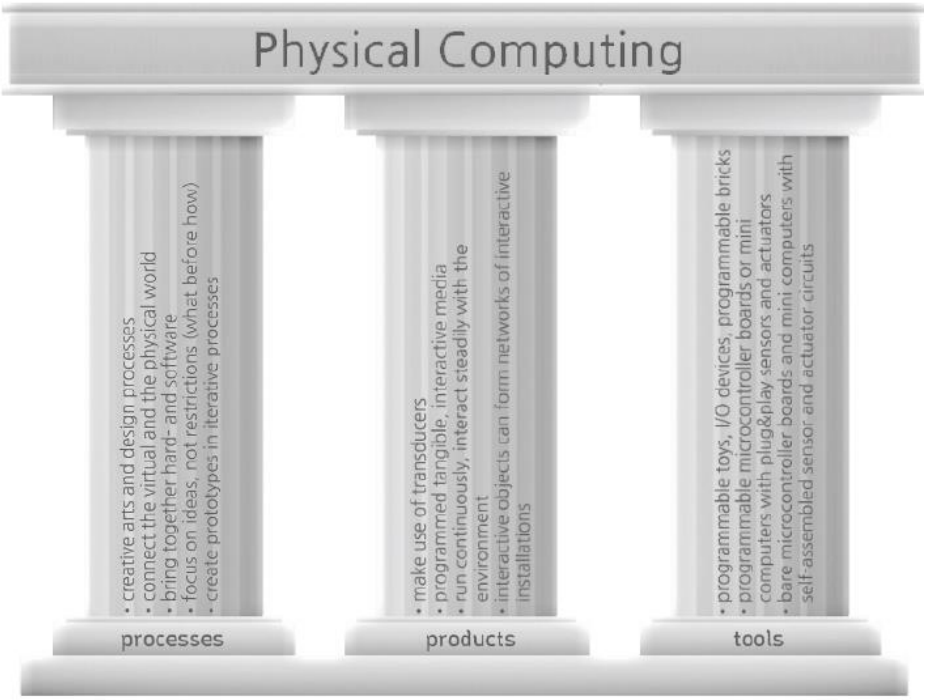
Καταρχάς, η περισσότερη πραγματική δουλειά γίνεται εκτός τάξης, στο εργαστήριο, κατασκευάζοντας και προγραμματίζοντας, αλλά και στην καθημερινότητα, παρατηρώντας τους ανθρώπους και ανακαλύπτοντας τι είναι αυτό που κάνουν που σας ενδιαφέρει να εντοπίσετε και να ερμηνεύσετε.

Κατά δεύτερον, αν δεν έχετε προγραμματίσει ποτέ ξανά έναν υπολογιστή, θα χρειαστεί να μάθετε κάποια πράγματα γι' αυτό. Πιο συγκεκριμένα, βοηθάει αν έχετε κάνει προγραμματισμό γραφικών διεπαφών, διότι αργότερα στο μάθημα, μαθαίνουμε πώς να ελέγχουμε τα γραφικά οθόνης και ήχου με τα αποτελέσματα των αισθητήρων. Θα έπρεπε, τουλάχιστον, να γνωρίζετε και να είστε εξοικειωμένοι/ες με τ' ακόλουθα:

- Τι είναι μια δήλωση if (if statement) και πώς τη χρησιμοποιούμε.
- Τι είναι οι μεταβλητές σ' ένα πρόγραμμα υπολογιστή και πώς χρησιμοποιούνται.
- Τι είναι ένας επαναλαμβανόμενος βρόχος (repeat loop) και πώς χρησιμοποιείται. Αν γνωρίζετε δηλώσεις (statements) όπως for-next, while-wend, ή repeat while… -end repeat, τότε είστε σε καλό δρόμο.

Τρεις πυλώνες του Εμπράγματου Προγραμματισμού

Ο εμπράγματος προγραμματισμός είναι μια σχετικά καινούρια κι άγνωστη έννοια μεταξύ των καθηγητών Πληροφορικής και των διδακτικών ερευνητών Πληροφορικής. Η βιβλιογραφική ανασκόπηση δείχνει ότι διαφορετικοί συγγραφείς δίνουν διαφορετική έμφαση στους τρεις ακόλουθους πυλώνες του εμπράγματου προγραμματισμού: τα τυπικά προϊόντα, εργαλεία και διαδικασίες του εμπράγματου προγραμματισμού πρέπει να ερευνηθούν, ώστε να διασαφηνιστεί το νόημά τους. Δεν είναι δυνατό να διαχωρίσουμε ξεκάθαρα τα τρία αυτά πεδία. Οι διαδικασίες περιλαμβάνουν εργαλεία και στοχεύουν σε συγκεκριμένα προϊόντα, οπότε, διαφορετικές αντιλήψεις θα εστιάσουν ως εκ τούτου περισσότερο σε οποιαδήποτε απ' τις τρεις μεταβλητές, χωρίς όμως να παραμελήσουν τις άλλες δύο. Ως σημείο εκκίνησης, θεωρείται ότι ο εμπράγματος προγραμματισμός συνεπάγεται το δημιουργικό σχεδιασμό απτών διαδραστικών αντικειμένων ή συστημάτων, με τη χρήση προγραμματίσιμου υλικού (hardware) (εικόνα 1).



ΕΙΚΟΝΑ 1. ΤΡΕΙΣ ΠΥΛΩΝΕΣ ΤΟΥ ΕΜΠΡΑΓΜΑΤΟΥ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ (PRZYBYLLA AND ROMEIKE)

7.2.2 Βασικές ικανότητες στον Εμπράγματο Προγραμματισμό

Κατανόηση των Υπολογιστικών Συστημάτων (Computing Systems)
Τα διαδραστικά αντικείμενα ή εγκαταστάσεις που δημιουργούνται με τον εμπράγματο προγραμματισμό, είναι ολόκληρα υπολογιστικά συστήματα που περιέχουν στοιχεία λειτουργικού και λογισμικού, τα οποία οι μαθητές μπορούν να συναρμολογήσουν μόνοι τους και να τα εξερευνήσουν περαιτέρω. Ανάλογα με το επίπεδο πολυπλοκότητας που υφίστανται στο συγκεκριμένο περιβάλλον, μπορούν να διανύσουν την απόσταση από τη διαισθητική αντίληψη (π.χ. κατά τον έλεγχο ενός προγραμματίσιμου παιχνιδιού) στη βαθιά γνώση των διαδραστικών υπολογιστικών συστημάτων (π.χ. κατά την κατασκευή ενός «έξυπνου» γραμματοκιβωτίου). Συγκεκριμένες πλευρές του σχεδιασμού του υλικού βοηθούν τους μαθητές ν' αναπτύξουν ικανότητες αναγνώρισης και κατανόησης των διαδραστικών συστημάτων στην καθημερινή τους ζωή.

Διατύπωση Προβλημάτων
Κατά τον εμπράγματο προγραμματισμό, διαμορφώνεται κι εξασκείται η βασική ικανότητα ακριβούς διατύπωσης προβλημάτων, ως ένα πρώτο βήμα στη διαδικασία σχεδιασμού και δημιουργίας διαδραστικών αντικειμένων. Οι μαθητές απαιτείται να περιγράφουν με σαφήνεια τι υποτίθεται ότι πρέπει να συμβεί από μια εξωτερική οπτική γωνία, εστιάζοντας έτσι στη διατύπωση προβλημάτων ξεχωριστά από το να σκέφτονται τους πιθανούς τρόπους επίλυσης προβλημάτων.

Οργάνωση κι Ανάλυση Δεδομένων

Στα έργα εμπράγματου προγραμματισμού, τα δεδομένα μπορούν να συλλεχθούν αυτόματα με αυτοματοποιημένους μετεωρολογικούς σταθμούς, ένα σύστημα ψηφοφορίας που εκλέγει τον επόμενο εκπρόσωπο της τάξης ή έναν αυτόματο καταγραφέα κυκλοφορίας που μετράει τον αριθμό των αυτοκινήτων που περνούν έξω απ' το σχολείο. Κατ' αυτόν τον τρόπο, οι μαθητές μαθαίνουν με δεδομένα συλλεγμένα απ' τον πραγματικό κόσμο στο δικό τους περιβάλλον, με αντικείμενα μέτρησης που σχεδίασαν και κατασκεύασαν οι ίδιοι. Θα μάθουν για την κωδικοποίηση και την αποκωδικοποίηση δεδομένων και πληροφοριών, ενώ δουλεύουν με αισθητήρες που εκδίδουν δεδομένα, τα οποία χρειάζεται να ερμηνευτούν, και ενεργοποιητές που λαμβάνουν δεδομένα, τα οποία χρειάζεται να παραχθούν απ' τις πληροφορίες.

Αλγοριθμική Σκέψη
Η αλγοριθμική σκέψη είναι ένα επίσης σημαντικό στοιχείο της διαδικασίας του εμπράγματου προγραμματισμού. Οι μαθητές όλων των επιπέδων απαιτείται να περιγράφουν με ακρίβεια ακολουθίες γεγονότων, και σειριακών και παράλληλων. Ο εμπράγματος προγραμματισμός ειδικότερα απαιτεί απ' τους μαθητές να αναπτύξουν αλγορίθμους που επιτρέπουν στα αντικείμενά τους να λειτουργούν αδιάλειπτα και να αλληλεπιδρούν σταθερά με το περιβάλλον.

Αποτελεσματικότητα και Αποδοτικότητα
Βασικές πτυχές της υπολογιστικής σκέψης περιλαμβάνουν την αναγνώριση, ανάλυση, και εφαρμογή πιθανών λύσεων, με σκοπό να επιτευχθεί ο πιο αποτελεσματικός και αποδοτικός συνδυασμός βημάτων και πόρων. Στα έργα εμπράγματου προγραμματισμού, αναποτελεσματικές ή αντιπαραγωγικές λύσεις είναι ιδιαιτέρως εμφανείς. Τα διαδραστικά αντικείμενα ή εγκαταστάσεις θα πρέπει να δίνουν άμεση ανατροφοδότηση, ίσως να περιλαμβάνουν ταυτόχρονες διεργασίες, ενώ πρέπει να περιλαμβάνουν και εύληπτες διεπαφές. Εάν δεν ανταποκριθούν σ' αυτές τις προσδοκίες, π.χ. λόγω της επιλογής ακατάλληλων αισθητήρων, υπερβολικών παύσεων ή καθυστερημένων ανταποκρίσεων, αυτό γίνεται άμεσα αντιληπτό.

7.3 Πρακτικές εφαρμογές

Για να δοκιμάσετε τις γνώσεις σας και να εξασκηθείτε περαιτέρω, προτείνουμε ορισμένα πρακτικά παραδείγματα.

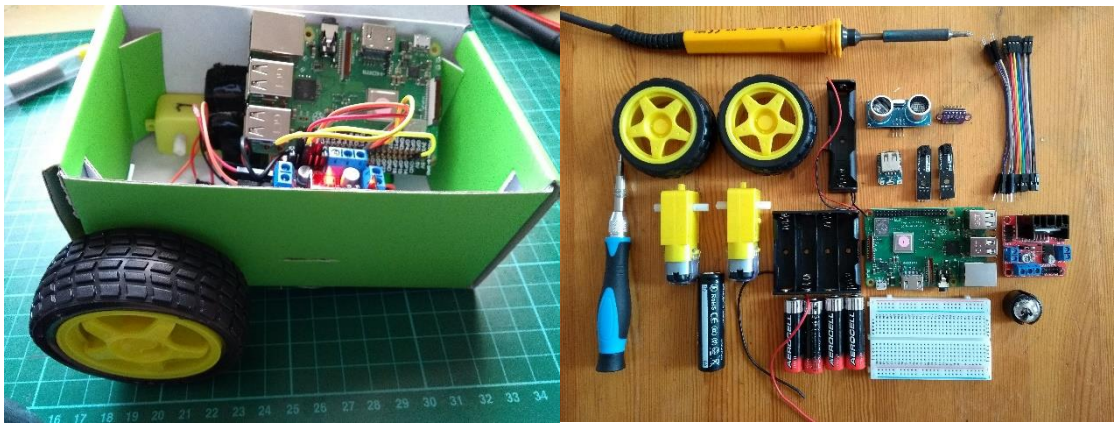
- **Παράδειγμα 1: Robot Buggy (Ένα Raspberry Pi σε τροχούς;)**
- **Παράδειγμα 2: YouTube Boombox (πρόγραμμα αναπαραγωγής βίντεο Lo-fi με Raspberry Pi για YouTube)**

7.3.1 Παράδειγμα 1

Αυτό το έργο για αρχάριους σας διδάσκει πώς να δημιουργήσετε μια πλακέτα ελεγκτή κινητήρα, να δημιουργήσετε ένα πλαίσιο και να χρησιμοποιήσετε το Python για τον έλεγχο των κινητήρων. Μην φοβάστε, είναι πολύ εύκολο να ρυθμίσετε. Σε αυτό το έργο θα δημιουργήσετε ένα ρομπότ που μπορείτε να προγραμματίσετε για να μετακινηθείτε χρησιμοποιώντας απλές εντολές Python.

- Πώς να εγκαταστήσετε μια πλακέτα ελεγκτή κινητήρα με δύο κινητήρες

- Πώς να ελέγχετε τους κινητήρες χρησιμοποιώντας το Python
- Πώς να φτιάξετε ένα πλαίσιο ρομπότ



ΕΙΚΟΝΑ 2. ROBOT BUGGY

Τι θα χρειαστείτε:

- Raspberry Pi 3
- Πλακέτα ελεγκτή κινητήρα
- Κινητήρες DC 2 × 3V - 6V
- 2 × τροχοί
- 1 × μπαταρία AA AA (για 4 μπαταρίες AA)
- 4 × μπαταρίες AA
- Τροχίσκο
- Καλώδια καλωδίων ή βραχυκυκλωτήρα
- Μια μπαταρία USB
- Κατσαβίδι
- Συγκόλληση και κολλητήρι
- Απογυμνωτές καλωδίων
- Μικρό κουτί από χαρτόνι ή πλαστικό και κόλλα / ταινία

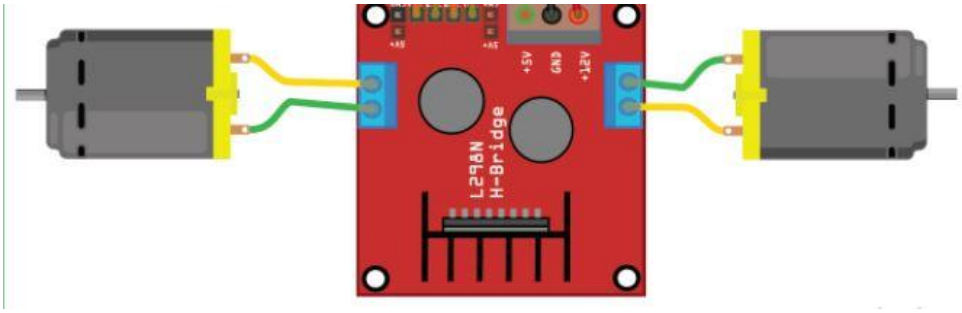
Συναρμολόγηση των κινητήρων και του κέλυφους

Το πρώτο πράγμα που θα θέλατε να κάνετε είναι να συνδέσετε την πλακέτα ελεγκτή του κινητήρα σας στο Raspberry Pi, την μπαταρία και τους δύο κινητήρες σας, για να ελέγξετε ότι όλα λειτουργούν.

Οι οδηγίες εδώ είναι για το L298N Dual H Bridge DC Stepper Motor Driver Controller Board και θα είναι αρκετά παρόμοιες για τις περισσότερες πλακέτες ελεγκτή κινητήρα.

Συνδέστε τους κινητήρες στην πλακέτα

Θα πρέπει να συνδέσετε τους κινητήρες στην πλακέτα. Για αυτό θα χρειαστείτε ένα μικρό κατσαβίδι.



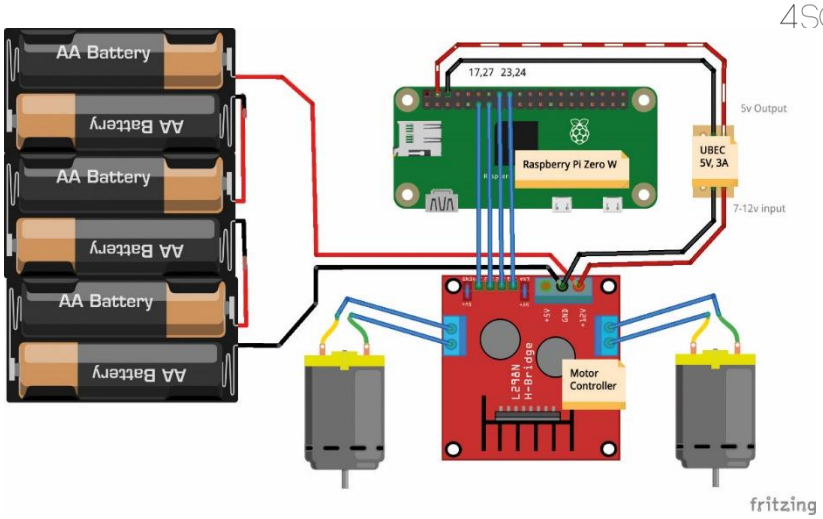
Τροφοδοσία των κινητήρων

Οι κινητήρες απαιτούν περισσότερη ισχύ από ό, τι μπορεί να προσφέρει το Raspberry Pi. Επομένως, θα χρησιμοποιήσετε τέσσερις μπαταρίες AA για να τις τροφοδοτήσετε.

Σύνδεση της πλακέτας με το Raspberry Pi

Ο πίνακας που χρησιμοποιείται σε αυτό το έργο πρέπει να συνδεθεί στο Raspberry Pi. Άλλες πλακέτες μπορεί να συνδεθούν διαφορετικά, και μερικές σανίδες μπορούν απλώς να τοποθετηθούν στις καρφίτσες Raspberry Pi GPIO ως HAT.

Στον πίνακα που χρησιμοποιείται εδώ υπάρχουν ακίδες με ετικέτες In1, In2, In3 και In4, καθώς και δύο καρφίτσες GND. Ποιες καρφίτσες GPIO στο Pi που χρησιμοποιείτε εξαρτάται από εσάς; σε αυτό το έργο, χρησιμοποιήθηκαν τα GPIO 17, 27, 23 και 24.



Αριστερά, δεξιά, εμπρός, πίσω
Πρέπει να καταλάβετε ποιος είναι ο αριστερός κινητήρας σας και ποιος είναι ο σωστός κινητήρας σας. Πρέπει επίσης να ξέρετε με ποιον τρόπο οδηγούν για να προχωρήσουν και με ποιον τρόπο οδηγούν για να πάνε πίσω.

Συναρμολογήστε το ρομπότ σας
Δεν υπάρχει «σωστός» τρόπος για να δημιουργήσετε το πρωτότυπο πλαίσιο ρομπότ σας, αλλά υπάρχουν μερικά πράγματα που πρέπει να θυμάστε:

Το πλαίσιο πρέπει να διαθέτει το Raspberry Pi, τον ελεγκτή κινητήρα και τις μπαταρίες.
Το πλαίσιο πρέπει να επιτρέπει την τοποθέτηση ενός ζεύγους τροχών.
Ίσως θέλετε αργότερα να προσθέσετε μερικούς αισθητήρες γραμμής και έναν αισθητήρα απόστασης υπερήχων ή έναν αισθητήρα lidar στο πλαίσιο.
Είναι πάντα καλή ιδέα να δημιουργήσετε πρώτα ένα πρωτότυπο πλαίσιο. Τελικά, μπορείτε να μάθετε πώς να κόβετε λέιζερ ή να εκτυπώνετε τρισδιάστατα ένα πλαίσιο, αλλά σε αυτό το έργο, ένα χαρτόκουτο χρησιμοποιείται ως προσωρινή λύση.

Για να χρησιμοποιήσετε το Raspberry Pi χωρίς να συνδέσετε ποντίκι, οθόνη ή πληκτρολόγιο, μπορείτε να αποκτήσετε απομακρυσμένη πρόσβαση σε αυτό μέσω SSH ή VNC.
Τώρα μπορείτε να γράψετε ένα πρόγραμμα για τον έλεγχο του ρομπότ σας και να το κάνετε να κάνει πολλά πράγματα.
Ακολουθεί ένα απλό σενάριο για να το κάνει σε τετράγωνο σχήμα (ίσως χρειαστεί να αλλάξετε ελαφρώς τις λειτουργίες ύπνου):

```
from gpiozero import Robot
from time import sleep
robot = Robot(left = (17, 27), right = (23, 24))
while True:
    robot.forward()
    sleep(3)
    robot.stop()
    robot.right()
    sleep(1)
    robot.stop()
```

Τώρα είναι η ευκαιρία σας να προγραμματίσετε το ρομπότ σας!

Δοκιμάστε και ολοκληρώστε μία από τις ακόλουθες προκλήσεις:

1. Κάντε το ρομπότ σας σε τέλειο κύκλο
2. Κάντε το ρομπότ σας σε μοτίβο ζιγκ-ζαγκ
3. Κάντε ένα μικρό λαβύρινθο από οικιακά αντικείμενα και προγραμματίστε το ρομπότ σας για να το πλοηγηθείτε

Μην ξεχνάτε, υπάρχουν μόνο πέντε βασικές εντολές για να μετακινήσετε το ρομπότ σας:

```
robot.forward()
robot.backward()
robot.right()
robot.left()
robot.stop()
```

7.3.2 Παράδειγμα 2

Αυτή η μικρή συσκευή μπορεί να μοιάζει με το πίσω μέρος μιας μικρής φωτογραφικής μηχανής, αλλά στην πραγματικότητα είναι ένα πρόγραμμα αναπαραγωγής βίντεο Lo-fi με Raspberry Pi για το YouTube. Η τρισδιάστατη τυπωμένη θήκη διαθέτει επίσης ενσωματωμένο ηχείο και μικρή οθόνη. Το Pi έχει ρυθμιστεί σε λειτουργία kiosk και αναπαράγει αυτόματα οποιαδήποτε ροή μουσικής YouTube. Ένα ενσωματωμένο χειριστήριο μπορεί να χρησιμοποιηθεί για να αλλάξετε τους σταθμούς και να ρυθμίσετε την ένταση.

- Το BrainCraft HAT διαθέτει όλα όσα χρειάζεστε για να δημιουργήσετε ένα πρόγραμμα αναπαραγωγής YouTube all-in-one.
- Το ενσωματωμένο χειριστήριο μπορεί να χρησιμοποιηθεί για αλλαγή των σταθμών και ρύθμιση της έντασης.
- Αυτό σας επιτρέπει να αλλάζετε γρήγορα μεταξύ διαφορετικών καναλιών YouTube χωρίς να χρειάζεται να χρησιμοποιήσετε πληκτρολόγιο ή ποντίκι.
- Μπορείτε επίσης να κάνετε παύση και αναπαραγωγή του βίντεο πατώντας το κουμπί δίπλα στο χειριστήριο.

Τι θα χρειαστείτε:

Ηλεκτρονικά μέρη:

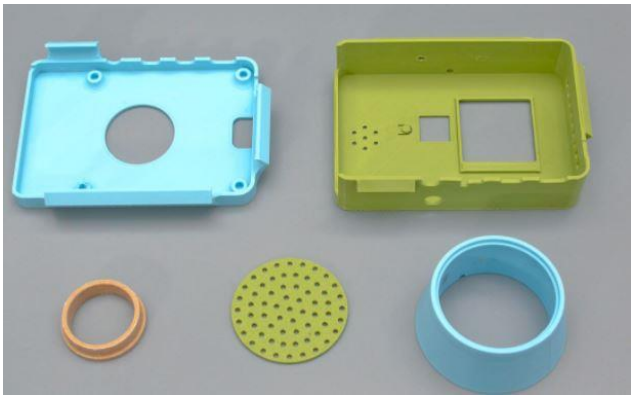
- Raspberry Pi 4 Model B - 4 GB RAM
- Κάρτα 16 GB με NOOBS 3.1 για υπολογιστές Raspberry Pi συμπεριλαμβανομένων 4
- Ηχείο - Διάμετρος 40mm - 4 Ohm 3 Watt
- Καλώδιο 2 ακίδων JST PH - Θηλυκός σύνδεσμος 100mm
- Ασύρματο πληκτρολόγιο πλήρους μεγέθους με Trackpad

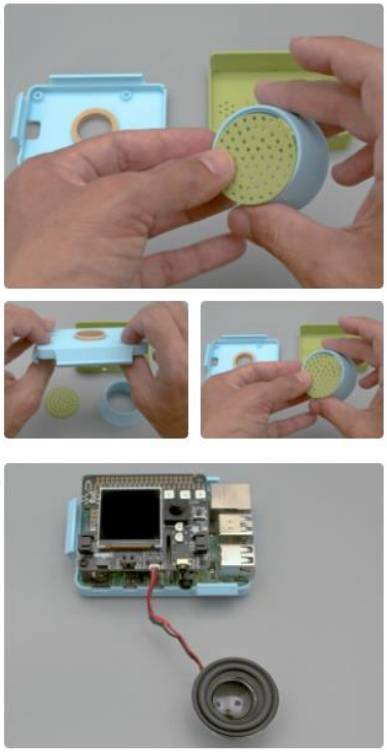
Εκτύπωση 3D:

Τα αρχεία STL για τρισδιάστατη εκτύπωση είναι προσανατολισμένα στην εκτύπωση "ως έχει" σε μηχανήματα τύπου FDM. Τα ανταλλακτικά έχουν σχεδιαστεί για εκτύπωση 3D χωρίς υλικό υποστήριξης. Μπορείτε να κατεβάσετε την αρχική πηγή σχεδίασης χρησιμοποιώντας τους παρακάτω συνδέσμους:

- BrainTube-screen.stl
- BrainTube-case.stl
- BrainTube-tripod.stl
- BrainTube-speaker-grill.stl
- BrainTube-speaker-cone.stl
- BrainTube-speaker-ring.stl

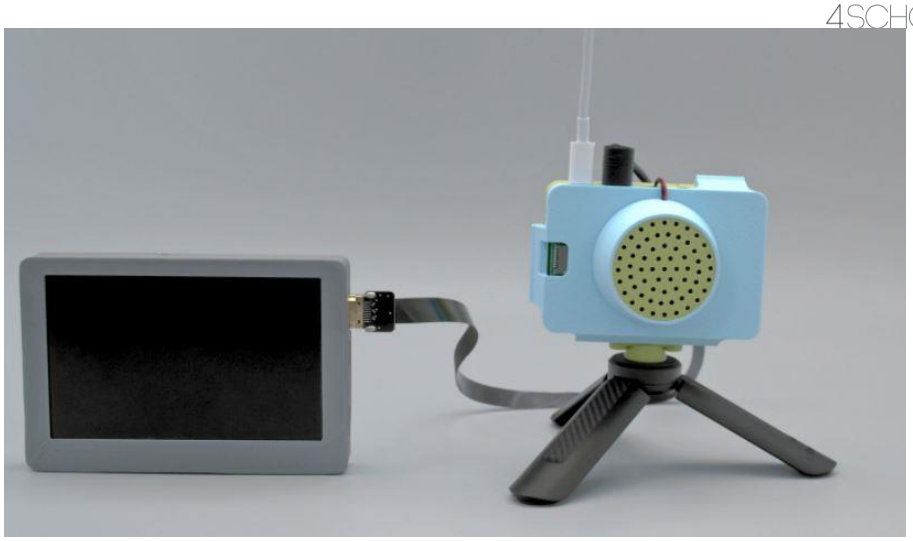
https://www.thingiverse.com/thing:4739213





Συναρμολόγηση

- Πατήστε το στήριγμα ηχείων Fit
  Το τμήμα του δακτυλίου των ηχείων ταιριάζει στη διακοπή του τμήματος Brain-Tube-Case.
- Γκριλ ηχείων
  Τοποθετήστε το μέρος του ηχείου-σχάρας στο τμήμα του ηχείου-κώνου.
- Προσθέστε το Raspberry Pi στη θήκη
  Τοποθετήστε το Pi πάνω στις αναμονές και ευθυγραμμίστε την πλακέτα με τα α-νοίγματα της θύρας στη θήκη.
- Ομιλητής
  Κολλήστε τα καλώδια του ηχείου στο θηλυκό βύσμα JST. Χρησιμοποιήστε συρρί-κνωση θερμότητας για να μονώσετε τις συνδέσεις
- Θήκη οθόνης
  - Το μπροστινό κάλυμμα είναι τοποθετημένο πάνω από το HAT με το καλώδιο ηχείου τοποθετημένο μέσω της σχισμής στο άνοιγμα της θύρας HDMI.
  - Προσθέστε ταινία γκάφερ ή ταινία καπόν για να μονώσετε τον μαγνήτη των ηχείων. Η πρέσα ηχείων ταιριάζει στο δαχτυλίδι.
  - Ευθυγραμμίστε τον κώνο του ηχείου πάνω από τα καλώδια του ηχείου και, στη συνέχεια, πιέστε το κουμπί πάνω από το ηχείο.



Ρυθμίστε το Raspberry Pi

- Ρύθμιση κάρτας SD
  Συνδέστε μια οθόνη HDMI, για να ρυθμίσετε το Pi OS. Χρησιμοποιήστε το επίσημο Raspberry Pi Imager για να εγγράψετε το πιο πρόσφατο λειτουργικό σύστημα σε μια κάρτα SD.
- Ενεργοποίηση SSH
  Αφού ολοκληρωθούν όλες οι ενημερώσεις, ενεργοποιήστε το SSH.
- Εγκαταστήστε τις βιβλιοθήκες Blinka
  - Επιτρέπει σε πολλές από τις βιβλιοθήκες που γράφτηκαν για το CircuitPython να εκτελούνται στο CPython για Linux. Ακολουθήστε το παρακάτω βήμα για να ρυθμίσετε τις βιβλιοθήκες Blinka:
    Ρύθμιση Blinka
    - Διαμόρφωση ήχου, ανεμιστήρα και οθόνης

- Εγκαταστήστε το σενάριο λειτουργίας Kiosk
  Χρησιμοποιήστε το σενάριο λειτουργίας kiosk για να ενεργοποιήσετε την αναπαραγωγή βίντεο πλήρους οθόνης.
- Επεξεργασία λίστας αναπαραγωγής

Προσθέστε τις δικές σας ροές.

7.4 Τεστ αξιολόγησης

1. Εφαρμογές εισόδου ή εξόδου;
   - έναν υπολογιστή όπως ένα iPad, smartphone, φορητό υπολογιστή ή υπολογιστή
   - **ένα κομμάτι υλικού που χρησιμοποιείται για την ανταλλαγή πληροφοριών με έναν υπολογιστή.**
   - ένας τύπος οχήματος όπως αυτοκίνητο ή φορτηγό
   - κάτι που θα κατεβάσατε και θα εγκαταστήσετε στο iPhone σας

2. Τι είναι τα δεδομένα;
   - Το μέρος όπου αποθηκεύονται οι πληροφορίες στον υπολογιστή
   - Ένα μικρό κομμάτι του υπολογιστή
   - Ένα είδος ρομπότ
   - **Πληροφορίες που χρησιμοποιούνται από υπολογιστές και ανθρώπους**

3. Το πληκτρολόγιο είναι μια από τις πιο διακεκριμένες _____ συσκευές ενός υπολογιστή.
   - **Είσοδος**
   - Έξοδος
   - Είσοδος και έξοδος

4. Όταν ένα φωτοτυπικό τυπώνει κάτι, τότε _____.
   - εισάγει
   - **εξάγει**
   - εισάγει και εξάγει

5. Τα ακουστικά εικονικής πραγματικότητας (VR) αποτελούν παράδειγμα _____ για έναν υπολογιστή.
   - εισόδου
   - εξόδου
   - **εισόδου και εξόδου**

6. Τι ΔΕΝ μπορείτε να κάνετε στο Shell;
   - **Να αποθηκεύσετε τον κώδικά σας και να τον «τρέξετε» ξανά**
   - Να εκτελέσετε τον κώδικά σας
   - Να εξάγετε τις απαντήσεις ενός προβλήματος μαθηματικών

7. Ποιο απ' τα παρακάτω αποτελεί πλεονέκτημα της χρήσης μεταβλητών; (Επιλέξτε όλες τις σωστές απαντήσεις).
   - **Μπορούν να χρησιμοποιηθούν για την αποθήκευση εισόδου.**
   - **Δε χρειάζεται να πληκτρολογήσουμε πληροφορίες ξανά και ξανά.**
   - **Μικραίνουν τον κώδικα και διευκολύνουν την ανάγνωσή του.**
   - **Ευκολότερη η χρήση πληροφοριών στον κώδικα.**

8. Έξοδος κειμένου στη Python: Ποιο είναι το σωστό;
   - print("Happy New Year!)
   - output("Happy New Year!")

- Print("Happy New Year!")
- **print("Happy New Year!")**

9. Ποιος είναι ο σωστός τρόπος να γράψετε «διαίρεσε το a διά 2 και πολλαπλασίασέ το επί του b» στην Python;
- a / 2 x b
- a \ 2 x b
- **a / 2 * b**
- a \ 2 * b

10. Πώς φτιάχνουμε μια μεταβλητή που ονομάζεται d και περιέχει το κείμενο «Good morning» στην Python;
- d(Good morning)
- d = (Good morning)
- d = Good morning
- **d = "Good morning"**

7.5 Παραπομπές

- https://en.wikipedia.org/wiki/Physical_computing
- https://www.tigoe.com/blog/what-is-physical-computing/
- https://www.qaeducation.co.uk/news/physical-computing#:~:text=Physical%20computing%20means%20interacting%20with,or%20even%20controlling%20robotic%20objects.
- https://guides.temple.edu/c.php?g=419841&p=2863656
- Mareen Przybylla, Ralf Romeike, Βασικές ικανότητες στον Εμπράγματο Προγραμματισμό https://publishup.uni-potsdam.de/opus4-ubp/frontdoor/deliver/index/docId/8290/file/cid07_S351-361.pdf
- https://sunnie-sva-physicalcomputing.tumblr.com/post/66904922347/physical-computing-mid-term-project-music
- https://quizizz.com/admin/quiz/5e153511ae952c001b88401a/physical-computing-review-quiz-1-d

7.6 Επιπρόσθετες πηγές

- Filiz Kalelioglu, Sue Sentence, (2020) Teaching with physical computing in school: the case of the micro:bit, *Education and Information Technologies* 25, 2577–2603.
- Tom Igoe, Making Things Talk: Practical Methods for Connecting Physical Objects, Make; 1 edition (September 28, 2007), ISBN-10: 0596510519 (source of examples). Second Edition, Released: August 2011 (est.) ISBN-10: 1449392431, ISBN-13: 978-1449392437, http://oreilly.com/catalog/0636920010920

- Joshua Noble, Programming Interactivity: A Designer's Guide to Processing, Arduino, and OpenFrameworks, O'Reilly Media, July 2009
- Alvaro Cassinelli, Daniel Saakes, Data Flow, Spatial Physical Computing, TEI 2017, March 20–23, 2017, Yokohama, Japan

7.7 Συμπεράσματα

Ο Εμπράγματος Προγραμματισμός είναι μία πολύπλοκη δραστηριότητα, που απαιτεί από τους μαθητές να προσέχουν θέματα στο λειτουργικό και το λογισμικό ταυτόχρονα. Σε εισαγωγικό επίπεδο (δημοτικό), οι μαθητές ίσως να δουλεύουν με προγραμματίσιμα παιχνίδια ή τουβλάκια και να μεταφέρουν (σύρουν) και να αποθέτουν περιβάλλοντα προγραμματισμού, για να μάθουν τα βασικά της αλγοριθμικής σκέψης. Σετ κατασκευών που περιλαμβάνουν προ-εγκατεστημένους αισθητήρες και ενεργοποιητές και μπορούν να συνδεθούν με ένα μικρο-ελεγκτή, είτε απευθείας είτε με μια ασπίδα, επιτρέπουν σε μεγαλύτερα παιδιά να πετύχουν ορατά και χειροπιαστά επιτεύγματα πολύ γρήγορα. Με την εξέλιξη στον εμπράγματο προγραμματισμό, τόσο από πλευράς λειτουργικού όσο και λογισμικού, οι μαθητές υποβάλλονται σε διεργασίες εκμάθησης που ενισχύουν την υπολογιστική σκέψη και τις βασικές ικανότητες που είναι απαραίτητες σε όλες τις πλευρές της ζωής. Ο εμπράγματος προγραμματισμός μπορεί να εμπλουτίσει τα μελλοντικά μαθήματα Πληροφορικής με πολύτιμες ικανότητες, που εστιάζουν στην εκπαίδευση στην Πληροφορική περισσότερο από πολλά άλλα μαθήματα, καθώς είναι εγγενείς στο μάθημα και όχι τεχνητά επιβεβλημένες. Μελλοντικές έρευνες πάνω στο θέμα θα εξερευνήσουν ως εκ τούτου το πώς βιώνουν οι μαθητές διαφορετικών ηλικιακών ομάδων τις δραστηριότητες εμπράγματου προγραμματισμού και ποιες διαδικασίες μάθησης υφίστανται.

8 Γενικά συμπεράσματα

Ακόμα κι αν τα παιδιά γεννιούνται μες στην τεχνολογία, φαίνεται ότι χρειάζεται να αποκτήσουν τεχνολογικές δεξιότητες, όπως ο προγραμματισμός. Νέοι τρόποι προσέλκυσης των παιδιών στον προγραμματισμό και το STEM είναι απαραίτητοι. Επιλέγουμε το ενεργό παιχνίδι μέσω της δημιουργίας παιχνιδιών που μπορούν να παιχτούν σε ένα ρετρό, DIY ξύλινο υπολογιστή, σε συνδυασμό με ηλεκτρονικές συσκευές που σχετίζονται με θέματα STEM. Αυτή η προσέγγιση είναι διασκεδαστική και πιο εκπαιδευτική σε σχέση με τον περισσότερο χρόνο μπροστά σε μια οθόνη. Η γεφύρωση μεταξύ των «online» και των «offline» κόσμων δύναται να προσφέρει ένα πιο ελκυστικό και υγιές περιβάλλον για να μάθουν τα παιδιά πώς να προγραμματίζουν και να αναπτύξουν δεξιότητες STEM.

Ο πρωταρχικός στόχος του προγράμματος Erasmus+, STEMKIT4Schools, είναι να παράξει προσεγγίσεις κι εργαλεία ώστε να βοηθήσει εκείνους που δουλεύουν με παιδιά, αναπτύσσοντας δεξιότητες σχετικές με το STEM. Στοχεύει στην επίτευξη αυτού, όχι με την αύξηση του χρόνου μπροστά στην οθόνη, αλλά ενθαρρύνοντας το ενεργό παιχνίδι.

Ένας ολοκληρωμένος οδηγός σπουδών έχει σχεδιαστεί από τους συνεργάτες του προγράμματος, για ν' ανταποκριθεί στους στόχους του STEMKIT4Schools, και περιλαμβάνει και πλάνα μαθημάτων για τη χρήση των Minecraft Pi/Scratch/Python/Kits με τον υπολογιστή STEMKIT στην τάξη. Τα προγράμματα μαθημάτων αποτελούν μέρος του Οδηγού Εκπαιδευτικών για δασκάλους. Τα σετ ηλεκτρονικών είναι σχεδιασμένα και κατασκευασμένα να συμπληρώνουν τη διδασκαλία του προγραμματισμού, του εμπράγματου προγραμματισμού και των μαθημάτων STEM. Το τελικό μάθημα STEMKIT διδάσκει βασικά στοιχεία των παιχνιδιών με τα Minecraft Pi, Python, Scratch, καθώς επίσης και του εμπράγματου προγραμματισμού (με τη χρήση βασικών ψηφιακών, αναλογικών, και ηλεκτρομηχανικών στοιχείων) και της συνεργασίας (ενασχόληση και μοίρασμα με τους άλλους). Ο DIY υπολογιστής STEMKIT βασίζεται στο Raspberry Pi, που σημαίνει ότι μπορεί να αξιοποιήσει την ισχύ και τις πρακτικά απεριόριστες πηγές για το Raspberry, ενώ τα χαρακτηριστικά του λειτουργικού συστήματος Raspbian παρέχουν επιπρόσθετες δυνατότητες. Ο αναθεωρημένος οδηγός σπουδών STEMKIT, χρησιμοποιώντας αρχές εκπαιδευτικού σχεδιασμού, μπορεί να βρεθεί μέσα στην Πύλη Μάθησης (Learning Portal), υπό τη μορφή διαδραστικών Μαθησιακών Αντικειμένων (Learning Objects), και θα παραδοθεί στους μαθητές που θα είναι σε θέση να ακολουθήσουν το διαδραστικό μάθημα στο δικό τους χρόνο και χώρο, ενώ μπορούν να χρησιμοποιήσουν τα προσφερόμενα εργαλεία κοινωνικής μάθησης για να εμπλακούν με τους συνομηλίκους τους (εμπλοκή με σχόλια, forum, chat). Επιπρόσθετα χαρακτηριστικά συνεργασίας, όπως ειδοποιήσεις, ομαδική ροή νέων, blog, διαμοιρασμός αρχείων, ερωταπαντήσεις, και ψηφοφορίες θα προσφέρονται επίσης.

Το αποτέλεσμα O2A1 «ΣΧΕΔΙΑΣΜΟΣ ΚΑΙ ΑΝΑΠΤΥΞΗ ΤΟΥ ΟΔΗΓΟΥ ΣΠΟΥΔΩΝ STEMKIT» περιέχει τις εξής 5 ενότητες ονομαστικά: Εισαγωγή στο Scratch 2.0 [P6-ARC], Το Scratch GPIO (Έλεγχος ακίδων GPIO/λήψη καταχωρήσεων) [P3-DANMAR], Εισαγωγή στην Έκδοση Raspberry Pi του Minecraft [P4-HESO / P5-SCHOLE], Προγραμματισμός του Raspberry Pi GPIO με τη χρήση Python [P2-AKNOW], και Εμπράγματος Προγραμματισμός [P1-ECAM].

Το Scratch έχει ένα μικρό αριθμό εντολών, επιτρέπει στα αντικείμενα να ανταλλαχθούν χωρίς τη διάσπαση των εξαρτήσεων, προάγοντας τη συνεργασία και τον διαμοιρασμό του κώδικα. Το σύστημα είναι πάντα ενεργό, χωρίς διακόπτη εκτέλεσης/επεξεργασίας, ώστε

οι εντολές ή τα αποσπάσματα κώδικα να μπορούν να εκτελεστούν με ένα κλικ, ενώ η εικονική ανατροφοδότηση δείχνει την εκτέλεση. Οι μεταβλητές και οι λίστες έχουν σαφείς απεικονίσεις, έτσι το αποτέλεσμα των χειρισμών των δεδομένων μπορεί να γίνει ορατό αμέσως (Maloney, 2010).
Η ικανότητα κωδικοποίησης προγραμμάτων υπολογιστών αποτελεί σημαντικό κομμάτι αλφαβητισμού στη σημερινή κοινωνία. Όταν μαθαίνει κανείς να γράφει κώδικα στο Scratch, μαθαίνει σημαντικές στρατηγικές για την επίλυση προβλημάτων, τη σχεδίαση έργων και την ανταλλαγή ιδεών.
Τα Έργα Scratch: βάσει μαθησιακών ενοτήτων, περιλαμβάνουν διαφορετικούς τομείς και αυτή η νέα προοπτική θα επιτρέψει στους μαθητές να εφαρμόσουν ό,τι έχουν μάθει σε νέες καταστάσεις, με στόχο τη βαθύτερη μάθηση. Οι μαθητές θα εμπλακούν στο σχεδιασμό δραστηριοτήτων, ακολουθώντας τα προσωπικά τους ενδιαφέροντα, αλληλεπιδρώντας μέσω δημιουργικών συνεργασιών, αναλογιζόμενοι τις χρήσιμες εμπειρίες.

Η εισαγωγή στο Minecraft Pi μας επιτρέπει τα ακόλουθα βασικά συμπεράσματα:
Αν ακολουθήσατε αυτό το βοήθημα με το Raspberry Pi, αναμένεται να:

- Έχετε πρόσβαση στο Minecraft Pi και να δημιουργείτε ένα νέο κόσμο.
- Πλοηγήστε στο Minecraft Pi με τη χρήση των πλήκτρων κίνησης στο πληκτρολόγιό σας.
- Γνωρίζετε πώς να τοποθετήσετε και να καταστρέψετε ένα μπλοκ, καθώς και να πλοηγήστε μεταξύ των διαφορετικών ειδών μπλοκ στο ευρετήριο του παιχνιδιού.
- Συνδέετε την Python στο Minecraft Pi.
- Χρησιμοποιείτε το περιβάλλον προγραμματισμού της Python.
- Χειρίζεστε μπλοκ με τη χρήση κώδικα και σεναρίων στην Python.
- Γνωρίζετε καλά τις υπόλοιπες λειτουργίες του Minecraft Pi.
- Κάνετε το Minecraft να αλληλεπιδρά με τον εξωτερικό κόσμο με τη χρήση κουμπιών και λυχνιών LED.
- Εισάγετε νέους κόσμους και ν’ ανακαλύπτετε πακέτα βοηθημάτων συμβατών με το Minecraft Pi.

Ο προγραμματισμός του Raspberry Pi με τη χρήση της Python μας επιτρέπει τα ακόλουθα συμπεράσματα:
Με την ολοκλήρωση αυτού του οδηγού σπουδών, αναμένεται να έχετε μάθει, καταλάβει και εξασκηθεί στα ακόλουθα μαθησιακά στοιχεία:

- Τι είναι μια ακίδα Raspberry Pi GPIO και τη βασική της λειτουργία.
- Πώς να φτιάχνετε βασικά κυκλώματα με λυχνίες LED, βομβητές (buzzer), αισθητήρες και να τα συνδέετε στις ακίδες GPIO του Raspberry Pi.
- Πώς να στήσετε, ελέγξετε και να προγραμματίσετε τις ακίδες GPIO με τη χρήση της γλώσσας προγραμματισμού Python.
- Εξοικείωση με και χρήση των κυριότερων δομών και συντακτικού της γλώσσας προγραμματισμού Python, συμπεριλαμβανομένων των λειτουργιών, μεταβλητών, των for loops, while loops, δηλώσεων if.

Ο Εμπράγματος Προγραμματισμός (Physical Computing) είναι μια πολύπλοκη δραστηριότητα που απαιτεί απ’ τους μαθητές να προσέχουν θέματα λειτουργικού και λογισμικού ταυτόχρονα. Στο εισαγωγικό επίπεδο (δημοτικό), οι μαθητές πιθανώς θα δουλέψουν με προγραμματίσιμα παιχνίδια ή τουβλάκια, και θα μεταφέρουν (σύρουν) και αποθέτουν περιβάλλοντα προγραμματισμού, για να μάθουν τα βασικά της αλγοριθμικής σκέψης. Σετ κατασκευών που περιλαμβάνουν προ-εγκατεστημένους αισθητήρες και ενεργοποιητές

που μπορούν να συνδεθούν με ένα μικρο-ελεγκτή, είτε απευθείας είτε με μια ασπίδα, επιτρέπουν σε μεγαλύτερα παιδιά να πετύχουν ορατά και χειροπιαστά επιτεύγματα πολύ γρήγορα. Με την εξέλιξη στον εμπράγματο προγραμματισμό, τόσο από πλευράς λειτουργικού όσο και λογισμικού, οι μαθητές υποβάλλονται σε διεργασίες εκμάθησης που ενισχύουν την υπολογιστική σκέψη και τις βασικές ικανότητες που είναι απαραίτητες σε όλες τις πλευρές της ζωής. Ο εμπράγματος προγραμματισμός μπορεί να εμπλουτίσει τις μελλοντικές τάξεις Πληροφορικής με πολύτιμες ικανότητες, που εστιάζουν στην εκπαίδευση στην Πληροφορική περισσότερο από πολλά άλλα μαθήματα, καθώς είναι εγγενείς στο μάθημα και όχι τεχνητά επιβεβλημένες. Μελλοντικές έρευνες πάνω στο θέμα θα εξερευνήσουν ως εκ τούτου το πώς βιώνουν οι μαθητές διαφορετικών ηλικιακών ομάδων τις δραστηριότητες εμπράγματου προγραμματισμού και ποιες διαδικασίες μάθησης υφίστανται.