Συνέδρια της Ελληνικής Επιστημονικής Ένωσης Τεχνολογιών Πληροφορίας & Επικοινωνιών στην Εκπαίδευση

Τόμ. 1 (2023)

13ο Πανελλήνιο και Διεθνές Συνέδριο «Οι ΤΠΕ στην Εκπαίδευση»

PhysicsCPP: modular πρόγραμμα φυσικών παραστάσεων σε C++

Νικόλαος Κωνσταντάτος, Ιωάννης Θεοχαρόπουλος, Παναγιώτα Βασιλείου, Ζωή Καραδήμα, Πολυξένη Μπίλλα

PhysicsCPP: modular πρόγραμμα φυσικών παραστάσεων σε C++

Νικόλαος Κωνσταντάτος[1], Ιωάννης Θεοχαρόπουλος[2], Παναγιώτα Βασίλειου[2] Ζωή Καραδήμα[2], Πολυξένη Μπίλλα[3]
1gelpetroup@gmail.com, ioannistheo@yahoo.com, zoikaradima2000@gmail.com, vgiota@gmail.com, pollybilla@gmail.com
[1] Μαθητής Α' Λυκείου 1ου ΓΕΛ Πετρούπολης,
[2] Εκπαιδευτικός 1ου ΓΕΛ Πετρούπολης,
[3] Εκπαιδευτικός Μουσικού Γυμνασίου Ιλίου

Περίληψη

Σε αυτό το άρθρο παρουσιάζεται η εφαρμογή ανοιχτού κώδικα PhysicsCPP, (πρόγραμμα και προγραμματιστική βιβλιοθήκη) εκπαιδευτικών φυσικών παρουσιάσεω ν, με την υποστήριξη πολλαπλών πλατφορμών (έχει δοκιμαστεί σε Windows και διανομές Linux), και χαρακτηριστικά τη μεγάλη επεκτασιμότητα, την επιλογή γλώσσας χρήστη (προς το παρόν μεταξύ αγγλικών και ελληνικών), την ευκολία χρήσης για προγραμματιστές και χρήστες εξίσου, και το προσαρμόσιμο γραφικό περιβάλλον. Η εφαρμογή αναπτύχθηκε σε Γενικό Λύκειο ως αποτέλεσμα συνεργασίας μαθητών και εκπαιδευτικών στο πλαίσιο των μαθημάτων της Φυσικής και της Πληροφορικής. Επιπλέον, αναλύονται η σχεδίαση του προγράμματος, όπως και τα χαρακτηριστικά και οι λεπτομέρειες υλοποίησής του, επισημαίνοντας τη δυνατότητα χρήσης του για γρήγορη, εύκολη δημιουργία εκπαιδευτικών παρουσιάσεων σχετικά με τη Φυσική, και συγκεκριμένα τη Μηχανική Α΄ Λυκείου. Η προτεινόμενη εφαρμογή έρχεται να καλύψει ένα κενό εφαρμογών ανοιχτού κώδικα με αντικείμενο τη Φυσική Λυκείου, με δεδομένο ότι τα προτεινόμενα, με βάση το Πρόγραμμα Σπουδών Φυσικής Λυκείου, προγράμματα είναι κυρίως κλειστού κώδικα με προκατασκευασμένα σενάρια. Η εν λόγω εφαρμογή αναπτύχθηκε σε C++ και ο πηγαίος κώδικας παρέχεται στον σύνδεσμο GitHub: https://github.com/undefinedpp/physics.

Λέξεις κλειδιά: C++, Φυσική, ανοιχτός κώδικας, προσομοιώσεις

Εισαγωγή

Στις περισσότερες χώρες τα εθνικά Προγράμματα Σπουδών έχουν ενσωματώσει τη διδασκαλία των ΤΠΕ (Τεχνολογίες Πληροφοριών και Επικοινωνιών) στα σχολεία. Ωστόσο, για πολλά χρόνια, είχαν επικεντρωθεί κυρίως στη διδασκαλία βασικών δεξιοτήτων Η/Υ, όπως στην επεξεργασία κειμένου, τη χρήση email, τη χρήση προγραμμάτων γραφικών, την επικοινωνία με χρήση Email και Chat, την αναζήτηση πληροφοριών μέσω Διαδικτύου, και όχι στη διδασκαλία της υπολογιστικής σκέψης, η οποία είναι μια πολύ σημαντική δεξιότητα του 21ου αιώνα. Έτσι, μία από τις σημαντικότερες προκλήσεις της εκπαίδευσης είναι η διαμόρφωση της σωστής υπολογιστικής σκέψης. Σύμφωνα με το K-12 Πλαίσιο της Επιστήμης Υπολογιστών (k12cs.org), ο όρος υπολογιστική σκέψη αναφέρεται στις «διαδικασίες σκέψης που εμπλέκονται στην έκφραση λύσεων ως υπολογιστικά βήματα ή αλγόριθμοι που μπορούν να εκτελεστούν από υπολογιστή».

Η υπολογιστική σκέψη (CT) βασίζεται σε έννοιες και πρακτικές που είναι θεμελιώδεις για τους υπολογιστές. Περιλαμβάνει γνωσιολογικές και αναπαραστατικές πρακτικές, όπως πρόβλημα, αναπαράσταση, αφαίρεση, αποσύνθεση, προσομοίωση, επαλήθευση και

πρόβλεψη. Αυτές οι πρακτικές είναι επίσης κεντρικές για την ανάπτυξη εμπειρογνωμοσύνης και σε επιστημονικούς και σε μαθηματικούς κλάδους.
Στο πλαίσιο των σπουδών στο Λύκειο και για την προετοιμασία των μαθητών/-τριών προκειμένου να συμμετέχουν ενεργά στην κοινωνία του 21ου αιώνα (Bocconi et al., 2016· Voogt et al., 2015· Wing, 2008) η υπολογιστική σκέψη είναι απαραίτητη για την ατομική εξέλιξη και την αποτελεσματικότητα των μαθητών/-τριών.

Η αποτελεσματική χρήση υπολογιστικών εργαλείων για την επίλυση προβλημάτων και η έμφαση στην καλλιέργεια της υπολογιστικής σκέψης στις τάξεις του Λυκείου έχει στόχο να οικοδομήσει σημαντικές ικανότητες για τη ζωή των νέων και των εκπαιδευτικών. Είναι δυνατόν να δώσει αυξημένες ευκαιρίες στους/στις μαθητές/-τριες, αλλά και στους/στις εκπαιδευτικούς, ώστε να είναι ενημερωμένοι χρήστες και να γίνονται κριτικά σκεπτόμενοι δημιουργοί μέσω των ψηφιακών τεχνολογιών. Αναπτύσσοντας την υπολογιστική σκέψη και χρησιμοποιώντας τρόπους λογικής αφαίρεσης σε πολλαπλά επίπεδα (δεδομένων, αναπαραστάσεων, λογικών διαδικασιών κ.λπ.) στόχος είναι να δημιουργηθούν, να αναπτυχθούν και να βελτιωθούν μοντέλα διαχείρισης πληροφοριών και αλγόριθμοι επίλυσης αυθεντικών προβλημάτων. Η κατανόηση της λειτουργίας υπολογιστικών και προγραμματιστικών εργαλείων θα οδηγήσει στην αξιοποίησή τους σε προγράμματα εμπνευσμένα από μαθητές/-τριες με αποτέλεσμα την επίλυση προβλημάτων του πραγματικού κόσμου.

Είναι γεγονός ότι οι μαθητές/-τριες έχουν ένα ευρύ φάσμα ενδιαφερόντων και ταλέντων και είναι ωφέλιμο να τους δίνεται η δυνατότητα να αποκτήσουν και βασικές γνώσεις προγραμματισμού, ακόμα και αν δε θέλουν να γίνουν μηχανικοί λογισμικού. Άλλωστε η κατοχή βασικών δεξιοτήτων συγγραφής κώδικα προετοιμάζει τους/τις μαθητές/-τριες για οποιαδήποτε σταδιοδρομία ακόμη και εκτός STEM. Όπως η ανάγνωση ή η γραφή, έτσι και ο προγραμματισμός είναι ένας αλφαβητισμός για τον 21ο αιώνα. Η κατανόηση του τρόπου με τον οποίο προγραμματίζονται οι υπολογιστές θα επιτρέψει τόσο στους/στις μαθητές/-τριες όσο στους/στις εκπαιδευτικούς να αντιληφθούν καλύτερα την τεχνολογία και να συνειδητοποιήσουν τον αντίκτυπο της τεχνολογικής προόδου στην οικονομία και τον κόσμο. Περαιτέρω, ο προγραμματισμός βοηθάει στην ενίσχυση της λογικής σκέψης. Οι μαθητές/-τριες μαθαίνουν πώς να προσεγγίζουν τα προβλήματα συστηματικά, πώς να εφαρμόζουν και να δοκιμάζουν τον κώδικά τους βήμα-βήμα. Επιπλέον, διδάσκει τους/τις μαθητές/-τριες να σκέφτονται δημιουργικά για πρωτότυπες λύσεις σε προβλήματα. Αυτός ο τύπος δημιουργικότητας είναι ευρέως εφαρμόσιμος σε κάθε γνωσιακό αντικείμενο. Αν και στη Δευτεροβάθμια Εκπαίδευση η γλώσσα προγραμματισμού που μαθαίνουν οι μαθητές/-τριες είναι η Python, ως μια γλώσσα σαφής με κατανοητή σύνταξη που διευκολύνει την εστίαση στη λογική πρακτική αντί για τις λεπτομέρειες, για το πρακτικό μέρος αυτής της εργασίας επιλέχτηκε η C++ ως μια γλώσσα αντικειμενοστρεφούς φύσεως που επιτρέπει ακόμη και σε αρχάριους προγραμματιστές να κατασκευάζουν μη τετριμμένα προγράμματα σε σύντομο χρονικό διάστημα χρησιμοποιώντας έτοιμες βιβλιοθήκες.

Η C++ είναι μια από τις πιο δημοφιλείς γλώσσες προγραμματισμού υπολογιστών στον κόσμο. Χρησιμοποιείται για την ανάπτυξη πολλών πραγμάτων, από διασκεδαστικά βιντεοπαιχνίδια, ιστότοπους, τεχνητή νοημοσύνη μέχρι μεγάλα λειτουργικά συστήματα, όπως τα Microsoft Windows και το iOS της Apple. Η C++ είναι μια γλώσσα που έχει προέλθει από τη C, αλλά με ποικίλα πλεονεκτήματα σε σχέση με αυτήν. Αναλυτικά, δίνει τη δυνατότητα οργάνωσης των δεδομένων σε μια κλάση (class) και υποστηρίζει την κληρονομικότητα (inheritance) στη δομή των κλάσεων. Επιπλέον, παρέχει δυνατότητα αρχικοποίησης των δεδομένων και μπορεί να πραγματοποιηθεί ορισμός μεταβλητών ως αναφορά (reference) σε κάποια άλλη. Υποστηρίζει την έννοια του πολυμορφισμού

(polymorphic) στον καθορισμό των συναρτήσεων με βάση τον τύπο του ορίσματός τους, καθώς και τη δυνατότητα ορισμού πολυμορφικών βασικών τύπων με βάση ένα πρότυπο (template). Παρέχει πλούσια βιβλιοθήκη βασικών δομών δεδομένων και μπορεί να γίνει υπερφόρτωση (overloading) τελεστών. Όλα τα παραπάνω στοιχεία επιτρέπουν στη γλώσσα να υποστηρίζει τον αντικειμενοστρεφή προγραμματισμό (object-oriented programming).

Στο πλαίσιο της προτεινόμενης εφαρμογής, η υπολογιστική σκέψη αναφέρεται στην ενοποιητική δεξιότητα, ικανή να συνδυάζει την ανακαλυπτική διαδικασία με την ανάπτυξη και εφαρμογή υπολογιστικών μοντέλων και προσομοιώσεων (Dolgopolovas, & Dagienė, 2021). Η ανάπτυξη των απαραίτητων υπολογιστικών δεξιοτήτων πραγματοποιήθηκε στο πλαίσιο του μαθήματος της Φυσικής Α΄ Λυκείου. Η αλγοριθμική σκέψη αναφέρεται στην ικανότητα δημιουργίας των βημάτων που απαιτούνται για τη μετατροπή μιας συγκεκριμένης εισόδου σε μια επιθυμητή έξοδο (Doleck et al., 2017). Στην προτεινόμενη υλοποίηση, η ανάπτυξη των αλγοριθμικών δεξιοτήτων πραγματοποιήθηκε στο πλαίσιο του μαθήματος της Πληροφορικής. Αν και οι δυο δεξιότητες αλληλεπικαλύπτονται, όταν η μόνη πηγή δράσεων είναι το πλαίσιο της διδασκαλίας, η παράλληλη ανάπτυξη εφαρμογής λογισμικού είναι σε θέση να τις διαχωρίσει, όπως δείχνει ο πίνακας 1.

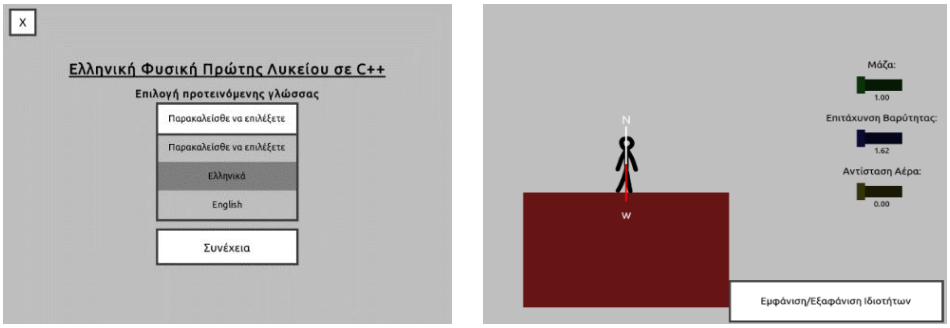
Πίνακας 1. Ενδεικτικές εργασίες υπολογιστικής και αλγοριθμικής σκέψης

Υπολογιστική σκέψη	Αλγοριθμική σκέψη
Δημιουργήστε μια λίστα διακριτών βημάτων για τον υπολογισμό του βάρους ενός σώματος	Καθορίστε τις εισόδους που πρέπει να έχει μια κλάση, ώστε η έξοδος να δίνει το βάρος ενός σώματος σε Ν
Να περιγράψετε μια διαδικασία διακριτών βημάτων προσδιορισμού και σχεδιασμού των δυνάμεων που ασκούνται σε ένα σώμα που ισορροπεί πάνω σε ένα τραπέζι	Να καθορίσετε μια διαδικασία μέσω βιβλιοθηκών και κλάσεων, ώστε να είναι δυνατή η απεικόνιση διανυσμάτων στην οθόνη
Να δώσετε σειρά βημάτων για τον υπολογισμό και τη γραφική απεικόνιση της αντίδρασης δοσμένης δύναμης	Να καθορίσετε μια διαδικασία, ώστε η εμφάνιση ή όχι της αντίδρασης μιας δύναμης στην οθόνη να είναι επιλέξιμη από τον χρήστη

Υλοποίηση

Αρχικά, για την επίτευξη ενός γρήγορου, ισχυρού, αξιόπιστου και ευέλικτου συστήματος, έπρεπε να επιλεγεί η C++, καθώς είναι γλώσσα προγραμματισμού που εμφανίζει πολύ μεγάλες αποδόσεις, με την προϋπόθεση σωστής εφαρμογής. Εκτός αυτού, βασιζόμενη στη γλώσσα C και στενά συνδεδεμένη με αυτήν, εμφανίζει εντυπωσιακά μεγάλη υποστήριξη και συμβασιμότητα, χαρακτηριστικά αρκετά χρήσιμα για κάθε project ανοιχτού κώδικα ειδικά για την εκπαίδευση όπου υπάρχει πολύ μεγάλη διαφοροποίηση τεχνικού εξοπλισμού και πληροφοριακών συστημάτων. Συνεχίζοντας, ελήφθη απόφαση να χρησιμοποιηθεί δομοστοιχειωτή λογική κώδικα, με σκοπό την επεκτασιμότητα και την πρόκληση του ενδιαφέροντος των προγραμματιστών. Όσον αφορά το σύστημα παραγωγής ενοτήτων (buildsystem), επιλέχθηκε το CMake, υποστηριζόμενο από την πλειονότητα των προγραμματιστικών βιβλιοθηκών, αποτελώντας ένα είδος «σύμβασης» μεταξύ των προγραμματιστών. Επιπλέον, όσον αφορά την υλοποίηση, ακολουθούνται πολλές αρχές του αντικειμενοστρεφούς τρόπου προγραμματισμού, καθώς είναι πιο κατανοητή μέθοδος, βασιζόμενη περισσότερο στην κοινή λογική σε σχέση με άλλες εναλλακτικές. Επίσης,

χρησιμοποιούνται πολλά πρότυπα κώδικα, όπως «DependencyInjection», «Singleton», πολυμορφισμός και αλυσιδωτές μέθοδοι. Τέλος, γίνεται χρήση της βιβλιοθήκης SFML, για φόρτωση εικόνων, γραμματοσειρών, πολυμέσων και λειτουργίες γραφικών, αποτελώντας βάση της βιβλιοθήκης του προγράμματος (βλ. Σχήμα 1).



Σχήμα 1. Ενδεικτικό περιβάλλον διάδρασης της εφαρμογής

Η βάση του προγράμματος είναι η μονάδα «συστήματος». Το πρόγραμμα επικεντρώνεται στην αφηρημένη κλάση της Application, από την οποία οφείλει κάθε client να κληρονομήσει. Περιέχει τις εικονικές μεθόδους «OnCreate» και «OnUpdate» που αντιστοιχούν στα συμβάντα αρχής (μετά από ρύθμιση βασικών παραγόντων) και «ενημέρωσης οθόνης» του προγράμματος. Αυτές μπορούν να επαναπροσδιοριστούν σε περίπτωση που θέλει ο χρήστης να αποδώσει κώδικα στα συμβάντα. Επίσης, περιέχει τη μέθοδο «Start» που δέχεται τις κατάλληλες παραμέτρους για να αρχίσει το πρόγραμμα (π.χ. μέγεθος και όνομα παραθύρου). Επιπλέον, περιέχει ένα «stack» από καταστατικές δομές που αντιστοιχούν στην κατάσταση της εφαρμογής, η καθεμία από τις οποίες έχει δική της λειτουργικότητα, με τη μορφή εικονικών μεθόδων ξανά αντιστοίχων με συμβάντα, όπως «OnCreate» κατά τη δημιουργία της δομής κατάστασης, «OnUpdate» για την ενημέρωση οθόνης κατά την κατάσταση, «OnShow» για κάθε εμφάνιση της κατάστασης, «OnHide» για απενεργοποίησή της και «OnDestroy» για τη διαγραφή της. Στην κλάση της εφαρμογής παρέχεται επίσης δείκτης στο παράθυρο, προς χρήση για γραφικά και άλλες δυνατότητες. Η μονάδα «συστήματος» περιέχει και κλάσεις για μια πληθώρα εργαλείων, όπως χρώματα, εντοπισμό σφαλμάτων, βελτιωμένη πρόσβαση στη θέση και την κατάσταση δείκτη ποντικιού, όπως και σε «παράθυρα μηνυμάτων», «γραμματοσειρές», «χρώματα», υποστήριξη πολλαπλών γλωσσών, όπως αγγλικά και ελληνικά, και κάποια μαθηματικά εργαλεία (εντοπισμός διασταύρωσης ορθογωνίων παραλληλογράμμων περιοχών).

Σημαντική είναι η εύκολη αλληλεπίδραση χρήστη, και για αυτό παρέχεται πλήρως προσαρμοσμένη βιβλιοθήκη γραφικού περιβάλλοντος. Περιλαμβάνονται γραφικά στοιχεία, όπως κουμπιά, «ετικέτες» κειμένου, οργανωμένες συγκεντρώσεις στοιχείων ή «Layouts» (Vertical/HorizontalLayout= Κάθετα/οριζόντια κατανεμημένη απλή συγκέντρωση, Vertical/HorizontalGrid= κάθετο/οριζόντιο πλέγμα), εμφανιζόμενες λίστες πολλαπλών επιλογών, ολισθητές επιλογής τιμών, πεδίων εισαγωγής τιμών. Η προσθήκη περαιτέρω στοιχείων καθίσταται εύκολη, καθώς όλα τα στοιχεία κληρονομούν από την κλάση «UIElement<Impl>», η οποία παρέχει την κατάλληλη βασική λειτουργικότητα, απαραίτητη για κάθε οπτικό αλληλεπιδράσιμο, προσαρμόσιμο γραφικό στοιχείο. Για την επίτευξη δημιουργίας συνόλων (όπως λίστες) με γραφικά στοιχεία, το «DependencyInjection» της κλάσης δε βοηθά. Για αυτό, δημιουργήθηκε η κλάση UIElementAbstract, η οποία περιέχει μη-

εξατομικευμένου –ανά στοιχείο– τύπου λειτουργίες, όπως αλλαγή μεγέθους, τοποθεσίας, χρώματος, κ.λπ. Επίσης η UIElementAbstract κληρονομείται από την UIElement<Impl>.

Στη μονάδα της Φυσικής παρέχεται η κλάση του σώματος, η οποία μέσω κληρονομικότητας χωρίζεται σε κινητικά και στατικά σώματα. Τα κινούμενα σώματα έχουν καθορισμένη μάζα και επηρεάζονται σύμφωνα με τους νόμους του Νεύτωνα. Η επίδραση δυνάμεων σε αυτά είναι μετρήσιμη, συμπεριλαμβανομένης αυτής του βάρους. Επίσης, σε αυτά υπάρχουν οι προσαρμοζόμενες μεταβλητές ιδιότητες της μάζας, της θέσης, της ταχύτητας, της επιτάχυνσης, και άλλων. Σε αντίθεση, τα στατικά σώματα, όπως το όνομά τους μαρτυρά, είναι αυτά για τα οποία η μάζα τους είναι τόσο μεγάλη, έτσι ώστε κάθε μεταβολή σε αυτά να θεωρείται αμελητέα και να μην υπολογίζεται (π.χ. η γη ως το κέντρο της παρουσίασης, καθώς δε χρειάζεται η προσομοίωση όλου του ηλιακού συστήματος ή του παρατηρήσιμου σύμπαντος). Δυνάμεις που ασκούνται σε αυτά αγνοούνται, αφού η εμφάνιση ή η μέτρηση της επίδρασής τους δεν είναι χρήσιμη, και έτσι, υπολογιστικός χρόνος και υπολογιστική ισχύς δε σπαταλώνται.

Επιπλέον, εμπεριέχεται η κλάση «επίλυσης σωμάτων» η οποία αντιμετωπίζει αλληλεπιδράσεις μεταξύ σωμάτων, ανεξάρτητα από το αν πρόκειται για κινητικά ή στατικά σώματα. Συγκεκριμένα, αντιμετωπίζει συγκρούσεις (οριζόμενη η ανταπόκριση από τον χρήστη προς το παρόν), και εφαρμόζει κατάλληλα κάθε μεταβολή που πρέπει να εφαρμοστεί στα σώματα.

Τέλος, περιλαμβάνεται και ρυθμιζόμενο εργαλείο μετατροπής φυσικών μονάδων σε εικονικές (π.χ. newton σε εικονοστοιχεία) και αποθήκευσης «παγκόσμιων» τιμών, όπως η επιτάχυνση της βαρύτητας. Η χρήση πολυγλωσσίας του προγράμματος δίνει δυνατότητα εμπλοκής στην εφαρμογή μαθητών/-τριών και εκπαιδευτικών από άλλες χώρες.

Προγραμματισικό Παράδειγμα Δημιουργίας Παράστασης

Ας υποθέσουμε πως θέλουμε να δημιουργήσουμε μία παρουσίαση προγραμματιστικά μέσω της βιβλιοθήκης του προγράμματος, η οποία θα συμπεριλαμβάνει έναν άνθρωπο να στέκεται πάνω στο έδαφος, με τη δυνατότητα μέτρησης των υπαρχουσών δυνάμεων, που θα εμφανίζονται οπτικά ως διανύσματα. Αυτό υλοποιείται κάπως έτσι: (παρέχεται η εικόνα 70x150 “assets/images/person.png”).

Ο χρήστης κληρονομεί από την κλάση Application της βιβλιοθήκης μας, από την οποία πρέπει να κάνει «override» τις μεθόδους OnStartκαι OnUpdate, για να παρέχει λειτουργικότητα στην εφαρμογή. Πρέπει κατά τη δημιουργία της εφαρμογής, να κληρονομήσει από τη γενικευμένη κλάση κατάστασης εφαρμογής, παρέχοντάς της λειτουργικότητα με τις εικονικές μεθόδους, όμοια με τις OnStart και OnUpdate της Application, μόνο με μεγαλύτερη λειτουργικότητα. Πρέπει παράλληλα στο workingdirectory να δημιουργηθεί φάκελος assets, στον οποίο θα τοποθετηθούν οποιαδήποτε πολυμέσα είναι απαραίτητα για την παρουσίαση, όπως εικόνες, βίντεο, και άλλα. Όσον αφορά το γραφικό περιβάλλον, μπορούν να χρησιμοποιηθούν κλάσεις κουμπιών, κειμένου και άλλων γραφικών στοιχείων, που παρέχουμε στη βιβλιοθήκη, τα οποία δημιουργούνται λαμβάνοντας δείκτη στην τρέχουσα εφαρμογή, και μέσω αυτής ενημερώνονται και εμφανίζονται στην οθόνη. Αυτό επιτυγχάνεται με τις μεθόδους Update και Draw, οι οποίες καλούνται μέσα στην OnUpdate της κατάστασης της εφαρμογής.

Για την προσθήκη σωμάτων, παρέχονται κλάσεις στατικών και δυναμικών σωμάτων, οι οποίες αντιμετωπίζονται όμοια με τα στοιχεία γραφικού περιβάλλοντος, με τη διαφορά πως μπορούν να δοθούν σε επιλυτή σωμάτων, με την κλάση BodyHandler, ο οποίος περιέχει το μαθηματικό μοντέλο του φυσικού συστήματος. Στην πρώτη έκδοση, χρησιμοποιείται δυσδιάστατος Νευτώνιος επιλυτής. Μελλοντικά, μπορεί να αναβαθμιστεί με τη χρήση και

τρίτης διάστασης, ενώ σε ενδεχόμενη τροποποίηση του Προγράμματος Σπουδών μπορούμε να προσθέσουμε για παράδειγμα και επιλυτή σχετικότητας. Ο επιλυτής εμφανίζεται και ενημερώνεται παρόμοια με τα στοιχεία γραφικού περιβάλλοντος, εμφανίζοντας και ενημερώνοντας όλα τα σώματα που έχουν προστεθεί σε αυτόν. Επιπρόσθετα, απελευθερώνει τη μνήμη αποδιδόμενη σε αυτά τα σώματα, έτσι ώστε ο χρήστης να μη χρειάζεται να ανησυχεί για θέματα αυτής της φύσης. Μετά από κάθε ενημέρωση του επιλυτή, δημιουργούνται αλληλεπιδράσεις προγραμματιστικά (με αντικείμενα - objects) μεταξύ των σωμάτων, όπως τριβές και σχέσεις δράσης αντίδρασης.

Συνεργασία

Σε περίπτωση που ένας προγραμματιστής ενδιαφέρεται να συμβάλει στη βιβλιοθήκη ή στην εφαρμογή του προγράμματος, είναι ελεύθερος να δημιουργήσει «fork» του πρότζεκτ στο GitHub, το οποίο θα περιέχει την ίδια άδεια χρήσης με αυτό, κάνοντας όποιες αλλαγές επιθυμεί. Αν θεωρήσει πως οι αλλαγές του πρέπει να προστεθούν στο κύριο πρόγραμμα, μπορεί να κάνει «pullrequest». Αυτό θα τεθεί σε κρίση βάσει μιας πληθώρας παραγόντων, όπως διατήρηση της αρχικής λειτουργικότητας του κώδικα, διατήρηση της σύμβασής του, ταχύτητας και άλλων. Αν αυτό εγκριθεί, ο προγραμματιστής θεωρείται επίσημα συνεργάτης. Αν όχι, θα υπάρξει απάντηση που πληροφορεί σχετικά με τις απαραίτητες αλλαγές, ώστε να γίνει μελλοντική έγκριση του pullrequest, ή θα δοθεί σημαντικός λόγος ώστε να μην προστεθεί ο υποψήφιος κώδικας.

Μελλοντικές Επεκτάσεις και Προσθήκες

Αν και το όλο πρότζεκτ στοχεύει στη συνεργασία Φυσικής και Προγραμματισμού, είναι στις προθέσεις μας να εκπονήσουμε ένα καθαρά γραφικό περιβάλλον δημιουργίας παρουσιάσεων, το οποίο δε θα απαιτεί προγραμματιστικές γνώσεις. Επίσης, έχει τεθεί στόχος για επέκταση και υποστήριξη πολλών γλωσσών, ευρωπαϊκών και μη, με σκοπό να υπάρχει η δυνατότητα χρήσης του προγράμματος, δίχως ιδιαίτερες ανησυχίες ως προς το εμπόδιο γλώσσας. Για αυτό και υποστηρίζονται ήδη οι γλώσσες των Ελληνικών και των Αγγλικών. Όσον αφορά τον κώδικα, στοχεύουμε να γενικεύσουμε τη χρήση γραφικών, έτσι ώστε να μην είναι απαραίτητη η χρήση της SFML, αλλά να υπάρχει και υποστήριξη για άλλες, λιγότερο αφηρημένες βιβλιοθήκες, όπως OpenGLκαι Vulcan. Παρόμοια εξέλιξη θέλουμε να έχει και το μαθηματικό τμήμα της βιβλιοθήκης (διανύσματα, πίνακες, κ.ά.), το οποίο θα έχει καλύτερη ενσωμάτωση σε αυτή.

Συμπεράσματα

Η εφαρμογή αυτή παρέχει διπλή παιδαγωγική αξία. Αποτελεί μία πολύ καλή εισαγωγή στον προγραμματισμό υψηλού επιπέδου, τόσο για μαθητές/-τριες όσο και για εκπαιδευτικούς, ενώ η συμβολή της Φυσικής την ανάγει σε χρηστική στη διδασκαλία και στην εκμάθησή της ως γνωστικού αντικειμένου. Η επεκτασιμότητα του προγράμματος μπορεί να εμπλέξει μαθητές/-τριες, προγραμματιστές και εκπαιδευτικούς σε μια κοινότητα ανοιχτού κώδικα, κάτι που είναι σύμφωνο με τη φιλοσοφία των νέων Προγραμμάτων Σπουδών για το Λύκειο. Από τεχνικής πλευράς, η επιλογή της C++ εγγυάται σε μεγάλο βαθμό τη μελλοντική λειτουργικότητα δίχως

απαξίωση του κώδικα, λόγω συχνών παραλλαγών των κανόνων της γλώσσας. Επίσης, εφαρμογές αυτού του είδους αποτελούν συνεισφορά προς την κατεύθυνση της τεχνολογικής αυτονόμησης των εκπαιδευτικών.

Αναφορές

Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., Engelhardt, K. (2016). Developing computational thinking in compulsory education – Implications for policy and practice. European Commission, Joint Research Centre.

Doleck, T., Bazelais, P., Lemay, D. J., Saxena, A., & Basnet, R. B. (2017). Algorithmic thinking, cooperativity, creativity, critical thinking, and problem solving: Exploring the relationship between computational thinking skills and academic performance. Journal of Computers in Education, 4(4), 355–369. https://doi.org/10.1007/s40692-017-0090-9.

Dolgopolovas, V., & Dagienė, V. (2021). Computational thinking: Enhancing STEAM and engineering education, from theory to practice. Computer Applications in Engineering Education, 29(1), 5–11. https://doi.org/10.1002/cae.22382.

Katalin Harangusa, Zoltán Kátaia (2019). Computational Thinking in Secondary and Higher Education.

Pratim Sengupta, John S. Kinnebrew, SatabdiBasu, Gautam Biswas, Douglas Clark (2013). Integrating Computational Thinking with K-12 Science Education Using Agent-based Computation: A Theoretical Framework.

Voogt, J., Fisser, P., Good, J., Mishra, P. & Yadav, A. (2015). Computational thinking in compulsory education: Towards an agenda for research and practice. Education & Information Technology, 20(4), 715-728.

Wing, J. M. (2008). Computational thinking and thinking about computing. Philosophical Transactions of the Royal Society A, 36, 3717-3725.