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

HerTour4Youth

2020-1-UK01-KA227-YOU-094543

IO1- A3: Σύνοψη Ψηφιακού Σχεδιασμού
Ενότητα: Προγραμματισμός Η/Υ

KA2 - **Συνεργασία για καινοτομία και ανταλλαγή καλών πρακτικών**
Συνεργασία για δημιουργικότητα

Έκδοση	Ημερομηνία	Συγγραφέας	Περιγραφή	Δράση	Σελίδες
1.0	15/10/2021	CIVIC	Δημιουργία (Creation)	C	12

ΙΣΤΟΡΙΚΟ ΑΝΑΘΕΩΡΗΣΗΣ

(*) Δράση: C = Δημιουργία (Creation), I = Εισαγωγή (Insert), U = Ενημέρωση (Update),
R = Αντικατάσταση (Replace), D = Διαγραφή (Delete)

ΕΓΓΡΑΦΑ ΑΝΑΦΟΡΑΣ

ID	Αναφορά		Τίτλος
1	2020-1-UK01-KA227-YOU-094543		Πρόταση HerTour4Youth
2			

ΙΣΧΥΟΝΤΑ ΕΓΓΡΑΦΑ

ID	Αναφορά		Τίτλος
1	Deliverable IO1.A2		Εκπόνηση Εκπαιδευτικής Μεθοδολογίας (Elaboration of Training Methodology)
2			

Περιεχόμενα

1. Εισαγωγή ..... **Σφάλμα! Δεν έχει οριστεί σελιδοδείκτης.**
1.1 Μαθησιακά Αποτελέσματα 4
1.2 Λέξεις κλειδιά ..... **Σφάλμα! Δεν έχει οριστεί σελιδοδείκτης.**
1.3 Εκτιμώμενος χρόνος παρακολούθησης 4
1.4 Γλωσσάρι όρων ..... **Σφάλμα! Δεν έχει οριστεί σελιδοδείκτης.**
2. Προγραμματισμός Η/Υ ..... **Σφάλμα! Δεν έχει οριστεί σελιδοδείκτης.**
2.1 Εισαγωγή ..... **Σφάλμα! Δεν έχει οριστεί σελιδοδείκτης.**
2.2 Προγραμματιστής υπολογιστών ..... **Σφάλμα! Δεν έχει οριστεί σελιδοδείκτης.**
2.1.1 Γλώσσες προγραμματισμού 9
2.1.2 Διαδικασία προγραμματισμού 12
2.1.3 Αρχές προγραμματισμού 15
2.1.4 Πώς μπορείτε να γίνετε προγραμματιστής υπολογιστών; 16
3. Αξιολόγηση ..... **Σφάλμα! Δεν έχει οριστεί σελιδοδείκτης.**
3.1 Αξιολόγηση γνώσεων ..... **Σφάλμα! Δεν έχει οριστεί σελιδοδείκτης.**
3.2 Αξιολόγηση δεξιοτήτων ..... **Σφάλμα! Δεν έχει οριστεί σελιδοδείκτης.**
4. Παραπομπές 21

1. Εισαγωγή

Αν αναρωτιέστε τι είναι ο προγραμματισμός υπολογιστών, απλά φανταστείτε ένα πολύ απλό μηχάνημα υπολογιστή στο σπίτι σας (μπορεί να είναι ένας θερμοστάτης, ο δρομολογητής σύνδεσης κ.λπ.). Τώρα, θα πρέπει να ξέρετε ότι αυτός ο υπολογιστής δεν λειτουργεί μόνος του, αλλά έχει προγραμματιστεί από κάποιον. Δηλαδή ότι υπάρχει ένας προγραμματιστής υπολογιστών. Κάθε υπολογιστής χρειάζεται ένα σύνολο οδηγιών ώστε να λειτουργεί ομαλά. Αυτό είναι το νόημα του προγραμματισμού υπολογιστών: μπορεί να αποτελείται από ένα απλό σύνολο οδηγιών για τη διευκόλυνση συγκεκριμένων ενεργειών σε ένα περίπλοκο σύνολο εντολών που περιλαμβάνει ανάγνωση και ταξινόμηση δεδομένων.

1.1 Μαθησιακά Αποτελέσματα

Μετά την ολοκλήρωση αυτής της ενότητας, θα είστε σε θέση:

-Να έχετε μια βασική κατανόηση των αρχών προγραμματισμού
-Να έχετε μια πιο ουσιαστική γνώση του τρόπου λειτουργίας των εργαλείων Τεχνολογίας πληροφοριών και επικοινωνίας (ΤΠΕ)
-Να μπορείτε να εφαρμόζετε τις αρχές προγραμματισμού

1.2 Λέξεις Κλειδιά

- Τεχνολογίες πληροφοριών και επικοινωνίας (ΤΠΕ)
- Προγραμματιστής υπολογιστών
- Γλώσσες προγραμματισμού

1.3 Εκτιμώμενος χρόνος παρακολούθησης

1 ώρα.

1.4 Γλωσσάρι όρων

ΤΠΕ: σημαίνει «Τεχνολογίες Πληροφορικής και Επικοινωνίας». Οι ΤΠΕ αναφέρονται σε τεχνολογίες που παρέχουν πρόσβαση σε πληροφορίες μέσω των τηλεπικοινωνιών. (TechTerms, 2021)

Προγραμματιστής υπολογιστών: ο προγραμματιστής υπολογιστών είναι το άτομο που είναι υπεύθυνο για τη δημιουργία των οδηγιών που θα εκτελούνται από έναν υπολογιστή, γράφοντας και

δοκιμάζοντας έναν κώδικα που θα επιτρέπει σε εφαρμογές και προγράμματα λογισμικού να λειτουργούν με επιτυχία. (Techopedia, 2014)

Γλώσσες προγραμματισμού: οι γλώσσες προγραμματισμού υπολογιστών είναι οποιεσδήποτε από τις διάφορες γλώσσες για την έκφραση ενός συνόλου λεπτομερών οδηγιών για έναν ψηφιακό υπολογιστή.
(D. Hemmendinger, 2021)

Αναλογικό: Επίθετο που αναφέρεται σε μηχανισμό ή συσκευή στην οποία οι πληροφορίες αντιπροσωπεύονται από συνεχώς μεταβλητά φυσικά μεγέθη. Είναι το αντίθετο του ψηφιακού. (Merriam Webster, 2021)

Τεχνητή Νοημοσύνη: Η τεχνητή νοημοσύνη αξιοποιεί υπολογιστές και μηχανές για να μιμηθεί τις ικανότητες επίλυσης προβλημάτων και λήψης αποφάσεων του ανθρώπινου μυαλού, χωρίς να περιορίζεται σε μεθόδους που είναι βιολογικά παρατηρήσιμες. (IBM, 2020)

Μηχανική μάθηση: Η μηχανική μάθηση είναι ένας κλάδος της τεχνητής νοημοσύνης και της επιστήμης των υπολογιστών που εστιάζει στη χρήση δεδομένων και αλγορίθμων για τη μίμηση του τρόπου με τον οποίο μαθαίνουν οι άνθρωποι, βελτιώνοντας σταδιακά την ακρίβειά της.
(IBM, 2020)

Δυαδικό: Το δυαδικό είναι ένα σύστημα αριθμών βάσης-2 που εφευρέθηκε από τον Gottfried Leibniz και αποτελείται από μόνο δύο αριθμούς ή ψηφία: 0 (μηδέν) και 1 (ένα). Αυτό το σύστημα αρίθμησης είναι η βάση για όλο τον δυαδικό κώδικα, ο οποίος χρησιμοποιείται για την εγγραφή ψηφιακών δεδομένων, όπως οι οδηγίες του επεξεργαστή υπολογιστή που χρησιμοποιούνται καθημερινά. (ComputerHope, 2021)

2. Προγραμματισμός Η/Υ

2.1 Εισαγωγή

Στο σύγχρονο κόσμο που βασίζεται στους υπολογιστές, γνωρίζουμε ότι μπορούν να κάνουν πολλά αξιοσημείωτα πράγματα.

Η ταχεία τεχνολογική πρόοδος που έχουμε δει τις τελευταίες δεκαετίες έχει μια ευρεία επίδραση στον τρόπο που λειτουργούν τα πάντα. Πρέπει να δείτε κάτω από την επιφάνεια για να κατανοήσετε πού έγιναν οι πιο σημαντικές αλλαγές. Οι υπολογιστές αντικατέστησαν εκατομμύρια ώρες εργασίας και αποθήκες αναλογικών μηχανών με ταχύτερα, ασφαλέστερα και πιο αξιόπιστα συστήματα.

Δεδομένου ότι οι υπολογιστές λειτουργούν με κώδικα, είναι προφανές γιατί μπορείτε να τον βρείτε οπουδήποτε. Οι υπολογιστές θα συνεχίσουν να αντικαθιστούν την ξεπερασμένη τεχνολογία σε όλα, από τα μικροκύματα έως τις μονάδες παραγωγής ενέργειας. Και η παρουσία του κώδικα στην καθημερινή μας ζωή θα αυξηθεί.

Επίσης, σήμερα σχεδόν όλοι οι άνθρωποι διαθέτουν φορητό υπολογιστή (laptop) που μπορεί να χρησιμοποιηθεί για απλές διαδικασίες, όπως η δημιουργία υπολογιστικών φύλλων ή η δημιουργία απλών εγγράφων. Ωστόσο, υπάρχουν και άνθρωποι που χρησιμοποιούν το laptop τους με πιο περίπλοκο και επαγγελματικό τρόπο. Αυτό μπορεί να είναι αναγκαίο για την ολοκλήρωση εκατομμυρίων οικονομικών συναλλαγών που γίνονται κάθε μέρα και για τον έλεγχο των υποδομών που καθιστούν δυνατή τη σύγχρονη ζωή.

Αφού αυτά έγιναν κατανοητά, είναι σημαντικό να θυμάστε ότι κανένας υπολογιστής δεν μπορεί να κάνει τίποτα εάν δεν έχει προγραμματιστεί από έναν άνθρωπο. Σε αυτή την περίπτωση μιλάμε για τον προγραμματιστή υπολογιστών.

Ο προγραμματιστής υπολογιστών είναι η επαγγελματική φιγούρα που είναι υπεύθυνη για τη σύνταξη κωδίκων ή την κωδικοποίηση, που καθοδηγούν το πώς πρέπει να λειτουργεί ένας υπολογιστής, μια εφαρμογή ή ένα πρόγραμμα λογισμικού.

Όπως αναφέρθηκε προηγουμένως, αυτό το σύνολο εντολών μπορεί να είναι αρκετά απλό και η όλη διαδικασία κωδικοποίησης μπορεί να απαιτεί απλώς την προσθήκη δύο αριθμών, αλλά θα μπορούσε επίσης να περιλαμβάνει μια πιο περίπλοκη ανάγνωση δεδομένων από αισθητήρες θερμοκρασίας για τη ρύθμιση ενός θερμοστάτη, την ταξινόμηση δεδομένων για την ολοκλήρωση ενός περίπλοκου προγραμματισμού ή κρίσιμων αναφορών, ή τη «μεταφορά» παικτών σε κόσμους πολλαπλών επιπέδων και προκλήσεων στα παιχνίδια.

Ενώ η επιστήμη των υπολογιστών ασχολείται με υψηλού επιπέδου θεωρητικές ιδέες, σχεδόν κάθε πτυχή της σύγχρονης ζωής βασίζεται στην κωδικοποίηση. Κάθε εφαρμογή στο τηλέφωνο, το tablet ή τον υπολογιστή μας χρησιμοποιεί γλώσσες υπολογιστή προκειμένου να εκτελεστεί.

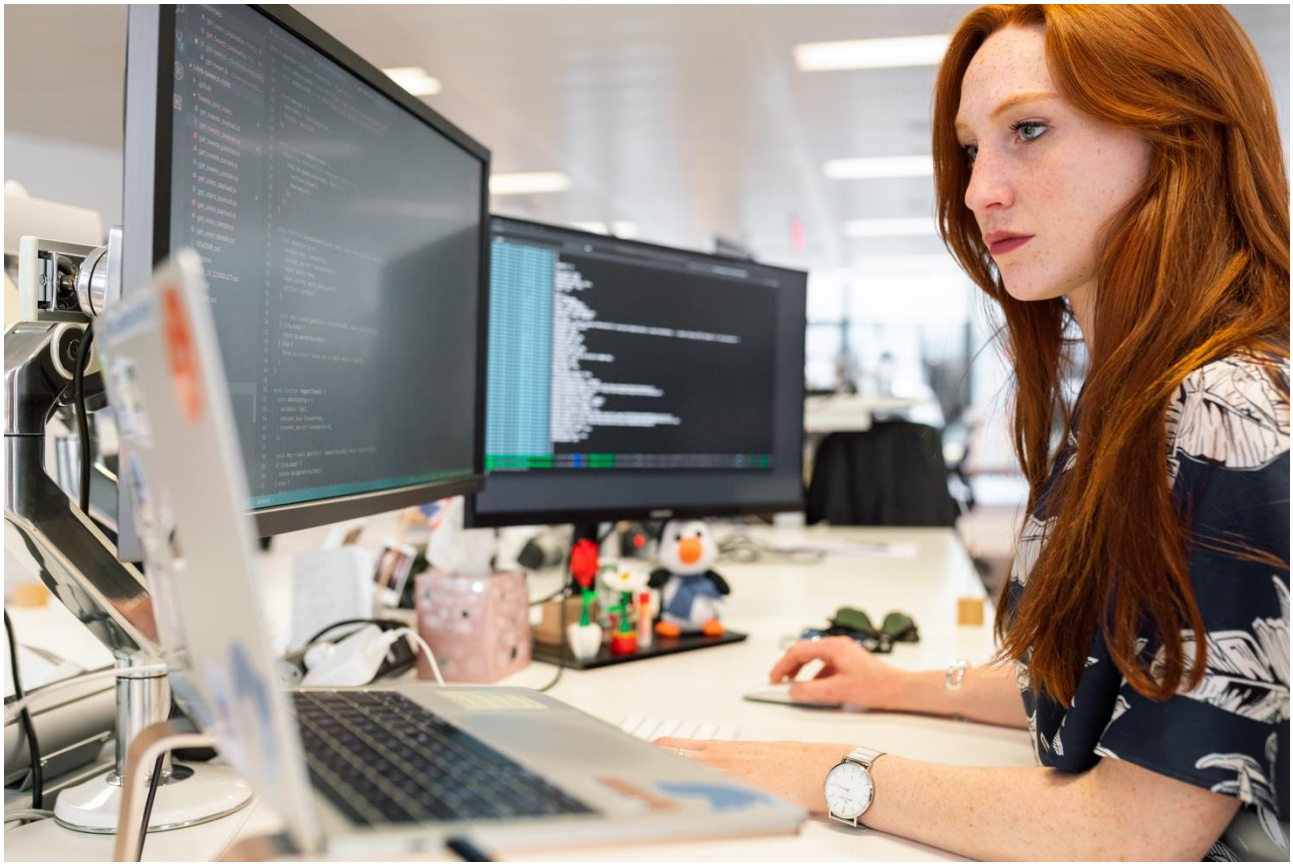
Πηγή: https://pixabay.com/photos/code-coding-computer-data-1839406/

Η κωδικοποίηση χρησιμοποιείται και από άλλα ψηφιακά συστήματα, όπως οι έξυπνες τηλεοράσεις και οι αριθμομηχανές. Σχεδόν κάθε νέο αυτοκίνητο την χρησιμοποιεί για να ελέγχει τα πάντα, από τα συστήματα κλιματισμού έως τα μπεκ ψεκασμού καυσίμου. Στις πόλεις χρησιμοποιούνται υπολογιστές για τη λειτουργία των σημάτων κυκλοφορίας.

Τα συστήματα που παλαιότερα ήταν αναλογικά βελτιστοποιήθηκαν με τη χρήση ηλεκτρονικών συστημάτων. Αυτό επιτρέπει στους μηχανικούς να κατασκευάσουν ένα πιο αποτελεσματικό και λιγότερο δαπανηρό σύστημα, δομή και μηχάνημα. Επιπλέον, ορισμένα από τα πιο προηγμένα τεχνικά πεδία (όπως η τεχνητή νοημοσύνη και η μηχανική μάθηση) χρησιμοποιούν κωδικοποίηση.[1]

[1] Weinstein, J. (2021, January 24). What Is Coding? Coding Definition and Uses. Career Karma. https://careerkarma.com/blog/what-is-coding-used-for/

2.2 Προγραμματιστής υπολογιστών



Πηγή: https://unsplash.com/photos/64YrPKiguAE

Όπως αναφέρθηκε προηγουμένως, ένας προγραμματιστής υπολογιστών είναι το άτομο που είναι υπεύθυνο για να δημιουργήσει τις οδηγίες για την εκτέλεση ενός υπολογιστή γράφοντας και δοκιμάζοντας τον κώδικα που επιτρέπει στις εφαρμογές και τα προγράμματα λογισμικού να λειτουργούν με επιτυχία.

Για να χαρακτηριστεί ένας προγραμματιστής υπολογιστών (computer programmer) χρησιμοποιούνται οι όροι programmer, coder, developer, ή software engineer, δηλαδή μηχανικός λογισμικού. Επίσης, ο όρος αυτός χρησιμοποιείται συχνά για να αναφερθεί σε έναν αυτόνομο προγραμματιστή λογισμικού, προγραμματιστή εφαρμογών για κινητά, προγραμματιστή Ιστού, αναλυτή λογισμικού, προγραμματιστή μόνιμου προγράμματος του λογισμικού κ.λπ.

Ένας προγραμματιστής υπολογιστών είναι ένας εξειδικευμένος επαγγελματίας που κωδικοποιεί, δοκιμάζει, διορθώνει και διατηρεί τις περιεκτικές οδηγίες, γνωστές ως προγράμματα υπολογιστών, που πρέπει να ακολουθούν οι υπολογιστές για να εκτελέσουν τις συγκεκριμένες λειτουργίες τους[2].

Μερικές από τις συνηθισμένες εργασίες που πρέπει να γνωρίζει καλά ένας προγραμματιστής υπολογιστών περιλαμβάνουν:

• Έλεγχος απόδοσης λογισμικού.
• Επίλυση προβλημάτων λογισμικού υπολογιστή.
• Τροποποίηση προγραμμάτων λογισμικού για τη βελτίωση της απόδοσης.
• Σύνταξη κώδικα προγραμματισμού υπολογιστή.
• Συνεργασία με άλλους για την επίλυση θεμάτων πληροφορικής.

Είναι σύνηθες να ταξινομούνται οι προγραμματιστές υπολογιστών σε δύο κατηγορίες: προγραμματιστές συστημάτων και προγραμματιστές εφαρμογών.
Και οι δύο τύποι προγραμματιστών κωδικοποιούν. Η διαφορά μεταξύ των δύο είναι ότι οι προγραμματιστές εφαρμογών εκτελούν κωδικοποίηση για να διαχειριστούν μια συγκεκριμένη εργασία, όπως η κωδικοποίηση ενός προγράμματος για την παρακολούθηση του αποθέματος μέσα σε μια εταιρεία.
Από την άλλη πλευρά, οι προγραμματιστές συστημάτων κωδικοποιούν προγράμματα για τη συντήρηση και τον έλεγχο του λογισμικού συστήματος, συμπεριλαμβανομένων των συστημάτων διαχείρισης βάσεων δεδομένων και των λειτουργικών συστημάτων (OSs).

2.1.1 Γλώσσες προγραμματισμού

Ακριβώς όπως οι γλώσσες των ανθρώπων και οι υπολογιστές έχουν τις δικές τους γλώσσες. Πράγματι, υπάρχουν πολλές γλώσσες προγραμματισμού υπολογιστών που μπορούν να χρησιμοποιήσουν οι προγραμματιστές για να επικοινωνήσουν με έναν υπολογιστή για να εκτελέσουν ένα σύνολο συγκεκριμένων εργασιών.

Το τμήμα της γλώσσας που μπορεί να καταλάβει ένας υπολογιστής ονομάζεται «binary» («δυαδικό»). Η μετάφραση της γλώσσας προγραμματισμού σε δυαδικό είναι γνωστή ως «compiling». Κάθε γλώσσα προγραμματισμού έχει τα δικά της ξεχωριστά χαρακτηριστικά.

Ωστόσο, μερικές φορές υπάρχουν ομοιότητες μεταξύ τους. [3]

[2] Techopedia. (2014, July 30). Computer Programmer. Techopedia.Com. https://www.techopedia.com/definition/6589/computer-programmer
[3] Writers, S. (2021, September 27). Guide to Programming Languages | ComputerScience.org. Get an Education the World Needs | ComputerScience.Org. https://www.computerscience.org/resources/computer-programming-languages/

Μερικές από τις πιο κοινές γλώσσες προγραμματισμού υπολογιστών είναι:

- C
- SQL
- PHYTON
- JAVA
- R
- HTML[4]

[4] Gallagher, J. (2021, May 4). The Most Popular Programming Languages. Career Karma. https://careerkarma.com/blog/top-programming-languages-2021/

Πηγή:https://pixabay.com/illustrations/programming-languages-icon-898961/

Πολλές γλώσσες προγραμματισμού είναι σχετικά απλές, αλλά κάνουν διαφορετικά πράγματα. Για παράδειγμα, μια από τις πιο δημοφιλείς γλώσσες, η JavaScript, χρησιμοποιείται κυρίως για ιστοσελίδες και ανάπτυξη «front-end». Από την άλλη πλευρά, η Python χρησιμοποιείται τόσο για ολοκληρωμένα προγράμματα λογισμικού όσο και για ιστότοπους.

Όποια και αν επιλέξει ένας προγραμματιστής λογισμικού είναι συνήθως στη διακριτική του ευχέρεια, καθώς περισσότερες από μία γλώσσες προγραμματισμού μπορούν συχνά να λειτουργήσουν για τον ίδιο σκοπό.

Η γνώση μιας από αυτές τις γλώσσες προγραμματισμού είναι απαραίτητη εάν θέλει κανείς να κάνει καριέρα όχι μόνο ως προγραμματιστής υπολογιστών, αλλά και ως μηχανικός λογισμικού, ή για οποιαδήποτε δουλειά στον τομέα της επιστήμης δεδομένων που απαιτεί γνώσεις υψηλής τεχνολογίας.

2.1.2 Διαδικασία προγραμματισμού

Ο κύριος στόχος του προγραμματισμού είναι γενικά η δημιουργία μιας λύσης για την επίλυση ενός προβλήματος. Το πρόβλημα μπορεί να είναι κάτι μεγάλο σε διεθνή κλίμακα ή απλά κάτι μικρό για να μας απαλλάξει από την πλήξη.

Πήγη:https://pixabay.com/illustrations/programming-languages-icon-898961/

Η ανάπτυξη ενός προγράμματος περιλαμβάνει βήματα παρόμοια με οποιαδήποτε εργασία επίλυσης προβλημάτων. Υπάρχουν τέσσερα βασικά βήματα στη διαδικασία προγραμματισμού:

- **Καθορισμός του προβλήματος:** σε αυτό το στάδιο προσπαθείτε να εντοπίσετε το πρόβλημα. Ωστόσο, στην ουσία αυτό που πραγματικά προσπαθείτε να προσδιορίσετε είναι η λύση του προβλήματος, καθώς προσπαθείτε να ορίσετε αυτό που θέλετε να επιτύχετε.

  Το πρώτο πράγμα που πρέπει να κάνετε είναι να αναλύσετε τις απαιτήσεις. Πράγματι, πρέπει να μάθετε τι θα πρέπει να κάνει το πρόγραμμά σας. Το δεύτερο είναι να παρατηρήσετε τη λίστα των απαιτήσεων και να αποφασίσετε τι ακριβώς θα πρέπει να περιλαμβάνει η λύση που θα δώσετε για να τις εκπληρώσει. Μερικές φορές ένα μεμονωμένο πρόβλημα μπορεί να απαιτεί πολλαπλές λύσεις, επομένως είναι σημαντικό να καταλάβετε ποια είναι η καλύτερη λύση για κάθε πρόβλημα. Έτσι, προσπαθείτε να προσδιορίσετε, για άλλη μια φορά, ποια λύση πρέπει να υιοθετήσετε για το πρόβλημά σας.
- **Οργάνωση και σχεδιασμός της λύσης:** αυτό είναι γενικά το πιο δύσκολο έργο, επειδή απαιτεί από εσάς να καταλάβετε πώς θα μετατρέψετε τις προδιαγραφές και τις απαιτήσεις σε ένα λειτουργικό πρόγραμμα.

  Ο σχεδιασμός μιας λύσης σημαίνει την περιγραφή σε ένα υψηλότερο επίπεδο όλων των βημάτων που πρέπει να ακολουθήσει ο υπολογιστής για να λειτουργήσει σωστά ή να ολοκληρώσει μια συγκεκριμένη εργασία.

  Αυτή η εργασία είναι η πιο δύσκολη για διαφορετικούς λόγους: για παράδειγμα, ίσως χρειαστεί να μάθετε περισσότερα σχετικά με τις λειτουργίες του υπολογιστή σας και τις γλώσσες προγραμματισμού που έχετε επιλέξει για να δείτε ποια πράγματα μπορεί να είναι εύκολα ή δύσκολα. Επίσης, όπως αναφέρθηκε προηγουμένως, ένα πρόβλημα μπορεί να απαιτεί διαφορετικές λύσεις και μόλις εντοπιστεί μια λύση, είναι σημαντικό να αναλύσετε όλα τα δυνατά και αδύνατα σημεία της και να κάνετε τη σωστή επιλογή.

  Ωστόσο, μόλις ολοκληρωθεί το δύσκολο μέρος, θα πρέπει να έχει ολοκληρωθεί και το σχεδιαστικό μέρος. Αυτό σημαίνει ότι θα έχετε μια πολύ ξεκάθαρη ιδέα για το πώς ο υπολογιστής πρόκειται να ικανοποιήσει τις προδιαγραφές, να ανταποκριθεί στις απαιτήσεις σας και τελικά να λύσει το αρχικό σας πρόβλημα.
- **Κωδικοποίηση του προγράμματος:** Σε αυτήν την εργασία θα πρέπει να δώσετε οδηγίες στον υπολογιστή σας για το πώς θα εκτελέσει το σχέδιό σας. Αυτή η φάση περιλαμβάνει 3 στάδια:

- **«Coding» («Κωδικοποίηση»):** Κωδικοποίηση σημαίνει μετατροπή του σχεδίου σας σε ένα λειτουργικό πρόγραμμα, χρησιμοποιώντας μια γλώσσα προγραμματισμού. Αυτό το στάδιο ουσιαστικά περιλαμβάνει να καθίσετε και να αρχίσετε να πληκτρολογείτε κωδικούς στον υπολογιστή σας. Ακριβώς όπως ένα δοκίμιο, η κωδικοποίηση ενός προγράμματος απαιτεί να συμπεριλάβετε πράγματα όπως τίτλους, σελίδα περιεχομένων, εισαγωγή και παραπομπές. Μόλις τελειώσετε με την κωδικοποίησή σας, πρέπει να την υποβάλετε στον υπολογιστή για να δείτε πώς αντιδρά ο υπολογιστής και πώς το αξιοποιεί.
- **«Compiling» («Μετατροπή»)**: Η φάση της «μετατροπής» απαιτεί να μετατρέψετε το γραπτό πρόγραμμα σε μια γλώσσα που κατανοεί ο υπολογιστής, εξ ου και ο δυαδικός κώδικας μηχανής. Υπάρχουν μερικά προγράμματα που ονομάζονται «compilers» που μπορούν να το κάνουν αυτό για εσάς. Ωστόσο, πρέπει να είστε πολύ ακριβείς όταν γράφετε το πρόγραμμά σας σε ένα «compiler», γιατί θα εντοπίσει και το παραμικρό λάθος.
- **«Debugging» («Διόρθωση σφαλμάτων»)**: Όπως αναφέρθηκε προηγουμένως, ο μετατροπέας («compiler») θα εντοπίσει κάθε λάθος που υπάρχει στο πρόγραμμα. Μην ανησυχείτε, αυτό είναι πολύ σύνηθες. Γι' αυτό υπάρχει αυτό το στάδιο. Μόλις εντοπίσετε όλα τα σφάλματα στο πρόγραμμά σας, χάρη στη βοήθεια των «compilers», κοιτάξτε ξανά το αρχικό πρόγραμμα, εντοπίστε τα λάθη, διορθώστε τον κώδικα και μετατρέψτε τον ξανά. Αυτός ο κύκλος κώδικα -> μετατροπή -> εντοπισμός σφαλμάτων θα επαναλαμβάνεται συχνά πολλές φορές πριν ο μετατροπέας είναι ικανοποιημένος και τελικά έχετε ένα πρόγραμμα που λειτουργεί. Αυτό μπορεί να διαρκέσει λίγη ώρα. Μπορείτε επίσης να αποφασίσετε να γράψετε μικρές ενότητες του προγράμματός σας, να τις μετατρέψετε και να τις διορθώσετε, αντί να εστιάζετε σε ολόκληρο το πρόγραμμα ταυτόχρονα.

- **Έλεγχος του προγράμματος:** Το τελευταίο μέρος της διαδικασίας προγραμματισμού περιλαμβάνει τη δοκιμή της δημιουργίας σας και τον έλεγχο ότι λειτουργεί ακριβώς όπως θέλετε. Μπορεί να πιστεύετε ότι οι «compilers» σας έχουν ήδη δώσει το σωστό πρόγραμμα, ωστόσο αυτό δεν σημαίνει ότι ένας σωστός κώδικας μπορεί να λύσει το αρχικό σας πρόβλημα. Αυτό θα πρέπει να το ελέγξετε. Εάν εντοπιστούν τυχόν λάθη, η μόνη λύση είναι να δείτε ξανά το πρόγραμμα, να διορθώσετε τα λάθη αλλάζοντας τον κώδικα και να τον μετατρέψετε ξανά. Θυμηθείτε να είστε προσεκτικοί όταν αλλάζετε κάτι στον κώδικα, γιατί η αλλαγή ενός μικρού πράγματος μπορεί να επηρεάσει και άλλα σε ολόκληρο το πρόγραμμα. Αυτή η φάση μπορεί να χαρακτηριστεί ως μια άλλη φάση εντοπισμού σφαλμάτων. Τέλος, αφού κάνετε ξανά μετατροπή, εντοπίσετε τα σφάλματα, διορθώσετε τα λάθη,

κάνετε δοκιμές και αφού βεβαιωθείτε ότι το πρόγραμμά σας λειτουργεί σύμφωνα με τις απαιτήσεις και τις προδιαγραφές σας, θα έχετε μια λύση για το πρόβλημά σας![5]

2.1.3 Αρχές προγραμματισμού

Σε αυτό το σημείο γνωρίζουμε ότι ένα από τα κύρια χαρακτηριστικά του προγραμματισμού ηλεκτρονικών υπολογιστών είναι η εγγραφή κωδίκων. Γενικά, είτε εργάζεστε μόνοι σας, είτε μέσα σε μια ομάδα, να θυμάστε ότι οι κώδικές σας πρέπει να είναι εύκολο να διαβαστούν και να διατηρηθούν από άλλα άτομα.
Γι' αυτό υπάρχουν μερικές βασικές αρχές προγραμματισμού που πρέπει να γνωρίζει κάθε προγραμματιστής για να γράφει εύκολους κώδικες, με καθαρές μεταβλητές, που να μπορούν να σταθούν ακόμα και μετά από δοκιμές και τυχόν τροποποιήσεις.

Κάποιες από αυτές τις αρχές είναι:

- **«KISS: Keep It Simple, Stupid»:** Η αρχή «KISS» είναι μία από τις πιο σημαντικές αρχές που πρέπει να τηρείτε στον κόσμο του προγραμματισμού. Σημαίνει ότι πρέπει να γράφετε τον κώδικα όσο πιο απλά γίνεται. Δεν χρειάζεται να προσπαθήσετε να επιδεικνύετε τις ικανότητές με τη χρήση προηγμένου κώδικα. Εάν μπορείτε να το γράψετε σε μία γραμμή, τότε χρησιμοποιήστε μία γραμμή. Και πρέπει να θυμάστε ότι ορισμένοι κώδικες πρέπει να επανεξεταστούν μετά από κάποιους μήνες, οπότε να το έχετε υπ' όψιν σας και να τον διατηρείται απλό.

- **«DRY: Don't Repeat Yourself»:** Αυτή η αρχή σας βοηθά να θυμάστε ένα πολύ συνηθισμένο λάθος στην κωδικοποίηση, δηλαδή τις επαναλήψεις. Πρέπει να αποφεύγετε κάθε αντιγραφή δεδομένων και συνόλου αρχών.
- **«Open/Closed»:** Αυτή η αρχή φροντίζει να θυμάστε να κρατάτε τους κωδικούς σας ανοιχτούς για επεκτάσεις, αλλά κλειστούς για τροποποιήσεις. Αυτός ο κανόνας είναι χρήσιμος όταν παρέχετε ένα σύνολο υπηρεσιών που είναι ανοιχτές για χρήση από άλλους. Με απλά λόγια, εάν κάποιος άλλος τροποποιήσει τον κώδικά σας, ο κώδικας θα σπάσει ή κάτι στον κώδικα θα επηρεαστεί και δεν θα

[5] The Programming Process. (2020). Cs.Bham.Ac.Uk. https://www.cs.bham.ac.uk/%7Erxb/java/intro/2programming.html

λειτουργεί όπως θα έπρεπε. Γι' αυτό θα πρέπει να μοιράζεστε μόνο κώδικες που εμποδίζουν την άμεση τροποποίηση και ενθαρρύνουν την επέκταση.

- **«Single responsibility»:** Αυτή η αρχή δηλώνει ότι κάθε επίπεδο ή ενότητα σε ένα πρόγραμμα πρέπει να παρέχει μόνο μία συγκεκριμένη λειτουργία. Αυτός είναι ο λόγος για τον οποίο με αυτόν τον τρόπο ο κώδικας διατηρείται απλός και επίσης αποφεύγεται μια πιο περίπλοκη διαδικασία εντοπισμού σφαλμάτων. Δεύτερον, γίνεται πιο δύσκολο να δημιουργηθεί πρόσθετη λειτουργία για μια συγκεκριμένη ενότητα.
- **«YAGNI: You Aren't Going to Need It»:** Αυτή η αρχή τονίζει ότι δεν πρέπει ποτέ να επιχειρήσετε να γράψετε κώδικες για μια λειτουργία που μπορεί να χρειαστείτε στο μέλλον. Αυτό θα σήμαινε ότι προσπαθείτε να λύσετε ένα πρόβλημα που δεν υπάρχει.

2.1.4 Πώς μπορείτε να γίνετε προγραμματιστής υπολογιστών;

Το να είσαι προγραμματιστής υπολογιστών μπορεί να σε βοηθήσει να βρεις δουλειά στον κλάδο του σχεδιασμού συστημάτων υπολογιστών και των σχετικών υπηρεσιών. Ωστόσο, πολλά πεδία της επιστήμης των υπολογιστών απαιτούν έναν προγραμματιστή υπολογιστών.

Αυτά περιλαμβάνουν για παράδειγμα:

- Μηχανική λογισμικού (Software engineering)
- Ανάπτυξη λογισμικού (Software development)
- Μηχανική Η/Υ (Computer engineering)
- Γραφικά Η/Υ (Computer graphics)
- Τεχνητή νοημοσύνη (Artificial Intelligence)

Γενικά, για τους προγραμματιστές υπολογιστών απαιτείται να έχουν πανεπιστημιακό πτυχίο στα δεδομένα (data) και την επιστήμη των υπολογιστών, που μπορεί να δώσει τα θεμέλια για την αρχή της καριέρας σας.

Ωστόσο, πολλοί προγραμματιστές μπορεί επίσης να είναι αυτοδίδακτοι ενθουσιώδεις, γεμάτοι με έντονο ενδιαφέρον για τον προγραμματισμό.

Είτε αποφασίσετε να ξεκινήσετε την καριέρα σας με πτυχίο πανεπιστημίου, είτε από μόνοι σας, το σημαντικό είναι να βελτιώσετε τις γνώσεις σας στις γλώσσες προγραμματισμού, να ενδιαφέρεστε για τον προγραμματισμό ηλεκτρονικών υπολογιστών και επίσης να ενημερώνεστε και να μην σταματήσετε ποτέ να μαθαίνετε, γιατί ο προγραμματισμός υπολογιστών είναι ένας τομέας που ενημερώνεται συνεχώς.

Πηγή: https://pixabay.com/photos/scrabble-board-game-game-4370255/

Ένα άλλο σημαντικό πράγμα που μπορεί να απαιτούν πολλές θέσεις εργασίας προγραμματισμού ηλεκτρονικών υπολογιστών είναι οι διαθέσιμες επαγγελματικές και μη κερδοσκοπικού χαρακτήρα πιστοποιήσεις.

Μερικές από αυτές περιλαμβάνουν:

- «CISCO» – Πιστοποιημένος Συνεργάτης Δικτύου, Πιστοποιημένο Επάγγελμα Δρομολόγησης και Εναλλαγής Δικτύου, Πιστοποιημένο Διαπιστευτήριο Ασφάλειας Συνεργάτη Δικτύου
- «Microsoft» – Πιστοποιημένος Προγραμματιστής Λύσεων για εφαρμογές Ιστού, Πιστοποιημένος Συνεργάτης Λύσεων του Διακομιστή των Windows
- Επαγγελματικοί Σύλλογοι – Πιστοποίηση Συνεργάτη Ανάπτυξης Λογισμικού, Πιστοποίηση Comptia's Security+, Πιστοποίηση Comptia's A+, Πιστοποίηση Comptia's Linux+
- Μη κερδοσκοπικοί οργανισμοί – Πιστοποιημένος Επαγγελματίας Ασφάλειας Πληροφοριακών Συστημάτων, Πιστοποιημένος Διευθυντής Ασφάλειας Πληροφοριών, Πιστοποιημένο Επαγγελματικό Διαπιστευτήριο Ασφαλούς Κύκλου ζωής Λογισμικού 6

3. Αξιολόγηση

3.1 Αξιολόγηση γνώσεων

Αξιολόγηση τύπου κουίζ που στηρίζεται στο βασικό περιεχόμενο. Σημειώστε τη σωστή απάντηση με έντονους χαρακτήρες όταν απαιτείται. Συμπεριλάβετε 10 ερωτήσεις για την ενότητα σας. Αυξήστε σταδιακά το επίπεδο δυσκολίας.

Ερώτηση 1: Ο προγραμματιστής υπολογιστών είναι η επαγγελματική φιγούρα που είναι υπεύθυνη για τη σύνταξη κωδίκων ή της κωδικοποίησης, που καθοδηγούν πώς πρέπει να λειτουργεί ένας υπολογιστής, μια εφαρμογή ή ένα πρόγραμμα λογισμικού.

[Σωστό] [Λάθος]

Ερώτηση 2: Η διαδικασία κωδικοποίησης μπορεί να κάνει μόνο έναν υπολογιστή να λειτουργήσει.
[Σωστό] **[Λάθος]**

Ερώτηση 3: Ποιες εργασίες πρέπει να γνωρίζει καλά ένας προγραμματιστής υπολογιστών;

[Επίλυση προβλημάτων λογισμικού υπολογιστή] [Τροποποίηση προγραμμάτων λογισμικού για τη βελτίωση της απόδοσης] [Γράψιμο κώδικα προγραμματισμού υπολογιστή] [Έλεγχος απόδοσης λογισμικού] **[Όλα τα παραπάνω]**

[6] Cote, J. (2021, August 19). What is Computer Programming? How to Become a Computer Programmer. SNHU.Edu. https://www.snhu.edu/about-us/newsroom/stem/what-is-computer-programming

Ερώτηση 4: Το τμήμα της γλώσσας που μπορεί να καταλάβει ένας υπολογιστής ονομάζεται «δυαδικό». Πώς ονομάζεται η διαδικασία μετατροπής της γλώσσας προγραμματισμού σε δυαδικό;
[Κωδικοποίηση - Coding] **[Μεταροπή - Compiling]** [Διόρθωση Σφαλμάτων - Debugging]

Ερώτηση 5: Ποια είναι η σωστή σειρά των εργασιών της διαδικασίας προγραμματισμού;

[Κωδικοποίηση του προγράμματος, Έλεγχος του προγράμματος, Καθορισμός του προβλήματος, Οργάνωση και σχεδιασμός της λύσης]
[Οργάνωση και σχεδιασμός της λύσης, Καθορισμός του προβλήματος, Κωδικοποίηση του προγράμματος, Έλεγχος του προγράμματος]
[Καθορισμός του προβλήματος, Οργάνωση και σχεδιασμός της λύσης, Κωδικοποίηση του προγράμματος, Έλεγχος του προγράμματος]

Ερώτηση 6: Ποια είναι τα τρία στάδια της φάσης Κωδικοποίησης στη διαδικασία Προγραμματισμού;
[Έλεγχος] **[Κωδικοποίηση]** [Συστημοποίηση] **[Μετατροπή] [Διόρθωση Σφαλμάτων]**

Ερώτηση 7: Τι υπενθυμίζουν οι αρχές προγραμματισμού στους προγραμματιστές να κάνουν κατά την κωδικοποίηση;
[να χρησιμοποιούν κώδικες που άλλοι προγραμματιστές δεν θα μπορούν να τροποποιήσουν στο μέλλον] [να χρησιμοποιούν σύνθετους κώδικες που βελτιώνουν τις λειτουργίες] **[να γράφουν εύκολους κώδικες] [να χρησιμοποιούν ξεκάθαρες μεταβλητές που θα είναι ισχυρές μετά τη δοκιμή]**

Ερώτηση 8 (αντιστοίχιση): Αντιστοιχίστε τους όρους με τους ορισμούς τους.

Δυαδικό: Ένα σύστημα αριθμών βάσης-2 που εφευρέθηκε από τον Gottfried Leibniz που αποτελείται μόνο από δύο αριθμούς ή ψηφία: 0 (μηδέν) και 1 (ένα).

ΤΠΕ: Αυτή η λέξη αναφέρεται σε τεχνολογίες που παρέχουν πρόσβαση σε πληροφορίες μέσω των τηλεπικοινωνιών.

Μηχανική μάθηση: ένας κλάδος της τεχνητής νοημοσύνης (AI) και της επιστήμης των υπολογιστών που επικεντρώνεται στη χρήση δεδομένων και αλγορίθμων για μίμηση του τρόπου με τον οποίο μαθαίνουν οι άνθρωποι, βελτιώνοντας σταδιακά την ακρίβειά του

Αναλογικό: Επίθετο που αναφέρεται σε μηχανισμό ή συσκευή στην οποία οι πληροφορίες αντιπροσωπεύονται από συνεχώς μεταβλητά φυσικά μεγέθη. Είναι το αντίθετο του ψηφιακού.

Ερώτηση 9 (αντιστοίχιση): Αντιστοιχίστε τις έννοιες με τις εξηγήσεις τους.

Προγραμματιστής υπολογιστών: άτομο που είναι υπεύθυνο για τη δημιουργία οδηγιών για εκτέλεση από έναν υπολογιστή γράφοντας και δοκιμάζοντας κώδικα που επιτρέπει σε εφαρμογές και προγράμματα λογισμικού να λειτουργούν με επιτυχία.

«Compiling»: διαδικασία μετατροπής της γλώσσας προγραμματισμού σε δυαδικό.

«Debugging»: Διαδικασία που περιλαμβάνει τον εντοπισμό όλων των σφαλμάτων στο πρόγραμμα, χάρη στη βοήθεια των «compilers», την αναθεώρηση του αρχικού προγράμματος, τον εντοπισμό των λαθών, τη διόρθωση του κώδικα και την εκ νέου μετατροπή.

«**Coding»:** διαδικασία μετάφρασης του σχεδίου σε ένα λειτουργικό πρόγραμμα, χρησιμοποιώντας μια γλώσσα προγραμματισμού.

«**Testing»**: διαδικασία που περιλαμβάνει την αναθεώρηση του προγράμματος για να ελέγξετε ότι λειτουργεί ακριβώς όπως θέλετε.

3.2 Αξιολόγηση δεξιοτήτων

Η αναλυτική σκέψη είναι ένα σημαντικό χαρακτηριστικό σε έναν προγραμματιστή. Αναφέρεται επίσης ως αναλυτικός συλλογισμός, αφηρημένη σκέψη ή κριτική σκέψη. Οι άνθρωποι που μπορούν να σκέφτονται λογικά είναι σε θέση να αναλύουν προβλήματα και να επινοούν λύσεις. Αυτό δεν είναι μόνο πολύτιμο κατά την ανάπτυξη προγραμμάτων, αλλά είναι ζωτικής σημασίας για κάθε κατάσταση που απαιτεί ορθολογική σκέψη. Άνθρωποι που είναι λογικοί:

- αναλύουν πληροφορίες ή πηγές που σχετίζονται με μια εργασία
- παρατηρούν προσεκτικά τι συμβαίνει
- μελετούν αντικειμενικά τις πληροφορίες για να προσδιορίσουν αν είναι σχετικές ή αληθείς
- εστιάζουν σε γεγονότα και όχι σε συναισθήματα
- αναπτύσσουν λύσεις σε προβλήματα βασισμένοι σε γεγονότα
- περιγράφουν τις ιδέες με σαφήνεια, χωρίζοντάς τις σε μέρη
- δίνουν προσοχή στις λεπτομέρειες
- δοκιμάζουν την αποτελεσματικότητα μιας λύσης και κάνουν αναθεωρήσεις

Υπάρχουν πολλές ασκήσεις και δραστηριότητες για την εξάσκηση αυτής της ικανότητας. Μερικές από αυτές είναι πολύ απλές, όπως το να διαβάζετε βιβλία, να παίζετε παιχνίδια σκέψης ή να συμμετέχετε σε ένα λέσχες «debate» για να διατηρείτε τον εγκέφαλό σας τονωμένο.
Στην περίπτωση ενός προγραμματιστή, ένας εύκολος τρόπος για να βελτιώσετε τις δεξιότητές σας στην αναλυτική σκέψη είναι να ακολουθήσετε και να έχετε υπόψη σας αυτόν τον οδηγό προσέγγισης 5 βημάτων:

1. Ανάλυση του προβλήματος: περιλαμβάνει τη συλλογή πληροφοριών, την εξέταση πηγών και τον προσδιορισμό κενών σε δεξιότητες ή γνώσεις.
2. Διατύπωση ενός σχεδίου: συλλέξτε ιδέες και οργανώστε τις σκέψεις σας για ένα σχέδιο.
3. Ανάπτυξη κώδικα για να λύσετε το πρόβλημα: ξεκινήστε να γράφετε τον κώδικα (η γνώση γλωσσών υπολογιστή είναι απαραίτητη).
4. Αξιολόγηση της λύσης και αναθεώρηση του κώδικα: Επανεξετάστε τη λύση και εντοπίστε τομείς προς βελτίωση, εάν χρειάζεται.
5. Αιτιολόγηση των αποφάσεων: να παρουσιάσετε στοιχεία που εξηγούν γιατί το πρόγραμμα είναι μια αποδεκτή λύση.

Περισσότερες πληροφορίες εδώ: https://www.technokids.com/blog/teaching-strategies/its-easy-to-improve-logical-thinking-with-programming/

4. Παραπομπές

Cote, J. (2021, August 19). What is Computer Programming? How to Become a Computer Programmer. SNHU.Edu. https://www.snhu.edu/about-us/newsroom/stem/what-is-computer-programming

Weinstein, J. (2021, January 24). What Is Coding? Coding Definition and Uses. Career Karma. https://careerkarma.com/blog/what-is-coding-used-for/

Writers, S. (2021, September 27). Guide to Programming Languages | ComputerScience.org. Get an Education the World Needs | ComputerScience.Org. https://www.computerscience.org/resources/computer-programming-languages/

Gallagher, J. (2021, May 4). The Most Popular Programming Languages. Career Karma. https://careerkarma.com/blog/top-programming-languages-2021/

Techopedia. (2014, July 30). Computer Programmer. Techopedia.Com.

https://www.techopedia.com/definition/6589/computer-programmer

The Programming Process. (2020). Cs.Bham.Ac.Uk.

https://www.cs.bham.ac.uk/%7Erxb/java/intro/2programming.html