***Πληροφορική***

***Θεωρία στις Δομές Δεδομένων - (3ο κεφ. (3.1, 3.2, 3.3) - 9ο κεφ. (9.1,9.2))***

Όνομα μαθητή:

***Συμπληρώνουμε τα κενά θεωρίας που αφορούν τα Δεδομένα και τις Δομές δεδομένων.***

**Δεδομένα**

Τα ......................... (data) είναι η αφαιρετική αναπαράσταση της πραγματικότητας και συνεπώς μία απλοποιημένη όψη της. Η συλλογή των ακατέργαστων δεδομένων και ο συσχετισμός τους δίνει ως αποτέλεσμα την ....................... (information). Ο ......................... είναι το μέσο για την παραγωγή πληροφορίας από τα δεδομένα.

Η μέτρηση, η κωδικοποίηση, η μετάδοση της πληροφορίας αποτελεί αντικείμενο μελέτης ενός ιδιαίτερου κλάδου, της Θεωρίας ............................ (Information Theory), που είναι ένα ιδιαίτερα σημαντικό πεδίο της Πληροφορικής.

Η πληροφορική μελετά τα δεδομένα από τις σκοπιές:

Υλικού. Το υλικό (hardware), δηλαδή η μηχανή, επιτρέπει στα δεδομένα ενός προγράμματος να ........................ στην κύρια μνήμη και στις περιφερειακές συσκευές του υπολογιστή με διάφορες αναπαραστάσεις (representations).

Γλωσσών προγραμματισμού. Οι γλώσσες προγραμματισμού υψηλού επιπέδου (high level programming languages) επιτρέπουν τη χρήση διάφορων .............. (types) μεταβλητών (variables) για να περιγράψουν ένα δεδομένο.

Δομών δεδομένων. Δομή δεδομένων (data structure) είναι ένα σύνολο δεδομένων μαζί με ένα σύνολο επιτρεπτών ................... επί αυτών.

Ανάλυσης δεδομένων. Τρόποι καταγραφής και αλληλοσυσχέτισης των δεδομένων μελετώνται έτσι ώστε να αναπαρασταθεί η γνώση για πραγματικά γεγονότα. Περιλαμβάνει τις τεχνολογίες των Βάσεων ................... (Databases), της Μοντελοποίησης Δεδομένων (Data Modelling) και της Αναπαράστασης ................ (Knowledge Representation) .

**Δομές Δεδομένων**

Όπως ειπώθηκε «Δομή Δεδομένων» είναι ένα σύνολο αποθηκευμένων δεδομένων που υφίστανται επεξεργασία από ένα σύνολο λειτουργιών. Κάθε μορφή δομής δεδομένων αποτελείται από ένα σύνολο .................... (nodes).

Οι βασικές λειτουργίες (ή αλλιώς πράξεις) επί των δομών δεδομένων είναι οι ακόλουθες:

* ....................... (access), πρόσβαση σε έναν κόμβο με σκοπό να εξετασθεί ή να τροποποιηθεί το περιεχόμενό του.
* ................... (insertion), δηλαδή η προσθήκη νέων κόμβων σε μία υπάρχουσα δομή.
* ................. (deletion), που αποτελεί το αντίστροφο της εισαγωγής, δηλαδή ένας κόμβος αφαιρείται από μία δομή.
* Αναζήτηση (searching), κατά την οποία προσπελαύνονται οι κόμβοι μιας δομής, προκειμένου να εντοπιστούν ένας ή περισσότεροι που έχουν μια δεδομένη ........................
* Ταξινόμηση (sorting), όπου οι κόμβοι μιας δομής διατάσσονται κατά αύξουσα ή ...................... σειρά.
* .................. (copying), κατά την οποία όλοι οι κόμβοι ή μερικοί από τους κόμβους μίας δομής αντιγράφονται σε μία άλλη δομή.
* Συγχώνευση (merging), κατά την οποία δύο ή ........................... δομές συνενώνονται σε μία ενιαία δομή.
* Διαχωρισμός (separation), που αποτελεί την ................................ πράξη της συγχώνευσης.

Στην πράξη σπάνια χρησιμοποιούνται και οι οκτώ λειτουργίες για κάποια δομή. Συνηθέστατα παρατηρείται το φαινόμενο μία δομή δεδομένων να είναι αποδοτικότερη από μία άλλη δομή με κριτήριο κάποια λειτουργία, για παράδειγμα την αναζήτηση, αλλά λιγότερο αποδοτική για κάποια άλλη λειτουργία, για παράδειγμα την εισαγωγή. Αυτές οι παρατηρήσεις εξηγούν αφ' ενός την ύπαρξη διαφορετικών δομών, και αφ' ετέρου τη σπουδαιότητα της επιλογής της κατάλληλης δομής κάθε φορά

Η παρατήρηση ότι το πρόγραμμα πρέπει να θεωρεί τη δομή δεδομένων και τον αλγόριθμο ως μία αδιάσπαστη ενότητα δικαιολογεί την εξίσωση που διατυπώθηκε το 1976 από τον Wirth (που σχεδίασε και υλοποίησε τη γλώσσα Pascal): ............................. + ........................................ = ..........................................

Οι δομές δεδομένων διακρίνονται σε δύο μεγάλες κατηγορίες: τις ................. (static) και τις ..................... (dynamic). Οι δυναμικές δομές δεν αποθηκεύονται σε συνεχόμενες θέσεις μνήμης αλλά στηρίζονται στην τεχνική της λεγόμενης δυναμικής παραχώρησης ............... (dynamic memory allocation). Με άλλα λόγια, οι δομές αυτές δεν έχουν σταθερό μέγεθος, αλλά ο αριθμός των κόμβων τους μεγαλώνει και μικραίνει καθώς στη δομή εισάγονται νέα δεδομένα ή διαγράφονται κάποια δεδομένα αντίστοιχα. Όλες οι σύγχρονες γλώσσες προγραμματισμού προσφέρουν τη δυνατότητα δυναμικής παραχώρησης μνήμης. Ωστόσο, εμείς στη συνέχεια θα εξετάσουμε μόνο τις στατικές δομές που είναι ευκολότερες στην κατανόηση και την υλοποίησή τους.

Με τον όρο στατική δομή δεδομένων εννοείται ότι το ακριβές μέγεθος της απαιτούμενης κύριας μνήμης καθορίζεται κατά τη στιγμή του ...................... τους, και κατά συνέπεια κατά τη στιγμή της μετάφρασής τους και όχι κατά τη στιγμή της εκτέλεσης τους προγράμματος. Μία άλλη σημαντική διαφορά σε σχέση με τις δυναμικές δομές είναι ότι τα στοιχεία των στατικών δομών αποθηκεύονται σε ...................... θέσεις μνήμης. Στην πράξη, οι στατικές δομές υλοποιούνται με πίνακες που μας είναι γνωστοί από άλλα μαθήματα και υποστηρίζονται από κάθε γλώσσα προγραμματισμού.

**Πίνακες**

Μπορούμε να ορίσουμε τον πίνακα ως μια δομή που περιέχει στοιχεία του .......... τύπου (δηλαδή ακέραιους, πραγματικούς κ.λπ). Η αναφορά στα στοιχεία ενός πίνακα γίνεται με τη χρήση του συμβολικού ονόματος του πίνακα ακολουθούμενου από την τιμή ενός ή περισσότερων .............. (indexes) σε παρένθεση ή αγκύλη. Ένας πίνακας μπορεί να είναι μονοδιάστατος, αλλά στη γενικότερη περίπτωση μπορεί να είναι δισδιάστατος, τρισδιάστατος και γενικά ν-διάστατος πίνακας. Όσον αφορά στους δισδιάστατους πίνακες σημειώνεται ότι, αν το μέγεθος των δύο διαστάσεων είναι ίσο, τότε ο πίνακας λέγεται .......................... (square) και γενικά συμβολίζεται ως πίνακας n x n. Μάλιστα μπορούμε να θεωρήσουμε το δισδιάστατο πίνακα ότι είναι ένας μονοδιάστατος πίνακας, όπου κάθε θέση του περιέχει ένα νέο ............................. πίνακα

Πέρα από τα πλεονεκτήματα των πινάκων στην αποθήκευση και επεξεργασία πολλών δεδομένων , υπάρχουν και δύο **μειονεκτήματα**.

Οι πίνακες απαιτούν ............ Κάθε πίνακας δεσμεύει από την αρχή του προγράμματος πολλές θέσεις μνήμης. Σε ένα μεγάλο και σύνθετο πρόγραμμα η άσκοπη χρήση μεγάλων πινάκων μπορεί να οδηγήσει ακόμη και σε αδυναμία εκτέλεσης του προγράμματος.

Οι πίνακες περιορίζουν τις δυνατότητες του προγράμματος. Αυτό γιατί οι πίνακες είναι ............ δομές και το μέγεθός τους πρέπει να δηλώνεται στην αρχή του προγράμματος, ενώ παραμένει υποχρεωτικά σταθερό κατά την εκτέλεση του προγράμματος.

Η απόφαση για τη χρήση ή όχι πίνακα για τη διαχείριση των δεδομένων είναι κυρίως θέμα εμπειρίας στον προγραμματισμό.

Γενικά, αν τα δεδομένα που εισάγονται σε ένα πρόγραμμα πρέπει να διατηρούνται στη μνήμη μέχρι το ............... της εκτέλεσης, τότε η χρήση πινάκων βοηθάει ή συχνά είναι απαραίτητη για την επίλυση του προβλήματος. Σε άλλη περίπτωση μπορεί να αποφεύγεται η χρήση τους.

***Ασκήσεις***

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Έχουμε πίνακα Α με μέγεθος 5, διαβάζουμε δεδομένα από το πληκτρολόγιο και τα αποθηκεύουμε σε αυτόν. Ένας μαθητής έγραψε τις εξής εντολές:   |  |  | | --- | --- | | **Διάβασε** Α[1]  **Διάβασε** Α[2]  **Διάβασε** Α[3]  **Διάβασε** Α[4]  **Διάβασε** Α[5]  Αλλά κατάλαβε ότι αυτό μπορεί να γίνει μονό για μικρούς πίνακες. Έτσι αποφάσισε να χρησιμοποιήσει μία εντολή επανάληψης ΓΙΑ. Βοηθείστε τον συμπληρώνοντας τις εντολές: | **Για** ι **από** ....................................  **Διάβασε** Α[....]  **Τέλος\_επανάληψης**  Για να διαπιστώσει τι δεδομένα μπήκανε στον πίνακα θέλει να εμφανίζει το περιεχόμενό του. Γράψτε παρακάτω τις εντολές: | | Αν τα δεδομένα του πίνακα ήτανε αριθμοί ας σκεφτούμε πως θα τα προσθέσουμε. Θέλουμε μία μεταβλητή αθροιστή που θα συγκεντρώνει το άθροισμα. Ας την ονομάσουμε Σ και αρχικά την μηδενίζουμε. Μετά με μία εντολή **Για** θα προσπελάσουμε τα στοιχεία του πίνακα ένα ένα και θα τα προσθέτουμε.  Σ **←**.............  **Για** ι **από** 1 **μέχρι 5**  Σ **←** ............ + Α[ι]  **Τέλος\_επανάληψης**  **Εμφάνισε** Σ  Αν ο πίνακας Α έχει τα παρακάτω στοιχεία ας συμπληρώσουμε τον πίνακα τιμών   |  |  |  |  |  | | --- | --- | --- | --- | --- | | 6 | 9 | 2 | 21 | 4 |  |  |  |  | | --- | --- | --- | | Σ | ι | Οθόνη | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |
|  |  |
| Ας προσπαθήσουμε μόνοι μας να διπλασιάσουμε τα στοιχεία του πίνακα :  **Για** ι **από** 1 **μέχρι 5**  .........................................................................  **Τέλος\_επανάληψης** | Ας προσπαθήσουμε μόνοι μας να αντικαταστήσουμε τα στοιχεία του πίνακα που είναι μεγαλύτερα του 5 με το 0. |
| Ας προσπαθήσουμε μόνοι μας να βρούμε πόσα στοιχεία του πίνακα είναι ανάμεσα στο 2 και το 5: | Ας προσπαθήσουμε μόνοι μας να βρούμε το μεγαλύτερο στοιχείο του πίνακα: |

***Σειριακή αναζήτηση(Κεφ 3ο (3.6))***

Όνομα μαθητή:

**Δραστηριότητα 1. (θεωρία)**

Αναφέρετε τρεις περιπτώσεις που χρησιμοποιείται η σειριακή αναζήτηση;

**Δραστηριότητα 2.**

|  |  |
| --- | --- |
| Σειριακή αναζήτηση που βρίσκει αν υπάρχει ένα κλειδί μέσα στον πίνακα.  **Αλγόριθμος** Σειριακή\_αναζήτηση1  **Διάβασε** Ν ! Ν το μέγεθος του πίνακα  **Για** ι **από** 1 **μέχρι** Ν  Δ**ιάβασε** Α[ι]  **Τέλος\_επανάληψης**  θέση**←**0  **Διάβασε** κλειδί  **Για** ι **από** 1 **μέχρι** Ν  **Αν** Α[ι]=κλειδί **τότε**  θέση←ι  **Τέλος\_Αν**  **Τέλος\_επανάληψης**  **Αν** θέση=0 **τότε**  **Εμφάνισε** "Δεν βρέθηκε"  **Αλλιώς**  **Εμφάνισε** θέση  **Τέλος\_αν**  **Τέλος** Σειριακή\_αναζήτηση1 | Αν το κλειδί υπάρχει 3 φορές στον πίνακα. Ποια θέση από αυτές θα εμφανίσει ο διπλανός αλγόριθμος;  ......................................................................................  Αν θέλουμε να εμφανίζει ο αλγόριθμος το κλειδί κάθε φορά που βρίσκει πως θα τροποποιηθεί;  **Αλγόριθμος** Σειριακή\_αναζήτηση2  **Διάβασε** Ν ! Ν το μέγεθος του πίνακα  **Για** ι **από** 1 **μέχρι** Ν  Δ**ιάβασε** Α[ι]  **Τέλος\_επανάληψης**  θέση**←**0  **Διάβασε** κλειδί  **Για** ι **από** 1 **μέχρι** Ν  **Αν** Α[ι]=κλειδί **τότε**  ......................................  **Τέλος\_Αν**  **Τέλος\_επανάληψης**  **Αν** θέση=0 **τότε**  **Εμφάνισε** ...................................................  **Τέλος\_αν**  **Τέλος** Σειριακή\_αναζήτηση2 |

**Δραστηριότητα 3. SOS:** αν ξέρουμε ότι το κλειδί υπάρχει μόνο μία φορά, η αλλιώς ότι κάθε στοιχείο του πίνακα είναι μοναδικό, πως θα τροποποιήσουμε τον αλγόριθμο ώστε μόλις το βρει να σταματά. Αυτή η βελτίωση είναι απαραίτητη γιατί φανταστείτε να βρούμε το στοιχείο στις πρώτες θέσεις του πίνακα , τότε ποιο το νόημα να εξετάσουμε και όλες τις υπόλοιπες;

Άρα η αναζήτηση συνεχίζεται όσο το στοιχείο δεν έχει βρεθεί και ένα τρόπος να το καταλάβουμε είναι όσο **θέση <>0.** Επίσης, συνεχίζεται όσο δεν φτάσαμε στο τέλος του πίνακα, δηλαδή όσο **ι<=Ν**. Συμπληρώστε παρακάτω τον αλγόριθμο:

|  |  |
| --- | --- |
| **Αλγόριθμος** Σειριακή\_αναζήτηση3  **Διάβασε** Ν  **Για** ι **από** 1 **μέχρι** Ν  Δ**ιάβασε** Α[ι]  **Τέλος\_επανάληψης**  ι**←**1  θέση**←**0  **Διάβασε** κλειδί  **Όσο** ................. **ΚΑΙ** ................... **επανάλαβε**  **Αν** Α[ι]=κλειδί **τότε**  θέση←ι  **Αλλιώς**  ι←........................  **Τέλος\_Αν**  **Τέλος\_επανάληψης**  **Αν** θέση=0 **τότε**  **Εμφάνισε** "Δεν βρέθηκε"  **Αλλιώς**  **Εμφάνισε** θέση  **Τέλος\_αν**  **Τέλος** Σειριακή\_αναζήτηση3 | **SOS με χρήση λογικής μεταβλητής = μεταβλητή σημαία**  Σύμφωνα με αυτή την προγραμματιστική τεχνική χρησιμοποιούμε μια λογική μεταβλητή που αρχικά έχει μία από τις δύο τιμές της (π.χ. ΨΕΥΔΗΣ) και όταν συμβεί το επιθυμητό γεγονός, π.χ. βρεθεί το κλειδί στον πίνακα, αλλάζει τιμή, πέφτει η σημαία.  **Αλγόριθμος** Σειριακή\_αναζήτηση4  **Δεδομένα //Ν, Α//**  ι**←**1  θέση**←**0  βρέθηκε**←ΨΕΥΔΗΣ**  **Διάβασε** κλειδί  **Όσο** ι<=Ν **ΚΑΙ** βρέθηκε=**ΨΕΥΔΗΣ** **επανάλαβε**  **Αν** Α[ι]=κλειδί **τότε**  θέση←ι  βρέθηκε**←.....................**  **Αλλιώς**  ι←........................  **Τέλος\_Αν**  **Τέλος\_επανάληψης**  **Αν** βρέθηκε =................... **τότε**  **Εμφάνισε** "Δεν βρέθηκε"  **Αλλιώς**  **Εμφάνισε** θέση  **Τέλος\_αν**  **Τέλος** Σειριακή\_αναζήτηση4 |

**Δραστηριότητα 4.** Σας δίνεται αρχείο εισόδου με 200 λαχνούς από το χορό του ΕΠΑΛ και τα δώρα που κερδίζουν. Να γράψετε πρόγραμμα που θα κάνει τα εξής:

1. Διαβάζει και αποθηκεύει σε δύο παράλληλους πίνακες τους λαχνούς (Λ) που κερδίζουν και τα δώρα (Δ)

2. Διαβάζει το λαχνό που αγόρασε κάποιος για την ενίσχυση των μαθητών.

3. Αναζητά στον πίνακα τον λαχνό και εμφανίζει κατάλληλο μήνυμα αν κέρδισε τον λαχνό που κέρδισε και το αντίστοιχο δώρο, αν δεν κέρδισε "Δυστυχώς δεν κερδίσατε".

4. Ρωτά τον χρήστη αν θέλει να συνεχίσει και αν η απάντηση είναι ΝΑΙ επαναλαμβάνει τα βήματα 2 και 3.

**Δραστηριότητα 5.**

Έχουμε 2 πίνακες Ν (ας πούμε 5) στοιχείων στους οποίους είναι αποθηκευμένα αντίστοιχα Ν τίτλοι τραγουδιών και οι εμφανίσεις τους στο Youtube. Για παράδειγμα:

**Τίτλος (Τιτ[Ν])**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Τι λάθος κάνω | Της λήθης το πηγάδι | Πριγκηπέσα | Φωτοβολίδα | Αόρατη πληγή |
| **Εμφανίσεις (Εμφ[Ν])** | | | | |
| 4900000 | 3253000 | 4900000 | 2500000 | 500000 |

Θέλουμε να βρούμε ποιο ή ποια τραγούδια είχαν τις ***περισσότερες εμφανίσεις***. Γράψτε στο προγραμματιστικό περιβάλλον του Διερμηνευτή το αντίστοιχο πρόγραμμα σε ΓΛΩΣΣΑ που να επιλύει το πρόβλημα.