«ΑΛΓΟΡΙΘΜΟΙ ΚΑΙ ΠΟΛΥΠΛΟΚΟΤΗΤΑ»

1^{Η} ΕΡΓΑΣΙΑ

(ΒΑΡΥΤΗΤΑ 50% ΣΤΗΝ ΤΕΛΙΚΗ ΒΑΘΜΟΛΟΓΙΑ)

Υλοποίηση και έλεγχος απόδοσης αλγορίθμων ταξινόμησης

- Θα πρέπει να υλοποιήσετε τον αλγόριθμο ταξινόμησης Quicksort σε γλώσσα προγραμματισμού δικής σας επιλογής. Αναφορικά με την υλοποίηση του αλγορίθμου Quicksort θα υλοποιήσετε τρεις παραλλαγές ως προς την επιλογή του στοιχείου οδηγού (pivot):
    1. Επιλογή πάντα του πρώτου στοιχείου του πίνακα (ή υποπίνακα).
    2. Τυχαία επιλογή από όλα τα στοιχεία του πίνακα (ή υποπίνακα).
    3. Τυχαία επιλογή τριών στοιχείων από όλα τα στοιχεία του πίνακα (ή υποπίνακα) και στη συνέχεια επιλογή του μεσαίου σε αύξουσα διάταξη εκ των τριών στοιχείων ως στοιχείο οδηγός.
- Επίσης υλοποιήστε και τον εξής συνδυασμό αλγορίθμων ταξινόμησης. Αρχικά η ταξινόμηση πραγματοποιείται με την εφαρμογή του αλγορίθμου Quicksort με τυχαία επιλογή του στοιχείου οδηγού από τα όλα τα στοιχεία του πίνακα και όταν το επίπεδο αναδρομής υπερβεί ένα συγκεκριμένο όριο, το οποίο θα δίνεται ως παράμετρος εισόδου, η ταξινόμηση του υποπίνακα θα γίνεται με τη χρήση του αλγορίθμου Heapsort.
- Για τον έλεγχο της απόδοσης των αλγορίθμων, θα πρέπει να χρησιμοποιήσετε δύο μετρητές *compareCount* και *swapCount*. Ο πρώτος θα πρέπει να αυξάνεται κατά ένα κάθε φορά που εκτελείται σύγκριση μεταξύ δύο στοιχείων ενώ ο δεύτερος θα αυξάνεται επίσης κατά ένα κάθε φορά ένα στοιχείο ανταλλάσσει θέση με κάποιο άλλο ή μετακινείται.
- Το κύριο πρόγραμμα θα έχει ένα επαναληπτικό βρόχο και σε κάθε επανάληψη θα εκτελεί τους αλγορίθμους (και τις τυχόν παραλλαγές του) για ένα διαφορετικό μέγεθος εισόδου. Συγκεκριμένα, το πλήθος των στοιχείων του πίνακα εισόδου θα κυμαίνεται στο διάστημα [50,2000] με βήμα αύξησης 50. Για κάθε μέγεθος εισόδου, θα γεμίζετε τον πίνακα με τυχαία στοιχεία και στη συνέχεια θα εκτελείτε τους αλγορίθμους πάνω στον ίδιο πίνακα. Θα επαναλάβετε δέκα φορές τη διαδικασία της τυχαίας επιλογής στοιχείων εισόδου και κατόπιν της εκτέλεσης των αλγορίθμων. Στο τέλος αυτών των δέκα επαναλήψεων για το ίδιο μέγεθος εισόδου, θα πρέπει να έχει υπολογιστεί επίσης η μέγιστη, η ελάχιστη και η μέση τιμή των μετρητών *compareCount* και *swapCount*. Επίσης πειραματιστείτε για διάφορες τιμές του επιπέδου αναδρομής στο οποίο γίνεται η αλλαγή των αλγορίθμων ταξινόμησης. Βρείτε τη βέλτιστη τιμή σύμφωνα με τα πειραματικά αποτελέσματα.
- Σχεδιάστε διαγράμματα που θα απεικονίζουν τη μεταβολή της ελάχιστης, μέγιστης και μέσης τιμής των μετρητών *compareCount* και *swapCount* συναρτήσει του μεγέθους εισόδου.

Υλοποίηση του αλγορίθμου Dijkstra και παραλλαγής του

- Δημιούργησε πλήρη μη κατευθυνόμενα γράφηματα 5 μέχρι 100 κόμβων με βήμα αύξησης 5, με τα βάρη στις ακμές των γραφημάτων να είναι τυχαία επιλεγμένοι θετικοί ακέραιοι αριθμοί με μέγιστη τιμή 100. Για κάθε πλήθος κόμβων, δημιουργήστε 3 τυχαία γραφήματα.

- Υλοποιήστε τον αλγόριθμο Dijkstra με χρήση δυαδικού σωρού ελαχίστων σε γλώσσα προγραμματισμού της επιλογής σας για την εύρεση της συντομότερης διαδρομής μεταξύ δύο συγκεκριμένων κόμβων και συγκεκριμένα αυτών που η απευθείας ακμή μεταξύ τους έχει το μέγιστο κόστος μεταξύ όλων των ακμών. Αν υπάρχουν περισσότερες του ενός ακμές με το ίδιο μέγιστο κόστος, επιλέξτε τυχαία μία από αυτές.
- Υλοποιήστε επίσης τον αλγόριθμο Dijkstra διπλής κατεύθυνσης για την εύρεση της συντομότερης διαδρομής πάλι μεταξύ δύο συγκεκριμένων κόμβων. Στην τεχνική αυτή, γίνεται «ταυτόχρονη» εκτέλεση του αλγόριθμου Dijkstra τόσο από την αφετηρία όσο και από τον προορισμό με ενναλακτική εκτέλεση των βημάτων των δύο εκτελέσεων. Η πρώτη φάση ολοκληρώνεται όταν ο ίδιος κόμβος οριστικοποιηθεί ως προς τη συντομότερη διαδρομή του και από τις δύο ταυτόχρονες εκτελέσεις. Στη συνέχεια ακολουθεί η δεύτερη φάση όπου με βάση τον παραπάνω κοινό κόμβο και τους κόμβους που έχουν απομείνει στους δυαδικούς σωρούς των δύο εκτελέσεων, υπολογίζεται η συντομότερη διαδρομή μεταξύ της αφετηρίας και του προορισμού. Συμπληρώστε τις λεπτομέρειες της δεύτερης φάσης.
- Για τις δύο παραπάνω παραλλαγές, σχεδιάστε τον χρόνο εκτέλεσης των αλγορίθμων συναρτήσει του πλήθους των κόμβων του γραφήματος. Για γραφήματα με το ίδιο πλήθος κόμβων, υπολογίστε το μέσο χρόνο εκτέλεσης.

Εύρεση του πλησιέστερου ζεύγους σημείων σε επίπεδο.

Δίνονται *n* σημεία στον δισδιάστατο χώρο. Σχεδιάστε και υλοποιήστε ένα αλγόριθμο διαίρει και βασίλευε πολυπλοκότητας χειρότερης περίπτωσης *O(nlogn)* ο οποίος θα επιστρέφει το ζεύγος των σημείων που έχουν την ελάχιστη απόσταση μεταξύ όλων των σημείων του επιπέδου. Σχεδιάστε ένα γράφημα με το χρόνο εκτέλεσης του αλγορίθμου συναρτήσει του πλήθους των σημείων στο επίπεδο. Για αυτό το σκοπό δημιουργήστε 20 στιγμιότυπα εισόδου τυχαία επιλεγμένων 10 μέχρι 200 σημείων με βήμα αύξησης 10. Επαληθεύστε επίσης την πολυπλοκότητα χρόνου του αλγορίθμου με βάση τα πειραματικά αποτελέσματα. Συγκεκριμένα, υπολογίστε τη μικρότερη τιμή της σταθεράς που πολλαπλασιάζει την παράσταση πολυπλοκότητας και «κρύβεται» στο συμβολισμό Ο έτσι ώστε πράγματι η πολυπλοκότητα αυτή να «λειτουργεί» ως άνω όριο του χρόνου εκτέλεσης που προέκυψε από τα πειράματα. Σημειώνεται επίσης ότι οι διαφάνειες του μαθήματος που βρίσκονται αναρτημένες στο Gunet, περιέχουν αναλυτική παρουσίαση του αλγορίθμου επίλυσης του εν λόγω προβλήματος.

Παράδοση Εργασίας

Η παράδοση της εργασίας θα γίνει μέσω της πλατφόρμας του Gunet. Η προθεσμία παράδοσης της εργασίας ορίζεται η **21/1/2024.** Θα πρέπει να παραδώσετε τον πηγαίο κώδικα, αναλυτικές οδηγίες εκτέλεσης του κώδικα, αναλυτική τεκμηρίωση της υλοποίησης και των αποτελεσμάτων των πειραμάτων σας. Η εργασία μπορεί να εκπονηθεί από ομάδα μέχρι **δύο ατόμων.**