ΕΡΩΤΗΜΑ 1

Το πρόγραμμα ξεκινάει με τη δημιουργία της γεννήτριας τυχαίων αριθμών

που περιγράφεται στην εκφώνηση. Στη συνέχεια αφού συμπεριλάβουμε τις

απαραίτητες βιβλιοθήκες που θα χρειαστούμε καθώς και το header αρχείο "list.h" το οποίο περιέχει τη δήλωση της κλάσης που αναπαριστά την δομή της συνδεδεμένης λίστας που θα χρησιμοποιηθεί στο πρόγραμμα (η υλοποίηση που χρησιμοποιείται αντιστοιχεί σε αυτή του βιβλίου “Δομές Δεδομένων, Αλγόριθμοι και Εφαρμογές στη C++” με τη διαφορά πως στο ιδιωτικό πεδίο

data έχουμε έναν πίνακα δύο στοιχείων ώστε να μπορούμε να κρατάμε την πληροφορία που ζητείται στη λίστα Histogram) παίρνουμε το μέγεθος της λίστας L από τον χρήστη. Στην δήλωση της κλάσης ListNode έχουμε φτιάξει δύο constructors. Ο πρώτος (με όρισμα έναν ακέραιο) αφορά την δημιουργία της λίστας L ενώ ο δεύτερος (με όρισμα μία άλλη λίστα) την δημιουργία της λίστας Histogram. Έτσι, δημιουργούμε αρχικά τη λίστα L και την γεμίζουμε με τυχαίους ακεραίους (στη συγκεκριμένη περίπτωση σε εύρος 0-20). Στη συνέχεια χρησιμοποιούμε τη μέθοδος sort() της κλάσης ListNode η οποία ταξινομεί την λίστα L και θα μας βοηθήσει ώστε να είναι πιο εύκολη η δημιουργία της λίστας Histogram. Έτσι δημιουργούμε τελικά την λίστα Histogram χρησιμοποιώντας τον δεύτερο constructor της κλάσης ListNode. Ο constructor αυτός προσπελαύνει τα στοιχεία της λίστας L, αποθηκεύει στο πεδίο data[0] τον κάθε ακέραιο και στο πεδίο data[1] το πλήθος των εμφανίσεων του. Αφού τελειώσει η προσπέλαση εμφανίζουμε τα αποτελέσματα χρησιμοποιώντας τα δεδομένα της λίστας Histogram.

ΕΡΩΤΗΜΑ 2

Ανάλογα με το πρώτο ερώτημα ξεκινάμε με τη δημιουργία της γεννήτριας τυχαίων αριθμών όπως περιγράφεται. Στη συνέχεια συμπεριλαμβάνουμε το αρχείο “heap.h” το οποίο περιέχει τις κλάσεις σωρού ελαχίστων και μεγίστων αντίστοιχα (έχει χρησιμοποιηθεί η υλοποίηση του σωρού μεγίστων που υπάρχει στο βιβλίο η οποία έχει προσαρμοστεί και για σωρό ελαχίστων αντίστοιχα). Αφού πάρουμε το πλήθος των αριθμών από τον χρήστη δεσμεύουμε δυναμικά μνήμη για τους δύο πίνακες heap\_min και heap\_max και τους αρχικοποιούμε με τους τυχαίους αριθμούς τύπου float (στην επανάληψη for ξεκινάμε τον μετρητή από 1 για να μπορούμε εύκολα να έχουμε πρόσβαση στο παιδί και τον γονέα κάθε κόμβου). Έτσι, δημιουργούμε τους σωρούς ελαχίστων και μεγίστων αντίστοιχα και τους αρχικοποιούμε με τους πίνακες heap\_min και heap\_max αντίστοιχα με τη χρήση της μεθόδου initialize(). Με τη χρήση της μεθόδου findSum() αφαιρούμε τα δύο μεγαλύτερα στοιχεία του σωρού μεγίστων και τα δύο μικρότερα του σωρού ελαχίστων αντίστοιχα και εισάγουμε το άθροισμά τους πίσω στο σωρό. Αυτή η διαδικασία επαναλαμβάνεται έως ότου αφαιρεθεί και το τελευταίο στοιχείο των σωρών (το οποίο θα είναι και το άθροισμα όλων των στοιχείων) και υπάρξει μια εξαίρεση out\_of\_range() καθώς οι μέθοδοι deleteMax() και deleteMin() αντίστοιχα θα προσπαθήσουν να αφαιρέσουν ένα στοιχείο από έναν κενό σωρό. Τέλος, εμφανίζουμε τα αποτελέσματα που προκύπτουν από τη χρήση κάθε σωρού (εμφανίζουμε τα max1 και min1 αντίστοιχα καθώς το τελευταίο στοιχείο κάθε σωρού θα αφαιρεθεί με τη πρώτη deleteMax() και deleteMax() αντίστοιχα) και τερματίζουμε την επαναληπτική διαδικασία αφού απενεργοποιήσουμε τους δύο σωρούς.