ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS
Σταμάτης Χουλιαράς
3190221
Ιωάννης Βουσβουκής
3190027

Δομές Δεδομένων (Εργασία 2)

Θέμα Α:

Αρχικά για την υλοποίηση του θέματος Α φτιάξαμε τρεις κλάσεις

1. City που είναι και η πιο βασική, έχει μεθόδους όπως την getName, getPopulation και getID. Μία επιπλέον μέθοδος είναι η calculateDensity η οποία πέρνει ως ορίσματα τα κρούσματα κορωνοϊού και τον πλυθησμό και υπολογίζει την πυκνότητα ανά 50.000 άτομα.
2. CityCompare που είναι βοηθητική κλάση και συγκρίνει δύο πόλεις ως εξής: Αρχικά ως προς την πυκνότητα στη συνέχεια αν η πυκνότητα είναι η ίδια ως προς το όνομα (αλφαβητικά δηλαδή) και τέλος αν τα όνοματα είναι ίδια τότε πέρνει την πόλη με το μικρότερο ID.
3. Quicksort που υλοποιεί τον αλγόριθμο quicksort που συνεχώς χωρίζει τον αρχικό πίνακα σε υποπίνακες με αναδρομή αλλά η διαφορά με την mergesort είναι οτι στο τέλος δεν χρειάζεται συγχώνευση.

Η mainA είναι πολύ απλή. Ζητάει να δώσει ο χρήστης το όνομα του αρχείου για να το ανοίξει και τον αριθμό των k πόλεων με την μεγαλύτερη πυκνότητα. Τελικά καλεί την κλάση quicksort και εμφανίζει τα αποτελέσματα στην οθόνη.

Θέμα Β:

Για το θέμα Β φτιάξαμε μόνο μία κλάση την HeapPriority η οποία αρχικοποιείται με ένα πίνακα που περιέχει στοιχεία τύπου City, ένα πίνακα χιλίων στοιχείων που κάθε index αναπαριστά ένα ID πόλης και το περιεχόμενο είναι η τοποθεσία της πόλης με αυτό το ID στο δέντρο, ένα μετρητή για να ξέρουμε το μέγεθος του δέντρου και τη χωρητικότητα του δέντρου. Έχει τις βασικές μεθόδους ενός δυαδικού δέντρου όπως:

1. swim που «ανεβάζει» ένα στοιχείο του δέντρου από «κάτω» προς τα «πάνω» .
2. sink που κάνει την αντίθετη πράξη του swim.
3. swap που ανταλάζει δύο στοιχεία του δέντρου.

Αλλά έχει και επιπλέον μεθόδους όπως την insert η οποία εισάγει στο τελευταίο index του δέντρου το νέο στοιχείο και μετά κάνει swim, την resize η οποία χρησιμοποιείται μόνο αν γεμίσει το 75% του δέντρου, την max η οποία επιστρέφει την «ρίζα» του δέντρου, την getmax η οποία κάνει το ίδιο με την max μόνο που αυτή τη φορά διαγράφει τη ρίζα και τέλος η remove η οποία πέρνει ως παράμετρο ένα ID μιας πόλης και διαγράφεται από το δέντρο. Η διαδικασία της διαγραφής γίνεται ως εξής: πέρνουμε το index του στοιχείου με την βοήθεια του πίνακα με τα IDs (id_heap) και κανουμε swap το στοιχείο αυτό με το τελευταίο στοιχείο του δέντρου και στη συνέχεια το κάνουμε sink έχοντας φροντίσει να μειώσουμε τον μετρητή που δείχνει το μέγεθος του πίνακα ώστε το στοιχειο που αφαιρείται να μην λαμβάνεται υπόψιν.

Θέμα Γ:

Στο θέμα Γ η insert είναι η μόνη μέθοδος που υλοποιείται διαφορετικά. Πιο συγκεκριμένα, ο χρήστης δίνει τον μέγιστο αριθμό k των πόλεων που θέλει να κρατήσει στο δέντρο και μέχρι να μπουν στο δέντρο k πόλεις η insert λειτουργεί όπως πριν, αλλά όταν εισαχθούν k στοιχεία τότε ώστε να μπορούμε να κρατάμε ποιο στοιχείο έχει το λιγότερο priority στο δέντρο, σε κάθε είσοδο βρήσκουμε το min στοιχείο και αν το στοιχείο που πάει να εισαχθεί στο δέντρο έχει μεγαλύτερο priority από το min στοιχείο τότε γίνεται swap με αυτό και γίνεται swim, αλλιώς αγνοήται και πάει στο επόμενο στοιχείο εισόδου.

Θέμα Δ:

Στο θέμα Δ πάλι η insert λειτουργεί διαφορετικά αλλά υπάρχει μία επιπλέον μέθοδος, η sort. Η insert αρχικά ελέγχει αν χρειάζεται να γίνει resize και στη συνέχεια εισάγει ένα στοιχείο το κάνει swim και καλέι την sort και ύστερα σε μία μεταβλητή City median αποθηκεύει το median του δέντρου ανάλογα με το αν το μέγεθος του πίνακα είναι άρτιος ή περιττός αριθμός. Η μέθοδος sort λειτουργεί πολύ απλά. Το μόνο που κάνει είναι να δημιουργήσει ένα νέο πίνακα City στον οποίο εισάγει κάθε φορά την ρίζα και μετά την διαγράφει από το δέντρο. Έτσι ο νέος πίνακας είναι ταξινομιμένος. Τέλος, τα στοιχεία του νέου πίνακα αντιγράφονται στον αρχικό πίνακα με αποτέλεσμα ο αρχικός πίνακας να είναι ταξινομιμένος άρα μπορούμε μετά με ευκολία να βρούμε το median.