**Στην τελευταία παράγραφο εξηγώ πως διαβάζεται το αρχείο

Μέρος Α:

Το πρώτο κομμάτι της εργασίας μου, το υλοποίησα με την μέθοδο ταξινόμησης QuickSort, διότι το ΑΜ μου τελιώνει σε περιττό αριθμό(3190179).Συγκεκριμένα, για τις ανάγκες του Α μέρους δημιούργησα την κλάση QuickSort, η οποία περιέχει μέσα την partition, την sort και 4 διαφορετικές swap , για να μπορέσω να αντιμεταθέτω ταυτόχρονα τα στοιχεία κάποιων πόλεων όταν αυτό είναι αναγκαίο. Η sort χρησιμοποιείται για να κάνει την αναδρομή της μεθόδου «διαίρει και βασίλευε» (έτσι μεταφράζεται ουσιαστικά η QuickSort) και η partition για να συγκρίνει τα δεδομένα με την εξής λογική:
Ορίζουμε ένα σημείο αναφοράς, από το οποίο ξεκινάνε όλες οι συγκρίσεις(το λεγόμενο pivot). Το pivot χωρίζει τη λίστα σε 2 μέρη. Το αριστερό και το δεξιό μέρος. Η μέθοδος αυτή λειτουργεί έτσι ώστε το αριστερό μέρος να αποτελείται από στοιχεία που να είναι μικρότερα από το pivot και το δεξιό μέρος να αποτελείται από στοιχεία που να είναι μεγαλύτερα από το pivot.Στη δικιά μου QuickSort, το pivot ορίζεται ως το τελευταίο στοιχείο της λίστας και συγκρίνεται με όλα τα στοιχειά από το πρώτο μέχρι και το προτελευταίο. Κάθε φόρα που το pivot θα συναντά ένα στοιχείο μεγαλύτερο από αυτό, θα ενεργοποιούνται οι 4 swap που θα αντιμεταθέτουν τη θέση των id, του ονόματος, του πληθυσμού , του πλήθους κρουσμάτων covid και της πυκνότητας της εκάστοτε πόλης.
Η σύγκριση και η ταξινόμηση των στοιχείων γίνεται με βάση την πυκνότητα των κρουσμάτων της κάθε πόλης, σύμφωνα με την υπόδειξη της εκφώνησης. Για το σκοπό αυτό δημιούργησα μέσα στην QuickSort την μέθοδο calculateDensity, η οποία υπολογίζει με ακρίβεια 2 δεκαδικών ψηφίων τη πυκνότητα της κάθε πόλης. Επίσης, μέσα στην QuickSort υπάρχει και η μέθοδος StCompare η οποία χρησιμεύει στο να συγκρίνει 2 πόλεις(δηλαδή strings) που έχουν την ίδια πυκνότητα. Κλείνοντας την αναφορά για το μέρος Α, πρέπει να αναφέρω και την υλοποίηση της διεπαφής compareTo , η οποία χρησιμοποιείται για να συγκρίνει 2 πόλεις που έχουν την ίδια πυκνότητα με 1^{ο} κριτήριο το όνομα τους (και γι' αυτό το λόγο καλείται η StCompare) και με 2^{ο} κριτήριο το id τους (αυτό συμβαίνει αν τύχει δυο διαφορετικές πόλεις να έχουν και ίδια πυκνότητα και ίδιο όνομα).

Μέρος Β και Γ:

Στο μέρος Β και συγκεκριμένα στην κλάση PQ υλοποίησα τη μέθοδο remove, η οποία δέχεται σαν μοναδικό όρισμα έναν ακέραιο αριθμό. Ο αριθμός αυτός αποτελεί το id κάποιας πόλης, της οποίας το όνομα της θα επιστρέφεται από τη μέθοδο και στη συνέχεια η πόλη αυτή με όλα της τα στοιχεία θα διαγράφεται από τη σωρό. Αρχικά, η πολυπλοκότητα της remove είναι O(n), λόγω του for loop που έχω χρησιμοποιήσει προκειμένου να εντοπίσω την πόλη, η οποία έχει ως όρισμα το συγκεκριμένο id. Για το σκοπό αυτό, χρησιμοποίησα 2 «tricks». Το 1^{ο} από αυτά ήταν η δημιουργία ενός επιπλέον εικονικού στοιχείου στη σωρό, στο οποίο το id είναι το όρισμα της συνάρτησης και τα υπόλοιπα στοιχεία είναι πλασματικά για τον σκοπό της άσκησης. Το 2^{ο} «κόλπο» είναι η χρησιμοποίησης της μεθόδου help, η οποία έχει κατά κάποιον τρόπο τη λειτουργεία του comparator.compare, αλλά με την διαφορά ότι η χρησιμοποίηση της είναι για να βρίσκει πότε 2 στοιχεία έχουν το ίδιο id.Η help έχει πάντα 2 ορίσματα τύπου City, εκ των οποίων το

ένα από τα δύο, είναι πάντα το πλασματικό στοιχείο που δημιούργησα στην αρχή της μεθόδου. Το άλλο στοιχείο είναι κάθε φορά διαφορετικό, αφού κάθε φορά δίνεται ένα στοιχείο, με βάση τη σειρά προτεραιότητας της σωρού. Μόλις εντοπιστεί ότι κάποιο στοιχείο από τη σωρό έχει ως id, το id που έχει που έχει δοθεί ως όρισμα στη help , γίνονται οι παρακάτω 4 ενέργειες:

1. Μέσω της swap αλλάζω τη θέση του ζητούμενου στοιχείου με αυτού του 1ου και το βάζω στην κορυφή της σωρού.
2. Αφού από την αρχή της όλης διαδικασίας έχω αρχικοποιήσει μια μεταβλητή root τύπου City ως null, τώρα της περνάω το πρώτο στοιχείο της σωρού μέσω της Max.
3. Κάλω την getMax όχι για να μου επιστρέψει το πρώτο στοιχείο της σωρού(παρ΄όλο που αυτή είναι η δουλεία της), αλλά για να «τακτοποιήσει» και πάλι τη σωρό, όσο αναφορά τη σειρά προτεραιότητας , πάντα με βάση την πυκνότητα κρουσμάτων των πόλεων.
4. Εφόσον πραγματοποιήσω όλα τα παραπάνω, κάνω break από την δομή επανάληψης, γνωρίζοντας ότι το id είναι μοναδικό και συνεπώς δεν υπάρχει λόγος για περαιτέρω επαναλήψεις και ελέγχους.

Στο τέλος αυτών των 4 βημάτων επιστρέφω την πόλη που αφαίρεσα και η μέθοδος ολοκληρώνεται.
Σε περίπτωση που το id που έχει δοθεί ως όρισμα στη help, δεν υπάρχει στη σωρό, η μέθοδος επιστρέφει null, χωρίς κανένα αρνητικό αντίκτυπο και καμία επιπλοκή.

Στο μέρος Γ, πραγματοποιείται πάνω κάτω η ίδια διαδικασία με το μέρος Α, όσον αφορά το διάβασμα του αρχείου. Από εκεί και πέρα, αλλάζει εντελώς η αποθήκευση των στοιχείων, αφού τώρα δεν χρησιμοποιώ πίνακες, αλλά τη σωρό που δημιούργησα στο μέρος Β. Συγκεκριμένα, χρησιμοποιώ κατά κόρον την μέθοδο insert προκειμένου να εισάγω τα στοιχεία μου στη σωρό , με τέτοιο τρόπο ώστε να ταξινομούνται κιόλας. Επίσης, χρησιμοποιώ τη μέθοδο removeLast , της οποίας η διεπαφή υλοποιείται μέσα στην κλάση PQ. Η removeLast, δέχεται ως όρισμα ένα αντικείμενο τύπου City και ενεργεί ως εξής: Αρχικά, καλεί την insert προκειμένου να εισάγει το νέο στοιχείο στη σωρό. Έπειτα αντικαθιστά το τελευταίο στοιχείο της σωρού με null, μειώνοντας ταυτόχρονα το μέγεθος της σωρού κατά 1. Με αυτόν τον τρόπο εξασφαλίζουμε ότι η σωρός θα έχει πάντα το πολύ k ενεργά αντικείμενα.
Ειδικότερα: Αρχικά, διαβάζω μία φορά το αρχείο προκειμένου να μετρήσω πόσες σειρές περιέχει το αρχείο που διαβάζω. Αφού το κάνω αυτό, κλείνω το αρχείο και διαβάζω από το πληκτρολόγιο τη μεταβλητή k. Εφόσον αυτή είναι μικρότερη ή ίση με το πλήθος των γραμμών του αρχείου, συνεχίζω κανονικά την εκτέλεση του προγράμματος. Σε διαφορετική περίπτωση το πρόγραμμα τερματίζεται και στην οθόνη εμφανίζεται μήνυμα λάθους. Εφόσον η μεταβλητή k είναι αυτή που πρέπει να είναι, ξαναδιαβάζω το αρχείο. Πριν το κάνω αυτό όμως ορίζω μία μεταβλητή sec_counter ίση με το 1.Κάθε φορά που έχω τελιώσει με την ανάγνωση των στοιχείων της κάθε σειράς ελέγχω αν η μεταβλητή αυτή είναι μικρότερη ή ίση με την μεταβλητή k. Αν είναι τότε απλά κάνω insert το στοιχείο στη σωρό, αλλιώς χρησιμοποιώ τη removeLast. Με αυτόν τον τρόπο εξασφαλίζω ότι η σωρός δεν θα έχει ποτέ παραπάνω στοιχεία από το πλήθος k που διαβάζει ο χρήστης, γιατί από τη στιγμή που ο sec_counter ξεπεράσει τον αριθμό k, κάθε φορά που θα εισάγει ένα στοιχείο, ταυτόχρονα και θα αφαιρεί το τελευταίο.
Δεδομένου ότι και η insert έχει πολυπλοκότητα O(logn), συνεπώς και η removeLast έχει την ίδια πολυπλοκότητα, αφού σαν μέθοδος περιέχει την insert και απλά αφαιρεί το τελευταίο στοιχείο της λίστας. Άρα, λόγω των n γραμμών του αρχείου το μέρος Γ αναμένεται να έχει πολυπλοκότητα O(nlogn), αφού για να ελέγξουμε όλες τις γραμμές του

αρχείου θα χρειαστούμε n επαναλήψεις και η κάθε εισαγωγή απαιτεί πολυπλοκότητα O(logn),η συνολική πολυπλοκότητα του προγράμματος θα είναι O(nlogn).
Τέλος, αν το k είναι αρκετά μικρότερο από το πλήθος των γραμμών, είμαστε σίγουρα σε πιο πλεονεκτική θέση από ότι αν χρησιμοποιούσαμε ταξινόμηση με QuickSort, γιατί αφενός στην περίπτωση αυτή, η μέθοδος έχει πολυπλοκότητα O(nlogn) και αφετέρου με τη χρήση της μεγιστοστραφούς σωρού έχουμε πολύ λιγότερα στοιχεία να συγκρίνουμε και συνεπώς μικρότερο υπολογιστικό κόστος.

Εισαγωγή αρχείου και μεταβλητής k:

Για την εισαγωγή της μεταβλητής αυτής, θα εμφανιστεί στην οθόνη κατάλληλο μήνυμα μόλις «τρέξετε» το πρόγραμμα. Το μήνυμα αυτό θα σας ζητάει απλά να πληκτρολογήσετε έναν αριθμό δίπλα στο μήνυμα της οθόνης.
Κλείνοντας, όσον αφορά το διάβασμα του αρχείου έχω τοποθετήσει το inputfile.txt μέσα ε έναν φάκελο test, και προκειμένου για να ελέγξετε την ορθότητα του κώδικα μου, πρέπει να βάλετε το αρχείο που θέλετε να δοκιμάσετε μέσα στον φάκελο αυτόν.

Από εκεί και πέρα, απλά θα δώσετε σαν είσοδο το όνομα του αρχείου.

Πχ: inputfile.txt (το συγκεκριμένο αρχείο βρίσκεται μέσα στον φάκελο που σας έστειλα)

Υ.Γ.:

"Όταν προσπάθησα στον κώδικα να τρέξω το αρχείο μόνο του (δηλαδή το είχα «χύμα» μέσα στο src, το πρόγραμμα μου έβγαζε ζητήματα ως προς την ανάγνωση των περιεχομένων των αρχείων. Μέσα στον φάκελο, όμως όλα λειτουργούσαν καλά. Γι αυτό το λόγο σας ζήτησα να βάλετε τα αρχεία σας μέσα στον φάκελο src. Ελπίζω όλα να κυλήσουν ομαλά και από τον δικό σας υπολογιστή.