ΕΡΓΑΣΙΑ 2

ΔΟΜΕΣ ΔΕΔΟΜΕΝΩΝ ΑΜ: Αριστείδης Καραγιαννάκος 3220066, Δανέζη Έλλη-Μαρία 3220037

ΘΕΜΑ Α:

Quicksort: η quicksort χωριζει τον πινακα σε δυο μεσα από ένα στοιχειο σε αυτή την περιπτωση το larg βαση αυτου του στοιχειου ο πινακας ταξινομειται δλδ γυρο από αυτό το σημειο τοποθετουνται αριστερα του τα μικροτερα στοιχεια και δεξια του τα μεγαλυτερα φτιαχνοντας και τους δυο υποπινακες ο διαχωρισμος γινεται με αναδρομικο τροπο καλωντας δυο φορες την quicksort αφου τοποθετηθει σωστα το στοιχειο larg.

Η partition απλως προσαπθει να βαλει την larg στην σωστη θεση και αφου περνα από όλα τα σημεια τα ταξινομει βασει αυτου.

Readcityy: η read δέχεται ενα αρχειο τυπου txt το οποιο το διαβαζει γραμμη γραμη και αποθηκευη εάν αντικείμενο ανα γραμμή.

Στην γραμμή που θα διαβάζει φτιάχνει εναν πίνακα στην οποία πρωτα θα αποθηκευη στοιχεία κάθε φορα που διαβάζει " ".Οταν τα αποθήκευση διαβάζει τον πίνακα αν αυτο που αποθηκευη δεν είναι αριθμός μέσω του try catch το προσθετη στο ονομα αλλιως αναλόγως με το counter το βαζει στην σωστη μεταβλητή.

Στην main συνάρτηση πρώτα διαβάζουμε το k το οποίο είναι ο αριθμός ο οποίος ο χρήστης Ειναι να εμφανίσει μετα φτιάχνουμε ενα νεο αντικείμενο η οποία διαβάζει το αρχείο που εχει της πολυ και της τοποθετει σε εναν πινακα ο οποιος δεχεται αντικειμενα City.αφου γινει αυτο τα εμγανιζει ταξινομιμενα αφου εχουμε καλεση την quicsort μεχρι το στοιχειο σε αριθμο κ. αν το

κ ειναι μεγαλυτερο απο το lenthg του πινα εφανιζει μηνημα λαθους.

Θεμα Β:

remove: καθως οιλοποιησουμε την ουρα φτιαχναμε έναν πινακα ο οποιος ειχε μεσα ως στοιεια την θεση που βρισκεται το id και σαν θεση τοιυ πινακα ειχαμε το id το ιδιο .

όταν πιγενουμε στην remove μεσα στην ουρα βαζουμε τον πινακα με το συκεκριμενη θεση id και μετα αλλαζουμε την τελευταια θεση του πινακα με την συκεκριμενη και ξαναφτιαχνουμε την ουρα σωστα μεσω τις sink η οποια εχει position του στοιχειου που περιεχει ο πινακας στην θεση id

Θέμα Γ:

Αρχικά, λαμβάνουμε από τον χρήστη τον αριθμό των πόλεων <<k>> που θέλουμε να εμφανίζει καθώς και το αρχείο txt των πόλεων.Έπειτα, διαβάζουμε γραμμή γραμμή το αρχείο και αν και αν τα στοιχεία που διαβάσαμε για την πόλη είναι σωστά τότε δημιουργούμε ένα αντικείμενο τύπου City και το προσθέτουμε στην PQ.Με την χρήση της μεθόδου swim εξασφαλίζουμε τα στοιχεία που μπαίνουν στην ουρά να είναι πάντα σε φθίνουσα σειρά και σε χρόνο O(logk).
Σε κάθε νέα προσθήκη ελέγχουμε αν το k-οστο στοιχείο είναι μικρότερο ή μεγαλύτερο από το μέγιστο στοιχείο της ουράς.Αν είναι μεγαλύτερο διαγράφουμε το μέγιστο και προσθέτουμε το καινούριο.
Τέλος, με χρήση δομής επανάληψης for εκτυπώνουμε τα k στοιχεία της PQ από το μικρότερο στο μεγαλύτερο με τη getmin().
πολυπλοκότητα εκτέλεσης ReadFile_MerosG + πολυπλοκότητα εκτέλεσης for loop= (k*O(logk)+(n-k)* 2*O(logk)) + (3k+2)*O(logk)=(2*n)* O(logk)=O(logk) , όπου n αριθμός πόλεων.
Ο αλγόριθμος για το μέρος Γ είναι αποδοτικός, αφού για μικρά k προστίθεται ελάχιστος χρόνος εκτέλεσης.

Θεμα Δ: Στο θεμα δ εχουμε χρησιμοιποηση δυο ουρες μια δυναμικη που αποθηκευη μεχρι κ και μια που αποθηκευη όλα τα στοιχεια.

Όταν αυτή που αποθηκευη όλα τα στοιχεια τελιωσει καλουμε την median η οποια βλεπει αν ο αριθμος των στοιων είναι ζυγος η μονος αναλογος καλουμε την getmin και περνουμε μεχρι την μεση αν είναι μονος απλως εμφανιζουμε το στοιχειο με calculation density αλλιως βρισκουμε τα δυο μεσεα μεσω της getmin περνουμε τα στοιχεια μεχρι να φτασει στην μεση τα βαζουμε σε δυο μεταβλητες και τα συκρινουμε.

Και επιστέφει το συγκεκριμένο στοιχειού.