ΕΡΓΑΣΙΑ 3

Δομές Δεδομένων

Ονοματεπώνυμο: Κεχριώτη Ελένη

2022-2023

Μέρος Α

Στη κλάση Rectangle:

contains (Point p):
Επιστρέφουμε μια τιμή boolean, αληθής ή ψευδής, ανάλογα με το αν το σημείο p που δίνεται ανήκει στο rectangle. Ο έλεγχος αυτός είναι αρκετά απλός, καθώς το μόνο που χρειάζεται είναι να ελέγξουμε αν το σημείο p με συντεταγμένες (x, y) $\in [x_{min}, x_{max}]$ x $[y_{min},y_{max}]$ Δηλαδή αν x $\in [x_{min}, x_{max}]$ και αν y $\in [y_{min},y_{max}]$. Επομένως αν ισχύουν την ίδια στιγμή όλες αυτές οι συνθήκες, τότε το σημείο ανήκει στο παραλληλόγραμμο.

intersects (Rectangle that):
Επιστρέφουμε μια τιμή boolean, αληθής ή ψευδής, ανάλογα με το αν το rectangle που δίνεται (that) τέμνεται με το rectangle. Μας είναι πιο εύκολο να ελέγξουμε αν το δοσμένο παραλληλόγραμμο δεν τέμνεται με το rectangle. Όταν δύο παραλληλόγραμμα τέμνονται τότε τουλάχιστον 2 από τις συντεταγμένες του ενός θα ανήκουν στο αντίστοιχο διάστημα συντεταγμένων του άλλου. Για παράδειγμα, δύο παραλληλόγραμμα τέμνονται όταν
$y_{max} \in [y_{min},y_{max}]$ και $x_{min} \in [x_{min}, x_{max}]$ (με μια γρήγορη σχεδίαση δύο rectangle εύκολα διαπιστώνεται το παραπάνω). Επομένως αν τουλάχιστον μία από τις συντεταγμένες δεν ανήκει στο αντίστοιχο της διάστημα τότε τα παραλληλόγραμμα θα είναι ξένα μεταξύ τους. Επειδή προφανώς μπορεί να ισχύουν αυτές οι συνθήκες that.ymin() < ymin και that.ymax() > ymax (αντίστοιχα και για x), αρκεί να ελέγξουμε αν that.ymax() < ymin. Προφανώς αν η μεγαλύτερη συντεταγμένη του ενός είναι μικρότερη από την μικρότερη συντεταγμένη του άλλου τότε θα τέμνονται. Αρκεί να ισχύει αυτή η συνθήκη για τουλάχιστον μια απο τις συντεταγμένες του παραλληλογράμμου (έλεγχος or ||). Άρα αν ισχύει η τελική συνθήκη τότε επιστρέφουμε false, διαφορετικά τα παραλληλόγραμμα τέμνονται και επιστρέφουμε true.

distanceTo (Point p):
Επιστρέφουμε τη μικρότερη απόσταση του σημείου p που δίνεται από το παραλληλόγραμμο.
Αρχικά έχουμε δυο ελέγχους.

1) Αν το σημείο βρίσκεται εκτός του παραλληλογράμμου. Σε αυτή τη περίπτωση υπολογίζουμε κανονικά τη μικρότερη απόσταση και την επιστρέφουμε (θα εξηγηθεί παρακάτω).
2) Αν το σημείο βρίσκεται μέσα στο παραλληλόγραμμο ή πάνω στις πλευρές του. Σε αυτή τη περίπτωση θέτουμε ως min απόσταση το 0 και το επιστρέφουμε.

Για την πρώτη περίπτωση έχουμε αρκετούς ελέγχους και υπολογισμούς για να βρούμε τη μικρότερη απόσταση. Αρχικά, βρίσκουμε και ελέγχουμε την απόσταση από τις γωνίες του παραλληλογράμμου. Για τον υπολογισμό αυτό χρησιμοποιούμε τη μέθοδο distanceTo της κλάσης Point στις γωνίες με παράμετρο το σημείο p. Αυτό γίνεται για τις περιπτώσεις που το σημείο βρίσκεται πάνω/κάτω δεξιά/αριστερά και η μικρότερη κοντινή απόσταση είναι η διαγώνια. Μετά από τις πρώτες ifs ακολουθούν έλεγχοι για τις περιπτώσεις που το σημείο βρίσκεται ακριβώς δίπλα από το παραλληλόγραμμο, οπότε η μικρότερη κοντινή απόσταση είναι η κάθετη απόσταση[1] από το παραλληλόγραμμο. Επομένως, ελέγχουμε αν η κάθετη απόσταση του σημείο από κάθε πλευρά είναι μικρότερη του minDis και αν αυτή η απόσταση είναι θετική, αφού δεν

[1] κάθετη απόσταση: είναι απλά το αποτέλεσμα της αφαίρεσης της συντεταγμένης του σημείου με την αντίστοιχη συντεταγμένη του παραλληλογράμμου από την κατάλληλη πλευρά. Για παράδειγμα από δεξιά η απόσταση είναι p.y() - ymax.

μπορούμε να έχουμε αρνητική απόσταση. Μετά από τον τελευταίο έλεγχο έχει σίγουρα βρεθεί και τεθεί η ελάχιστη απόσταση ως minDis, και επιστρέφεται.

Μέρος Β

rangeSearch (Rectangle rect):
Στη μέθοδο χρησιμοποιούμε την rangeSearch_helper ως βοηθητική συνάρτηση για να βρούμε αναδρομικά τα σημεία που περιέχονται στο rectangle. Αρχικά, δημιουργούμε μια κενή λίστα αντικειμένων Point. Κάθε φορά που καλείται η μέθοδος rangeSearch "καθαρίζουμε" τη λίστα, δηλαδή διαγράφουμε τα σημεία που περιείχε η λίστα από την προηγούμενη αναζήτηση. Αν η ρίζα του δέντρου είναι null τότε επιστρέφουμε null, αφού δεν υπάρχουν σημεία για να ελέγξουμε. Διαφορετικά, καλείται η βοηθητική μέθοδος rangeSearch_helper, με παραμέτρους την ρίζα και το rectangle που έδωσε ο χρήστης.

rangeSearch_helper (TreeNode t, Rectangle rect):
Στη μέθοδο rangeSearch_helper αρχικά ελέγχουμε αν το δοσμένο rectangle τέμνεται με το rectangle του εκάστοτε κόμβου που ελέγχεται. Αν τέμνονται τότε υπάρχουν σημεία να προστεθούν στη λίστα και συνεχίζουμε την αναζήτηση. Διαφορετικά, δεν υπάρχει λόγος για περαιτέρω αναζήτηση και σταματάμε εκεί. Στη περίπτωση που υπάρχουν σημεία, τότε ελέγχουμε αν στο rectangle ανήκει το σημείο του κόμβου, και αν ισχύει το προσθέτουμε στη λίστα. Ύστερα, ελέγχουμε αν ο αριστερός κόμβος υπάρχει και καλούμε ξανά τη μέθοδο για τον αριστερό κόμβο και κατα συνέπεια το αριστερό υποδέντρο. Αντίστοιχα, και για τον δεξί κόμβο. Τέλος, όταν ολοκληρωθεί η αναδρομή, επιστρέφουμε στη κύρια μέθοδο rangeSearch και επιστρέφουμε τη λίστα με τα σημεία.

nearestNeighbor (*Point p*):
Στη μέθοδο χρησιμοποιούμε την nearestneighbour_helper ως βοηθητική συνάρτηση για να εντοπίσουμε τον κοντινότερο "γείτονα" σημείο του σημείου που έδωσε ο χρήστης. Αρχικά, αν η ρίζα του δέντρου είναι κενή, δεν υπάρχει σημείο για να επιστραφεί και επιστρέφουμε null. Διαφορετικά καλούμε την nearestneighbour_helper με ορίσματα την ρίζα και το σημείο p και επιστρέφουμε το point του κόμβου που επιστρέφει η nearestneighbour_helper.

nearestNeighbour_helper(Treenode P,Point p):
Η μέθοδος αυτή είναι αναδρομική. Προφανώς, αν τα παιδιά του κόμβου είναι και τα τα δύο null, τότε έχουμε καταλήξει σε φύλλο και δεν υπάρχει άλλο κόμβος για να ελέγξουμε αν το σημείο του είναι κοντινότερο. Άρα, αυτός ο κόμβος θα έχει το κοντινότερο σημείο στο σημείο του χρήστη και επιστρέφεται. Διαφορετικά, ελέγχουμε ξεχωριστά για τα παιδιά. Αν το δεξί παιδί του κόμβου ειναι null τότε θα πρέπει να συνεχίσουμε το ψάξιμο στο αριστέρο κόμβο και αντίστροφα. Ακόμα, αν ο κόμβος έχει 2 παιδιά μη null, τότε ψάχνουμε και θέτουμε στο x το κοντινότερο γείτονα κόμβο στο δεξί υποδέντρο και στο y το κοντινότερο γείτονα κόμβο στο αριστερό υποδέντρο. Ύστερα, ελέγχουμε ποιος κομβος απο τους x και y είναι κοντινότερος στο σημείο που έδωσε ο χρήστης και θέτουμε ως x τον κοντινότερο. Τέλος, το συγκρίνουμε με τον πατέρα κόμβο(επιστρέφοντας το πιο κοντινό σημείο.