Μαρία – Ιωάννα Μουζουράκη p3200105

ΕΡΓΑΣΊΑ 3

Μέρος Δ – Αναφορά Παράδοσης

Μέρος Α

- **contains:** Για να ανήκει ένα σημείο στο παραλληλόγραμμο ,πρέπει να ισχύουν τα εξής :
Η x-συντεταγμένη του βρίσκεται εντός στο εύρος τιμών των x-συντεταγμένων του παραλληλόγραμου
(p.x() >= this.xmin && p.x() <= this.xmax)
και η y-συντεταγμένη του βρίσκεται εντός στο εύρος τιμών των y-συντεταγμένων του παραλληλόγραμου
(p.y() >= this.ymin && p.y() <= this.ymax).

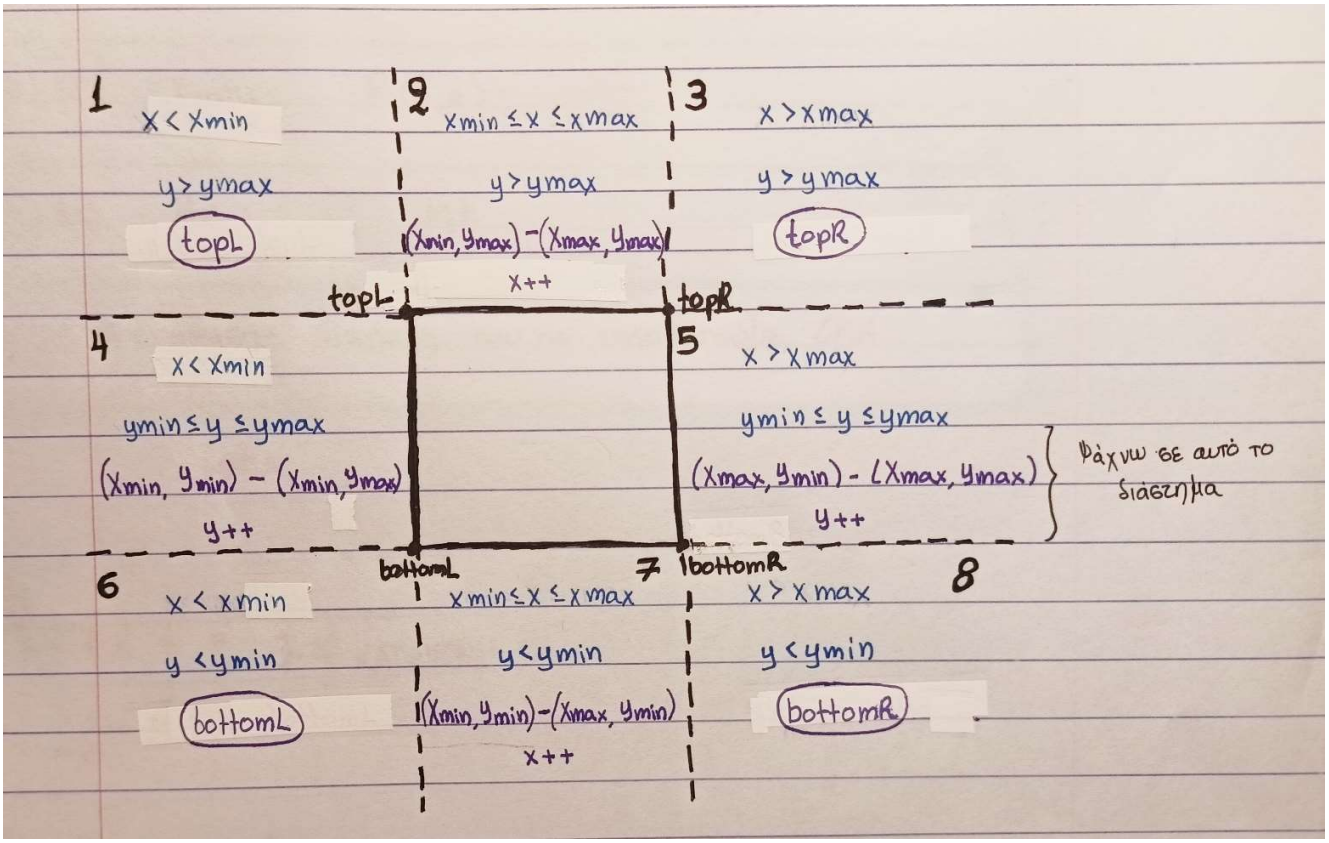
- **intersects:** Παρατηρούμε ότι αν τα παραλληλόγραμμα δεν έχουν κοινά σημεία τότε είτε το ένα βρίσκεται πιο πάνω από το άλλο, είτε το ένα βρίσκεται αριστερά του άλλου.

Αν ισχύει this.ymax < that.ymin τότε το παραλληλόγραμμο που δόθηκε ως παράμετρος βρίσκεται πάνω από το δικό μας , και το αντίστροφο συμβαίνει αν ισχύει this.ymin > that.ymax. Σε αυτές τις περιπτώσεις επιστρέφουμε false.

Αν ισχύει this.xmax < that.xmin τότε το παραλληλόγραμμο μας βρίσκεται αριστερά από αυτό που που δόθηκε ως παράμετρος, και το αντίστροφο συμβαίνει αν ισχύει this.xmin > that.xmax. Σε αυτές τις περιπτώσεις επιστρέφουμε false.

Αν δεν ισχύει τίποτα από τα παραπάνω , τα παραλληλόγραμμα έχουν κοινά σημεία, οπότε επιστρέφουμε true.

- **distanceTo:** Για να βρούμε σε ποιο σημείο του παραλληλογράμμου βρίσκεται πιο κοντά το σημείο που δίνεται ως παράμετρος εξετάζουμε τις εξής περιπτώσεις:



1η περίπτωση : επιστρέφει την απόσταση μεταξύ του σημείου και του σημείου topL(xmin ,ymax)

2η περίπτωση : βρίσκει το σημείο με y=ymax και xmin<=x<=xmax και με την μικρότερη απόσταση από το σημείο παράμετρος και επιστρέφει την απόσταση

3η περίπτωση : επιστρέφει την απόσταση μεταξύ του σημείου και του σημείου topR(xmax ,ymax)

4η περίπτωση : βρίσκει το σημείο με x=xmin και ymin<=y<=ymax και με την μικρότερη απόσταση από το σημείο παράμετρος και επιστρέφει την απόσταση

5η περίπτωση : βρίσκει το σημείο με x=xmax και ymin<=y<=ymax και με την μικρότερη απόσταση από το σημείο παράμετρος και επιστρέφει την απόσταση

6η περίπτωση : επιστρέφει την απόσταση μεταξύ του σημείου και του σημείου bottomL(xmin ,ymin)

7η περίπτωση : βρίσκει το σημείο με y=ymin και xmin<=x<=xmax και με την μικρότερη απόσταση από το σημείο παράμετρος και επιστρέφει την απόσταση

8η περίπτωση : επιστρέφει την απόσταση μεταξύ του σημείου και του σημείου bottomR(xmax ,ymin)

Μέρος Β

- **nearestNeighbor:** Η nearestNeighbor καλεί την βοηθητική μέθοδο nearestNeighbor _RecursiveFunction με αρχικές παραμέτρους

  Rectangle(0, 100, 0, 100), και cord="x" (ώστε να ελέγχουμε πρώτα τις χ συντεταγμένες), για να επιστρέψει το σημείο του δέντρου που βρίσκεται πιο κοντά στο σημείο που δίνετε ως παράμετρος (σημείο αναζήτησης).

- **nearestNeighbor_RecursiveFunction:** Καλείται αναδρομικά .Αν έχουμε φτάσει σε άδειο δέντρο , επιστρέφει null .

  Αν το head του δέντρου στο οποίο βρισκόμαστε, τότε προσθέτουμε τον κόμβο στην λίστα , βρίσκεται πιο κοντά στο σημείο αναζήτησης, από ότι το πλησιέστερο έως τώρα σημείο , τότε θέτουμε ως πλησιέστερο σημείο το head.

  Εξετάζουμε εάν η απόσταση του πλησιέστερου σημείου με το σημείο αναζήτησης είναι μεγαλύτερη από την απόσταση του παραλληλόγραμμου-κόμβου με το σημείο αναζήτησης.

  Αν ισχύει , και επίσης είναι η σειρά να ελέγξουμε τις χ συντεταγμένες , τότε η nearestNeighbor_RecursiveFunction καλείται

από το LeftSubTree , τώρα όμως έχοντας για παραλληλόγραμμο κόμβου το Rectangle(rectangle.xmin(), head.getData().x(), rectangle.ymin(), rectangle.ymax())

και θέτοντας cord==”y” ώστε να ελεγχθούν οι y συντεταγμένες την επόμενη φορά,

και από το RightSubTree, τώρα όμως έχοντας για παραλληλόγραμμο κόμβου το Rectangle(head.getData().x(), rectangle.xmax(), rectangle.ymin(), rectangle.ymax()) και θέτοντας cord==”y”.

Αν είναι η σειρά να ελέγξουμε τις y συντεταγμένες , τότε η nearestNeighbor_RecursiveFunction καλείται

από το LeftSubTree , τώρα όμως έχοντας για παραλληλόγραμμο κόμβου το Rectangle(rectangle.xmin(), rectangle.xmax(), rectangle.ymin(), head.getData().y()) και θέτοντας cord==”x” ώστε να ελεγχθούν οι χ συντεταγμένες την επόμενη φορά,

και από το RightSubTree, τώρα όμως έχοντας για παραλληλόγραμμο κόμβου το Rectangle(rectangle.xmin(), rectangle.xmax(), head.getData().y(), rectangle.ymax()) και θέτοντας cord==”x”.

- **rangeSearch:** Η rangeSearch καλεί την βοηθητική μέθοδο rangeSearch_RecursiveFunction με αρχικές παραμέτρους

Rectangle(0, 100, 0, 100), και cord="x" (ώστε να ελέγχουμε πρώτα τις χ συντεταγμένες), για να επιστρέψει μία λίστα με τα σημεία του δέντρου που περιέχει το παραλληλόγραμμο.

- **rangeSearch_RecursiveFunction:** Καλείται αναδρομικά .Αν έχουμε φτάσει σε άδειο δέντρο , επιστρέφει την λίστα .

Αν το παραλληλόγραμμο περιέχει το head του δέντρου στο οποίο βρισκόμαστε , τότε προσθέτουμε τον κόμβο στην λίστα.

Εξετάζουμε αν το παραλληλόγραμμο το οποίο ελέγχουμε έχει κοινά σημεία με το παραλληλόγραμμο-κόμβου (περιγράφεται στην εκφώνηση πως κάθε κόμβος σε ένα 2d-tree αντιστοιχεί σε ένα παραλληλόγραμμο).

Αν ισχύει , και επίσης είναι η σειρά να ελέγξουμε τις χ συντεταγμένες , τότε η rangeSearch_RecursiveFunction καλείται

από το LeftSubTree , τώρα όμως έχοντας για παραλληλόγραμμο κόμβου το Rectangle(pointRectangle.xmin(), head.getData().x(), pointRectangle.ymin(),pointRectangle.ymax()) και θέτοντας cord==”y” ώστε να ελεγχθούν οι y συντεταγμένες την επόμενη φορά,

και από το RightSubTree, τώρα όμως έχοντας για παραλληλόγραμμο κόμβου το Rectangle(head.getData().x(), pointRectangle.xmax(), pointRectangle.ymin(), pointRectangle.ymax()) και θέτοντας cord==”y”.

Αν είναι η σειρά να ελέγξουμε τις y συντεταγμένες , τότε η rangeSearch_RecursiveFunction καλείται

από το LeftSubTree , τώρα όμως έχοντας για παραλληλόγραμμο κόμβου το Rectangle(pointRectangle.xmin(), pointRectangle.xmax(), pointRectangle.ymin(),head.getData().y()) και θέτοντας cord==”x” ώστε να ελεγχθούν οι χ συντεταγμένες την επόμενη φορά,

και από το RightSubTree, τώρα όμως έχοντας για παραλληλόγραμμο κόμβου το Rectangle(pointRectangle.xmin(), pointRectangle.xmax(), head.getData().y(),pointRectangle.ymax()) και θέτοντας cord==”x”.