Πανεπιστήμιο Ιωαννίνων

ΔΙΑΧΕΙΡΙΣΗ ΣΥΝΘΕΤΩΝ ΔΕΔΟΜΕΝΩΝ

ΕΡΓΑΣΙΑ 2η - Χωρικά Δεδομένα

ΝΤΟΝΤΗΣ ΒΑΣΙΛΕΙΟΣ

ΑΜ: 3300

ΜΕΡΟΣ 1ο

Διάβασμα αρχείου και διαχείριση των δεδομένων

Αρχικά, ξεκινάμε διαβάζοντας το αρχείο tiger_roads.csv και καθώς το διαβάζουμε παίρνουμε τα απαραίτητα δεδομένα για να γεμίσουμε τον πίνακα με όλα τα αντικείμενα. Αρχικοποιούμε έναν counter ίσο με 0 για να μην λάβουμε υπόψιν μας την 1η γραμμή του αρχείου επειδή περιέχει το μήκος του αρχείου το οποίο δεν μας ενδιαφέρει. Αρχικοποιούμε επίσης τους πίνακες AllX και AllY, στους οποίους θα προσθέσουμε όλα τα x και y των linestrings, και έτσι θα έχουμε εύκολη πρόσβαση στα min και max τους. Επίσης να σημειωθεί ότι μετά από κάθε διάβασμα αντικειμένου ο counter αθροίζετε με το 1, καθώς σηματοδοτεί και το Identifier του κάθε αντικειμένου.

Στην συνέχεια, για κάθε γραμμή εκτός της 1ης, δημιουργούμε τους παρακάτω πίνακες:

- ***recordArray*** : Πίνακας που του προστίθενται ο counter (αριθμός Identifier) και οι υπόλοιποι παρακάτω πίνακες και καταλήγει στο επιθυμητό *[ID, [[MBRminX MBRminY],[MBRmaxX MBRmaxY]],[[x1,y1],[x2,y2]...]]*.
- ***MBRarray*** : Πίνακας που κρατάει τα ελάχιστα X και Y σε έναν πίνακα εντός του και τα μέγιστα X και Y σε έναν άλλον πίνακα. Δηλαδή την κάτω αριστερά γωνία του MBR του αντικειμένου και την πάνω δεξιά. Κρατάει αυτό το κομμάτι -> *[MBRminX MBRminY],[MBRmaxX MBRmaxY]*
- ***Xarray*** : Βοηθητικός πίνακας που κρατάει όλα τα X του αντικειμένου, για να προστεθούν τα ελάχιστα και μέγιστα του στους MBRarray και AllX πίνακες.
- ***Yarray*** : Βοηθητικός πίνακας που κρατάει όλα τα Y του αντικειμένου, για να προστεθούν τα ελάχιστα και μέγιστα του στους MBRarray και AllY πίνακες.
- ***linestringArray*** : Πίνακας που κρατάει όλες τις δυάδες X1,Y1 X2,Y2…XN, YN, των linestrings.

Έπειτα, διαβάζοντας τα linestrings της γραμμής γεμίζουμε τους πίνακες και προσθέτουμε το τελικό recordArray (με το επιθυμητό format) στον πίνακα **records**. Ο πίνακας **records**, κρατάει όλα τα δεδομένα για όλα τα αντικείμενα σε μορφή *[ID, [[MBRminX MBRminY],[MBRmaxX MBRmaxY]],[[x1,y1],[x2,y2]...]]*, για κάθε αντικείμενο.

Υπολογισμός σχάρας

Για να υπολογίσουμε την σχάρα αρχικά δημιουργούμε τον πίνακα **gridArray**, στον οποίο αποθηκεύουμε για κάθε κελί τα ελάχιστα και μέγιστα X και Y, έτσι ώστε να συγκρίνουμε με αυτά κάθε αντικείμενο κατά τον έλεγχο τοποθεσίας αντικειμένων. Ο πίνακας gridArray είναι φτιαγμένος έτσι ώστε με την εκτέλεση gridArray[5][7] να παίρνουμε τα όρια συντεταγμένων του κελιού με x = 5 και y = 7. Στην συνέχεια, αποθηκεύουμε τα όρια του X και του Y καθώς είναι απαραίτητα για τον υπολογισμό των ορίων των κελιών. Αρχικοποιούμε τον πίνακα **resultsList**, ο οποίος έχει ίδιες διαστάσεις με τον gridArray, σε κάθε κελί αρχικοποιείται με 0 και μετά τον έλεγχο τοποθεσίας αντικειμένων θα κρατάει τον αριθμό των αντικειμένων που βρίσκονται σε κάθε κελί. Ο πίνακας **gridIntersectsWith**, έχει και αυτός ίδιες διαστάσεις με τον gridArray, αλλά αντί να κρατάει τον αριθμό των αντικειμένων που βρίσκονται σε κάθε κελί, κρατάει τα Identifiers τους. Εντός της for για τον υπολογισμό της σχάρας τοποθετούμε σε όλες τις θέσεις του resultsList 0, ενώ σε όλες τις θέσεις του gridIntersectsWith έναν κενό πίνακα. Εντός της for για τον υπολογισμό της σχάρας, δημιουργούμε μερικούς βοηθητικούς πίνακες και σε κάθε επανάληψη υπολογίζουμε τα όρια των κελιών με την βοήθεια των τύπων :

- ***minX + v * (maxX - minX) / 10*** , με **0 < v < 10**
- ***minY + v * (maxY - minY) / 10*** , με **0 < v < 10**

και έτσι παίρνουμε τα απαραίτητα δεδομένα για να γεμίσουμε τον gridArray με τις επιθυμητές συντεταγμένες των ορίων των κελιών, αλλά και τους πίνακες resultsList και gridIntersectsWith που μας είναι απαραίτητοι για την συνέχεια. Αφού δημιουργήσουμε τα αρχεία *grid.grd* και *grid.dir*, προχωράμε στον έλεγχο τοποθεσίας αντικειμένων.

Έλεγχος τοποθεσίας αντικειμένων

Διατρέχοντας όλα τα αντικείμενο από τον πίνακα records, και για κάθε αντικείμενο διατρέχοντας όλα τα κελιά, εκτελούμε την παρακάτω συνθήκη if, το αποτέλεσμα της οποίας αν είναι θετικό, θα ενημερωθούν οι πίνακες resultsList και gridIntersectsWith που αναλύσαμε προηγουμένως. Να σημειωθεί ότι στην φωτογραφία το **i** είναι το αντικείμενο που εξετάζουμε από τον records πίνακα και έχει την μορφή *[ID, [[MBRminX MBRminY],[MBRmaxX MBRmaxY]],[[x1,y1],[x2,y2]...]]*. Επίσης ο πίνακας gridArray περιέχει τις συντεταγμένες τις κάτω αριστερά γωνίας και πάνω δεξιά γωνίας του εκάστοτε κελιού.

```
if (gridArray[x][y][0] <= i[1][0][0] <= gridArray[x][y][2] and  # 1st - case
    gridArray[x][y][1] <= i[1][0][1] <= gridArray[x][y][3]) or \
        (gridArray[x][y][2] >= i[1][1][0] >= gridArray[x][y][0] and  # 2nd - case
         gridArray[x][y][3] >= i[1][1][1] >= gridArray[x][y][1]) or \
        (gridArray[x][y][0] <= i[1][0][0] <= gridArray[x][y][2] and  # 3rd - case
         gridArray[x][y][1] <= i[1][1][1] <= gridArray[x][y][3]) or \
        (gridArray[x][y][2] >= i[1][1][0] >= gridArray[x][y][0] and  # 4th - case
         gridArray[x][y][3] >= i[1][0][1] >= gridArray[x][y][1]) or \
        (i[1][0][0] <= gridArray[x][y][0] and i[1][0][1] <= gridArray[x][y][1] and  # 5th - case
         i[1][1][0] >= gridArray[x][y][2] and i[1][1][1] >= gridArray[x][y][3]) or \
        (i[1][0][0] <= gridArray[x][y][0] and i[1][1][0] >= gridArray[x][y][2] and  # 6th - case
         (gridArray[x][y][1] <= i[1][0][1] <= gridArray[x][y][3]
          or gridArray[x][y][1] <= i[1][0][1] <= gridArray[x][y][3]
          or gridArray[x][y][1] <= i[1][0][1] <= gridArray[x][y][3]
          or gridArray[x][y][1] <= i[1][1][1] <= gridArray[x][y][3]
          or gridArray[x][y][1] <= i[1][1][1] <= gridArray[x][y][3]
          or gridArray[x][y][1] <= i[1][1][1] <= gridArray[x][y][3])) or \
        (i[1][0][0] <= gridArray[x][y][0] and i[1][0][1] <= gridArray[x][y][1] and  # 7th - case
         i[1][1][0] >= gridArray[x][y][2] and i[1][1][1] >= gridArray[x][y][3]) or \
        (i[1][0][1] <= gridArray[x][y][1] and i[1][1][1] >= gridArray[x][y][3] and  # 8th - case
         (gridArray[x][y][0] <= i[1][0][0] <= gridArray[x][y][2]
          or gridArray[x][y][0] <= i[1][0][0] <= gridArray[x][y][2]
          or gridArray[x][y][0] <= i[1][0][0] <= gridArray[x][y][2]
          or gridArray[x][y][0] <= i[1][1][0] <= gridArray[x][y][2]
          or gridArray[x][y][0] <= i[1][1][0] <= gridArray[x][y][2]
          or gridArray[x][y][0] <= i[1][1][0] <= gridArray[x][y][2])):
    resultsList[x][y] += 1
    gridIntersectsWith[x][y].append(i[0])
```

Συνθήκη ελέγχου τοποθεσίας αντικειμένων

Παρακάτω θα αναλύσουμε όλες τις περιπτώσεις που εξετάζει αυτή η συνθήκη, αφού πρώτα δείξουμε τι σημαίνει κάθε χρήση του i και του gridArray.

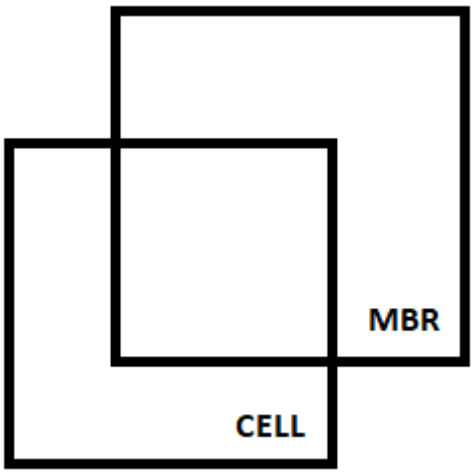
- ***gridArray[x][y][0]*** : minX του κελιού - (x, y)
- ***gridArray[x][y][1]*** : minY του κελιού - (x, y)
- ***gridArray[x][y][2]*** : maxX του κελιού - (x, y)
- ***gridArray[x][y][3]*** : maxY του κελιού - (x, y)

- ***i[1][0][0]*** : minX του MBR του αντικειμένου i
- ***i[1][0][1]*** : minY του MBR του αντικειμένου i
- ***i[1][1][0]*** : maxX του MBR του αντικειμένου i
- ***i[1][1][1]*** : maxY του MBR του αντικειμένου i

Περιπτώσεις που ελέγχονται

1η περίπτωση: Η κάτω αριστερή γωνία του αντικειμένου βρίσκεται εντός του κελιού.

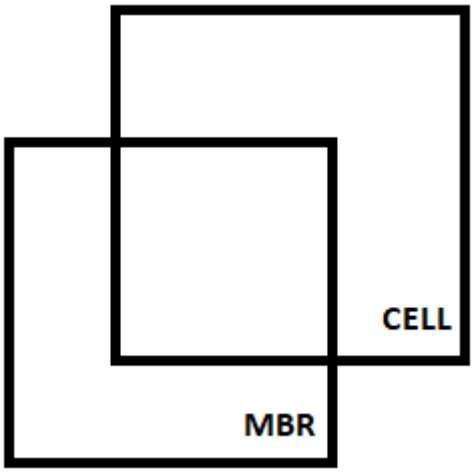
(gridArray[x][y][0] <= minX του MBR <= gridArray[x][y][2] and gridArray[x][y][1] <= minY του MBR <= gridArray[x][y][3])



Παράδειγμα 1ης περίπτωσης

2η περίπτωση: Η πάνω δεξιά γωνία του αντικειμένου βρίσκεται εντός του κελιού

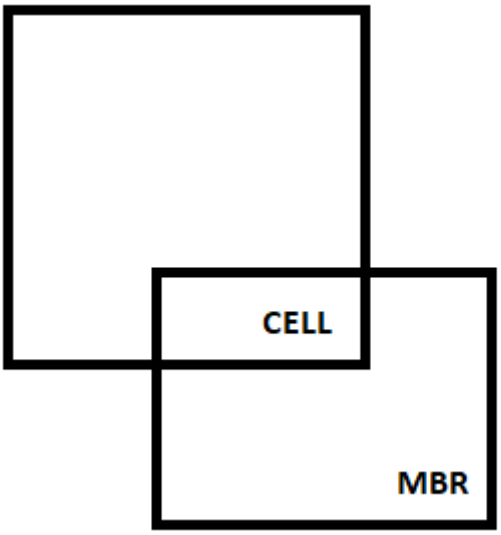
(gridArray[x][y][2] >= maxX του MBR >= gridArray[x][y][0] and gridArray[x][y][3] >= maxY του MBR >= gridArray[x][y][1])



Παράδειγμα 2ης περίπτωσης

3η περίπτωση: Η πάνω αριστερή γωνία του αντικειμένου βρίσκεται εντός του κελιού

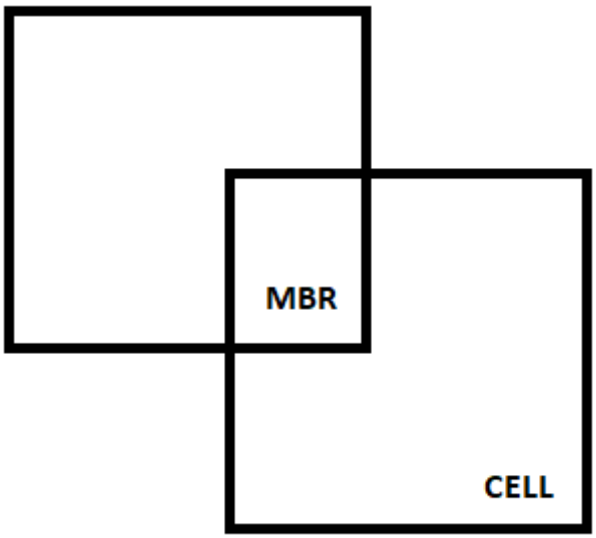
(gridArray[x][y][0] <= minX του MBR <= gridArray[x][y][2] and gridArray[x][y][1] <= maxY του MBR <= gridArray[x][y][3])



Παράδειγμα 3ης περίπτωσης

4η περίπτωση: Η κάτω δεξιά γωνία του αντικειμένου βρίσκεται εντός του κελιού

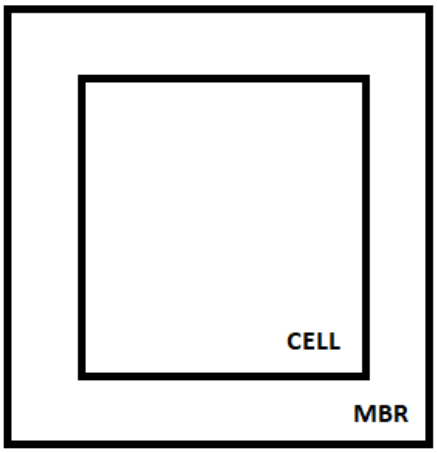
(gridArray[x][y][2] >= maxX του MBR >= gridArray[x][y][0] and gridArray[x][y][3] >= minY του MBR >= gridArray[x][y][1])



Παράδειγμα 4ης περίπτωσης

5η περίπτωση: Το MBR του αντικειμένου περιέχει όλο το κελί

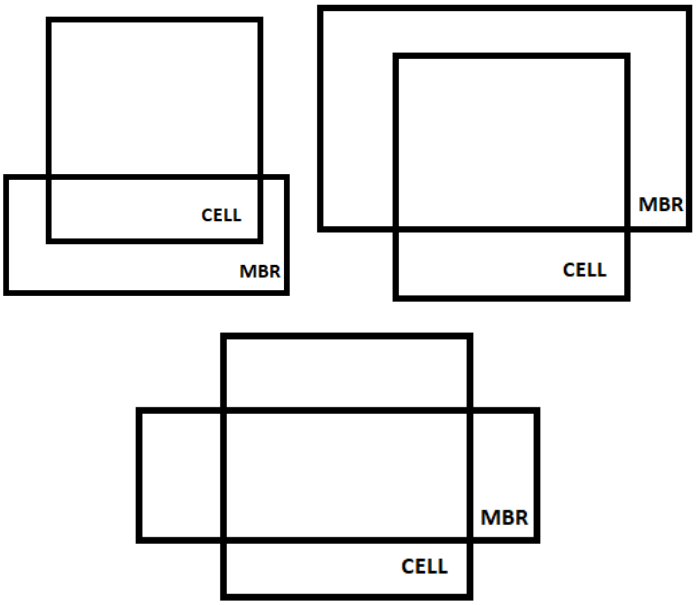
(minX του MBR <= gridArray[x][y][0] and
minY του MBR <= gridArray[x][y][1] and
maxX του MBR >= gridArray[x][y][2] and
maxY του MBR >= gridArray[x][y][3])



Παράδειγμα 5ης περίπτωσης

6η περίπτωση: Καμία γωνία του MBR του αντικειμένου δεν βρίσκεται εντός του κελιού, όμως οι γραμμές τους τέμνονται παράλληλα με τον x άξονα.

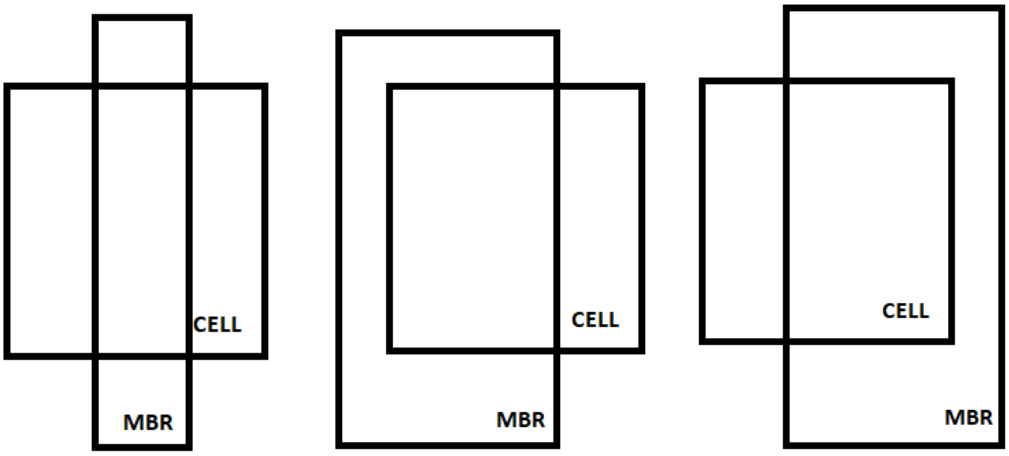
(minX του MBR <= gridArray[x][y][0] and maxX του MBR >= gridArray[x][y][2] and
(gridArray[x][y][1] <= minY του MBR <= gridArray[x][y][3] or
gridArray[x][y][1] <= maxY του MBR <= gridArray[x][y][3]))



Παραδείγματα 6ης περίπτωσης

7η περίπτωση: Καμία γωνία του MBR του αντικειμένου δεν βρίσκεται εντός του κελιού, όμως οι γραμμές τους τέμνονται παράλληλα με τον y άξονα.

(minY του MBR <= gridArray[x][y][1] and maxY του MBR >= gridArray[x][y][3] and (gridArray[x][y][0] <= minX του MBR <= gridArray[x][y][2] or gridArray[x][y][0] <= maxX του MBR <= gridArray[x][y][2]))



Παραδείγματα 7ης περίπτωσης

Έχοντας περάσει ο κώδικας όλα τα αντικείμενα από τον έλεγχο τοποθεσίας αντικειμένων, προχωράμε στο τελευταίο κομμάτι του 1ου μέρους.

Εγγραφή αρχείων grid.grd και grid.dir

Έχοντας ενημερωμένους τους πίνακες resultsList και gridIntersectsWith, διατρέχοντας τα δεδομένα τους γράφουμε τα αποτελέσματα στα αρχεία. Οι πίνακες που χρησιμοποιούνται για τα αρχεία:

- **resultsList -> grid.dir αρχείο** (έχει αναλυθεί παραπάνω)
- **gridIntersectsWith + records -> grid.grd αρχείο** (Διατρέχοντας με την σειρά τα κελιά, ξεκινώντας από το (0,0), παίρνουμε από το gridIntersectsWith τα Identifiers που αναλογούν στο εκάστοτε κελί. Έχοντας το Identifier, πηγαίνουμε στην σωστή θέση του αντικειμένου στον πίνακα records, και γράφουμε στο αρχείο τα δεδομένα του)

Το πρόγραμμα τελειώνει τυπώνοντας την επιτυχή εγγραφή και των δύο αρχείων.

ΜΕΡΟΣ 2ο

Διάβασμα αρχείου grid.dir

Ξεκινώντας το 2ο μέρος, ορίζουμε τον πίνακα resultsList, ο οποίος κρατάει τον αριθμό των αντικειμένων που βρίσκονται σε κάθε κελί, και διαβάζοντας το αρχείο grid.dir, γεμίζουμε τον resultsList με τα σωστά δεδομένα. Διαβάζοντας την πρώτη γραμμή του αρχείου, αφού splitάρουμε το αλφαριθμητικό της γραμμής, παίρνουμε τα min και max X και Y, αρχικοποιούμε :

- έναν counter για να γνωρίζουμε πότε αλλάζει το X
- τον πίνακα resultY, ο οποίος είναι βοηθητικός για το σωστό γέμισμα του resultsList (κρατάει το πλήθος των intersections για τα κελιά με ίδιο X)
- το sumOfConnections, που κρατάει το πλήθος των intersections, για να αρχικοποιήσουμε σε αυτό το μήκος τον πίνακα όλων των αντικειμένων records.

Στην συνέχεια, διαβάζοντας κάθε γραμμή που έχει την μορφή (x, y) κελιού : πλήθος intersections , ανεβάζουμε τον counter κατά 1, και αφού τσεκάρουμε αν η γραμμή δεν σημαίνει το τέλος του αρχείου, προσθέτουμε το πλήθος των intersections στο sumOfConnections και ελέγχουμε αν ο αριθμός counter είναι ίσος με το 11. Αν ο αριθμός counter είναι 11, σημαίνει ότι ανέβηκε το x, για παράδειγμα από το (0, 9) πήγαμε στο (1, 0), προστίθεται το resultY με το πλήθος για κάθε κελί στον resultsList και στην συνέχεια εκκαθαρίζεται και ενημερώνεται με το νέο πλήθος κελιού της επόμενης γραμμής. Αν δεν είναι 11 ο counter, απλά προσθέτουμε το πλήθος στον βοηθητικό resultY.

Έπειτα αρχικοποιούμε τον πίνακα records με None τιμές παντού, τον οποίο θα γεμίσουμε στην συνέχεια του προγράμματος με την βοήθεια του αρχείου grid.grd.

Υπολογισμός σχάρας

Ο υπολογισμός της σχάρας είναι ίδιος με το μέρος 1ο , οπότε δεν θα αναλυθεί.

Διάβασμα αρχείου grid.grd

Ξεκινώντας αρχικοποιούμε το line σε True, γιατί υπήρχε πρόβλημα αλλιώς στο διάβασμα του αρχείου. Διαβάζοντας την κάθε γραμμή, αρχικοποιούμε το found σε False, αρχικοποιούμε τον πίνακα recordList, στον οποίο θα αποθηκεύσουμε τα δεδομένα έτσι ώστε να τα μεταφέρουμε μετά στον γενικό records. Άρα θέλουμε να το φτάσουμε στην μορφή : *[ID, [[MBRminX, MBRminY],[MBRmaxX, MBRmaxY]],[[x1,y1],[x2,y2]...]]*. Αρχικοποιούμε τον πίνακα minMaxList, ο οποίος θα κρατάει τα

στοιχεία για το MBR του κάθε αντικειμένου και θα προστίθεται μετά στον recordList, θέλουμε να τον φτάσουμε στην μορφή : *[[MBRminX, MBRminY], [MBRmaxX, MBRmaxY]]* . Αφού προσθέσουμε το Identifier του αντικειμένου που διαβάσαμε και τα στοιχεία του MBR του, εκτελώντας τον ακόλουθο κώδικα, διαβάζουμε κάθε φορά μια δυάδα x1,y1 με τις συντεταγμένες ενός αντικειμένου. Ο πίνακας linestring θέλουμε να είναι της μορφής : *[x1, y1]* . Ο πίνακας linestringList θέλουμε να είναι της μορφής : *[[x1, y1], [x2, y2]...]]* .

```
for i in range(5, len(lineSplitted), 2):
    linestring = [float(lineSplitted[i]), float(lineSplitted[i + 1])]
    linestringsList.append(linestring)
recordList.append(linestringsList)
records[recordList[0] - 1] = recordList
```

Προσθέτουμε όλα τα δεδομένα στον recordList, και αντικαθιστούμε την θέση του Identifier - 1 στον πίνακα records με αυτόν. Τέλος, διατρέχοντας τα x και y ενός 10x10 πίνακα, ελέγχουμε σε ποιο κελί αντιστοιχεί το αντικείμενο που διαβάσαμε και ενημερώνουμε με αυτό το αντικείμενο τον πίνακα gridIntersectsWith που δημιουργήθηκε κατά τον υπολογισμό της σχάρας. Αν βρούμε σε ποιο κελί ανήκει το αντικείμενο που διαβάσαμε ορίζουμε το found σε True, για να σταματήσει εκεί η εκτέλεση της for, και να μην συνεχίσουμε σε κελιά που δεν χρειάζεται να διαβάσουμε.

Παρακάτω ο κώδικας :

```
for x in range(len(gridIntersectsWith)):
    for y in range(len(gridIntersectsWith[x])):
        if resultsList[x][y] > 0:
            resultsList[x][y] -= 1
            gridIntersectsWith[x][y].append(recordList[0])
            found = True
            break
    if found:
        break
```

Ουσιαστικά, είναι σαν να κοιτάμε στο αρχείο grid.dir το (0, 6) αρχικά που έχει τα πρώτα 4 αντικείμενα, όταν διαβάσουμε το 1^{ο} αντικείμενο, θα κάνουμε το 4 -> 3. Έτσι γνωρίζουμε ότι τα επόμενα 3 αντικείμενα ανήκουν εκεί, όταν μηδενίσει το κελί (0, 6) προχωράμε να τοποθετούμε τα αντικείμενα στο επόμενο αντίστοιχο κελί.

Έπειτα, αφαιρούμε όλες τις None τιμές από τον πίνακα records. Έτσι είναι ίδιος με αυτόν του 1^{ου} μέρους.

Διάβασμα αρχείου queries.txt

Δημιουργούμε τον πίνακα query και αφού πάρουμε τις συντεταγμένες από το αρχείο σε float, τις προσθέτουμε στον πίνακα **queries**.

- **queries** : πίνακας που κρατάει τις συντεταγμένες του παραθύρου που θα καλεστούμε να ελέγξουμε, με την μορφή [Xmin, Xmax, Ymin, Ymax]. Η θέση 0 αφορά το query με αριθμό 1.

Αφού διαβάσουμε τα queries, δημιουργούμε τον πίνακα **queriesCellsXY**.

- **queriesCellsXY** : πίνακας με μήκος ίσο με τον αριθμό τον queries, στην θέση του κάθε query όμως, έχει έναν πίνακα αρχικά κενό. Στην συνέχεια θα τον γεμίσουμε με τα κελιά που αντιστοιχούνε στο κάθε window από το query. Για παράδειγμα, για το window του πρώτου query πρέπει να ελεγχθεί το κελί (4,2), άρα στην θέση queriesCellsXY[0], υπάρχει το [4,2].

Υπολογισμός αποτελεσμάτων των queries

Αρχικά, για να γεμίσουμε τον πίνακα queriesCellsXY, τρέχουμε μια for για κάθε query, και εντός της 2 for για τα x και y της σχάρας. Μέσα της ελέγχουμε τις περιπτώσεις που είχαμε χρησιμοποιήσει για τον έλεγχο τοποθεσίας αντικειμένων στο 1^{ο} μέρος. Απλά αντί να ελέγχουμε ένα Cell με το MBR ενός αντικειμένου, ελέγχουμε ένα Cell με το παράθυρο του κάθε query. Έτσι συμπληρώνουμε τον πίνακα queriesCellsXY με τα κελιά που αντιστοιχούν σε κάθε query.

Αφού ενημερώσουμε τον πίνακα queriesCellsXY, προχωράμε στον έλεγχο για τα αποτελέσματα των queries.

Κάνουμε μια for για κάθε query και αρχικοποιούμε τις παρακάτω μεταβλητές:

- **uniqueResults** : πίνακας ο οποίος θα μας βοηθήσει να μην έχουμε διπλότυπα αντικειμένων στα αποτελέσματα του κάθε query.
- **results** : μετρητής αποτελεσμάτων του κάθε query.
- **resultsMBRs** : πίνακας που κρατάει τα Identifiers των αντικειμένων που πληρούν τις προϋποθέσεις.
- **coordinatesCounter** : βοηθητική μεταβλητή που κρατάει την θέση της δυάδας (x, y) στον πίνακα queriesCellsXY, του υπό εξέταση κελιού για το κάθε query.

```
for i in range(len(queries)):
    uniqueResults = []
    results = 0
    resultsMBRs = []
    coordinatesCounter = 0
    for coordinates in queriesCellsXY[i]:
        for mbr in gridIntersectsWith[coordinates[0]][coordinates[1]]:
```

- **coordinates** : η δυάδα (x, y) του υπό εξέταση κελιού.
- **coordinates[0]**, **coordinates[1]** : το X και Y αντίστοιχα, του υπό εξέταση κελιού.
- **mbr** : το Identifier του υπό εξέταση αντικειμένου.

Για όλα τα Identifiers των αντικειμένων εντός της κάθε δυάδας που αντιπροσωπεύει ένα κελί που συσχετίζεται με το παράθυρο του query, κάνουμε πάλι την σύγκριση των πολλών περιπτώσεων που έχει αναλυθεί στο 1ο μέρος και χρησιμοποιήθηκε και λίγο παραπάνω. Σε αντίθεση με την ανάλυση της σύγκρισης που συγκρίναμε ένα Cell με το MBR ενός αντικειμένου, εδώ συγκρίνουμε το παράθυρο με το MBR ενός αντικειμένου. Αν το αντικείμενο πληροί τις προϋποθέσεις και δεν έχει συμπεριληφθεί ξανά στα αποτελέσματα του query που εξετάζουμε, προστίθεται στον πίνακα uniqueResults με τα μοναδικά αποτελέσματα και στον πίνακα που κρατάει τα αποτελέσματα. Επίσης ενημερώνεται και ο αριθμός του πλήθος των αποτελεσμάτων. Όταν τελειώσουμε με τον έλεγχο των συγκρίσεων των αντικειμένων για την δυάδα του κελιού που εξετάζουμε από τον queriesCellsXY πίνακα, ελέγχουμε αν έχει αντικείμενα εντός του αυτό το κελί. Στην περίπτωση που δεν έχει κανένα αντικείμενο, διαγράφουμε την θέση του queriesCellsXY που περιέχει αυτήν την δυάδα που εξετάζουμε, έτσι ώστε να μην προστεθεί στα εξεταστέα κελιά, που θα τυπωθούν στον χρήστη κατά το τέλος του προγράμματος. Επίσης αυξάνουμε και την μεταβλητή coordinatesCounter κατά 1.

Αφού τελειώσει ο έλεγχος των παραθύρων για κάθε query και ενημερωθούν οι πίνακες του, εκτυπώνεται στον χρήστη ο αριθμός του query, τα αποτελέσματα του, ο αριθμός των κελιών που ψάχτηκαν και το πλήθος των αποτελεσμάτων.

Reference point

Στο πρόγραμμα δεν γίνεται έλεγχος του σημείου αναφοράς. Αυτό γίνεται επειδή στον έλεγχο που κάνουμε τα κελιά αρχικά είναι ταξινομημένα. Δηλαδή σε παράδειγμα με τα κελιά (8,0) , (8,1) , (9,0) και (9,1), θα ελεγχθούν πρώτα τα κελιά του (8,0) και αν βρεθούν και σε άλλο κελί δεν θα ξαναχρησιμοποιηθούν. Άρα με τον έλεγχο των ταξινομημένων κελιών, λύνεται το πρόβλημα της καταμέτρησης ενός αποτελέσματος πάνω από μια φορά.

ΜΕΡΟΣ 3ο

Διάβασμα αρχείου grid.dir

Το διάβασμα του αρχείου grid.dir είναι ίδιο με το μέρος 2ο , οπότε δεν θα αναλυθεί.

Υπολογισμός σχάρας

Ο υπολογισμός της σχάρας είναι ίδιος με το μέρος 1ο και το μέρος 2ο , οπότε δεν θα αναλυθεί.

Διάβασμα αρχείου grid.grd

Το διάβασμα του αρχείου grid.grd είναι ίδιο με το μέρος 2ο , οπότε δεν θα αναλυθεί.

Διάβασμα αρχείου queries.txt

Το διάβασμα του αρχείου queries.txt είναι ίδιο με το μέρος 2ο , οπότε δεν θα αναλυθεί.

Συνάρτηση εύρεσης τομής μεταξύ δυο ευθύγραμμων τμημάτων

Παρακάτω ο κώδικας της συνάρτησης:

```
def line_intersection(p1, p2, p3, p4):
    [x1, y1] = p1
    [x2, y2] = p2
    [x3, y3] = p3
    [x4, y4] = p4
    # calculate the denominator
    denom = (y4 - y3) * (x2 - x1) - (x4 - x3) * (y2 - y1)
    # if the denominator is 0, the lines are parallel
    if denom == 0:
        return False
    # calculate the numerator for the first line
    numer1 = (x4 - x3) * (y1 - y3) - (y4 - y3) * (x1 - x3)
    # calculate the numerator for the second line
    numer2 = (x2 - x1) * (y1 - y3) - (y2 - y1) * (x1 - x3)
    # calculate the parameters t and u
    t = numer1 / denom
    u = numer2 / denom
    # if 0 <= t <= 1 and 0 <= u <= 1, the lines intersect
    if 0 <= t <= 1 and 0 <= u <= 1:
        return True
    # otherwise, the lines do not intersect
    return False
```

Αρχικά, ας αναλύσουμε τις παραμέτρους της συνάρτησης:

- **p1** : Συντεταγμένες 1ου σημείου 1ου ευθύγραμμου τμήματος. Υπο μορφή->[x, y]
- **p2** : Συντεταγμένες 2ου σημείου 1ου ευθύγραμμου τμήματος. Υπο μορφή->[x, y]
- **p3** : Συντεταγμένες 1ου σημείου 2ου ευθύγραμμου τμήματος. Υπο μορφή->[x, y]
- **p4** : Συντεταγμένες 2ου σημείου 2ου ευθύγραμμου τμήματος. Υπο μορφή->[x, y]

Αφού αναθέσουμε σε πίνακες 2 μεταβλητών για να μπορέσουμε να διαχειριστούμε πιο εύκολα τις συντεταγμένες των ευθειών, υπολογίζουμε αρχικά τον παρονομαστή. Σε περίπτωση που ο παρονομαστής είναι 0, τότε καταλαβαίνουμε ότι τα δύο ευθύγραμμα τμήματα είναι παράλληλα, δηλαδή δεν γίνεται να τέμνονται, άρα και επιστρέφουμε False. Αν δεν είναι παράλληλα, υπολογίζουμε τους αριθμητές του t και του u . Έχοντας τους δυο αριθμητές και τον παρονομαστή, υπολογίζουμε την μεταβλητή t και την μεταβλητή u. Αν τα t και u είναι μεγαλύτερα ή ίσα του μηδενός και μικρότερα ή ίσα του 1, τότε τα δύο ευθύγραμμα τμήματα τέμνονται και έτσι επιστρέφουμε True. Σε περίπτωση αποτυχίας της σύγκρισης του t και του u, επιστρέφουμε False, καθώς αν και δεν είναι παράλληλα τα ευθύγραμμα τμήματα, δεν τέμνονται. Για την πραγματοποίηση της συγκρικριμένης συνάρτησης, χρησιμοποιήθηκε η παρακάτω φόρμουλα : (https://en.wikipedia.org/wiki/Line%E2%80%93line_intersection#Given_two_points_on_each_line_segment)

Given two points on each line segment [edit]

See also: Intersection_(geometry) § Two_line_segments

The intersection point above is for the infinitely long lines defined by the points, rather than the line segments between the points, and can produce an intersection point not contained in either of the two line segments. In order to find the position of the intersection in respect to the line segments, we can define lines L_1 and L_2 in terms of first degree Bézier parameters:

$$L_1 = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} + t \begin{bmatrix} x_2 - x_1 \\ y_2 - y_1 \end{bmatrix}, \qquad L_2 = \begin{bmatrix} x_3 \\ y_3 \end{bmatrix} + u \begin{bmatrix} x_4 - x_3 \\ y_4 - y_3 \end{bmatrix}$$

(where t and u are real numbers). The intersection point of the lines is found with one of the following values of t or u, where

$$t = \frac{\begin{vmatrix} x_1 - x_3 & x_3 - x_4 \\ y_1 - y_3 & y_3 - y_4 \end{vmatrix}}{\begin{vmatrix} x_1 - x_2 & x_3 - x_4 \\ y_1 - y_2 & y_3 - y_4 \end{vmatrix}} = \frac{(x_1 - x_3)(y_3 - y_4) - (y_1 - y_3)(x_3 - x_4)}{(x_1 - x_2)(y_3 - y_4) - (y_1 - y_2)(x_3 - x_4)}$$

and

$$u = \frac{\begin{vmatrix} x_1 - x_3 & x_1 - x_2 \\ y_1 - y_3 & y_1 - y_2 \end{vmatrix}}{\begin{vmatrix} x_1 - x_2 & x_3 - x_4 \\ y_1 - y_2 & y_3 - y_4 \end{vmatrix}} = \frac{(x_1 - x_3)(y_1 - y_2) - (y_1 - y_3)(x_1 - x_2)}{(x_1 - x_2)(y_3 - y_4) - (y_1 - y_2)(x_3 - x_4)},$$

with

$$(P_x, P_y) = \big(x_1 + t(x_2 - x_1),\ y_1 + t(y_2 - y_1)\big) \quad \text{or} \quad (P_x, P_y) = \big(x_3 + u(x_4 - x_3),\ y_3 + u(y_4 - y_3)\big)$$

There will be an intersection if $0 \le t \le 1$ and $0 \le u \le 1$. The intersection point falls within the first line segment if $0 \le t \le 1$, and it falls within the second line segment if $0 \le u \le 1$. These inequalities can be tested without the need for division, allowing rapid determination of the existence of any line segment intersection before calculating its exact point.[2]

Υπολογισμός αποτελεσμάτων των queries

Αρχικά εκτελείται ο ίδιος κώδικας για τον υπολογισμό των αποτελεσμάτων των queries, με την μόνη διαφορά να είναι η χρήση του allResultsMBRs, ενός πίνακα μήκους ίσο με τον αριθμό των queries. Εντός του κρατούνται πίνακες με όλα τα αποτελέσματα αντικειμένων για το κάθε query. Ο πίνακας με τα αποτελέσματα για το query 1, βρίσκονται στο allResultsMBRs[0].

Στην συνέχεια, για να κάνουμε το refinement stage πρέπει να ελέγξουμε όλα τα αποτελέσματα περαιτέρω με την βοήθεια της συνάρτησεις που δημιουργήσαμε.

Ξεκινάμε με την εκτέλεση μιας for, που τρέχει μια φορά για κάθε query. Στο εσωτερικό της, αρχικοποιούμε τις παρακάτω μεταβλητές:

- **uniqueResults2** : πίνακας ο οποίος θα μας βοηθήσει να μην έχουμε διπλότυπα αντικειμένων στα αποτελέσματα του κάθε query.
- **queryResults** : πίνακας που κρατάει τα αποτελέσματα που θα τυπωθούν στον χρήστη για κάθε query.

Σύμφωνα με την παρακάτω εικόνα, αρχικοποιούμε τις συντεταγμένες των 4 ευθειών του παραθύρου.

```
# queries[i][0] = Xmin OF THE WINDOW
# queries[i][1] = Xmax OF THE WINDOW
# queries[i][2] = Ymin OF THE WINDOW
# queries[i][3] = Ymax OF THE WINDOW
```

Προχωράμε στον ορισμό των συντεταγμένων των 8 σημείων που καθορίζουν τα 4 ευθύγραμμα τμήματα του παραθύρου.

- **gridFirstLineCoordinate1** : 1^{ο} σημείο του 1^{ου} ευθύγραμμου τμήματος του παραθύρου.
- **gridFirstLineCoordinate2** : 2^{ο} σημείο του 1^{ου} ευθύγραμμου τμήματος του παραθύρου.
- **gridSecondLineCoordinate1** : 1^{ο} σημείο του 2^{ου} ευθύγραμμου τμήματος του παραθύρου.
- **gridSecondLineCoordinate2** : 2^{ο} σημείο του 2^{ου} ευθύγραμμου τμήματος του παραθύρου.
- **gridThirdLineCoordinate1** : 1^{ο} σημείο του 3^{ου} ευθύγραμμου τμήματος του παραθύρου.
- **gridThirdLineCoordinate2** : 2^{ο} σημείο του 3^{ου} ευθύγραμμου τμήματος του παραθύρου.
- **gridFourthLineCoordinate1** : 1^{ο} σημείο του 4^{ου} ευθύγραμμου τμήματος του παραθύρου.
- **gridFourthLineCoordinate2** : 2^{ο} σημείο του 4^{ου} ευθύγραμμου τμήματος του παραθύρου.

Στην συνέχεια εκτελούμε μια for για κάθε δυάδα (x, y) του πίνακα queriesCellsXY, και εντός της μια for για κάθε Identifier των αντικειμένων που επιλέχθηκαν για το κάθε query. Αρχικά ελέγχουμε αν το MBR του αντικειμένου που εξετάζουμε υπερκαλύπτεται εντελώς σε τουλάχιστον μία από τις δύο διαστάσεις x ή y από το παράθυρο. Αν δεν ισχύει αυτό, με μια for διατρέχουμε όλες τις δυάδες των linestring του αντικειμένου που εξετάζουμε [x1, y1], και στο **coordinatesIndex** έχουμε την θέση της δυάδας στον πίνακα. Έχοντας στην διάθεση μας τις δυάδες των linestrings, δημιουργούμε τις παρακάτω μεταβλητές με την βοήθεια των δυάδων που έχουμε στην θέση coordinatesIndex και τις δυάδες στην θέση coordinatesIndex + 1, για να δημιουργήσουμε ευθύγραμμο τμήμα.

- **LineCoordinate1** : 1^{ο} σημείο ενός ευθύγραμμου τμήματος του υπό εξέταση αντικειμένου.
- **LineCoordinate2** : 2^{ο} σημείο ενός ευθύγραμμου τμήματος του υπό εξέταση αντικειμένου.

Εφόσον έχουμε τις συντεταγμένες του ευθύγραμμου τμήματος, εκτελούμε την συνάρτηση εύρεσης τομής δυο ευθύγραμμων τμημάτων line_intersection μεταξύ του ευθύγραμμου τμήματος του αντικειμένου και των 4 ευθυγράμμων τμημάτων του παραθύρου. Αν ισχύει έστω και μια από αυτές τις 4 συγκρίσεις, τοποθετούμε τον Identifier του αντικειμένου στον πίνακα uniqueResults2 και queryResults, αν δεν έχει καταχωρηθεί ξανά στο παρελθόν. Αυτό εκτελείται για όλα τα ευθύγραμμα τμήματα ενός αντικειμένου.

Αφού τελειώσει ο έλεγχος των ευθυγράμμων τμημάτων όλων των αντικειμένων για κάθε query και ενημερωθούν οι πίνακες του, εκτυπώνεται στον χρήστη ο αριθμός του query, τα αποτελέσματα του, ο αριθμός των κελιών που ψάχτηκαν και τέλος, το πλήθος των αποτελεσμάτων.

ΤΕΛΟΣ