Πολυτεχνείο Κρήτης

Γεωγραφικά Συστήματα Πληροφοριών

Διδάσκων: Παναγιώτης Παρτσινέβελος, Αν. Καθηγητής

Εργαστήριο: Γεώργιος Πετράκης, Υπ. Διδάκτωρ

Όνομα / Επίθετο:

Σειρά Ασκήσεων Python

1) Γράψτε ένα πρόγραμμα Python το οποίο να δέχεται από το πληκτρολόγιο τις καρτεσιανές συντεταγμένες δύο σημείων `A(x1, y1)` και `B(x2, y2)` και να υπολογίζει το μήκος του ευθύγραμμου τμήματος AB με χρήση του τύπου:

AB = $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$

In [ ]:
```
# Your code here...
```

2) Γράψτε ένα πρόγραμμα Python το οποίο δέχεται από το πληκτρολόγιο τις καρτεσιανές συντεταγμένες (x, y) σημείων στο επίπεδο. Μετά από την εισαγωγή του ζεύγους συντεταγμένων(x, y) το πρόγραμμα πρέπει να εμφανίζει το μήνυμα "T1", "T2", "T3" ή "T4" ανάλογα με το τεταρτημόριο στο οποίο ανήκει το σημείο του οποίου οι συντεταγμένες δόθηκαν.

In [ ]:
```
# Your code here...
```

3) Δημιουργήστε μια συνάρτηση (function) που να ονομάζεται `create_point()` και θα δέχεται δύο παραμέτρους (x, y). Η συνάρτηση θα δημιουργεί ένα Point αντικείμενο και θα το επιστρέφει. Χρησιμοποιείστε τη βιβλιοθήκη shapely. Τέλος καλέστε την συνάρτηση ώστε να αποδείξετε τη λειτουργία της.

In [ ]:
```
# Your code here...
```

4) Δημιουγήστε μια συνάρτηση που να ονομάζεται `create_line()` η οποία να δέχεται ως παράμετρο μία λίστα από shapely point αντικείμενα που θα ονομάζονται points, και θα επιστρέφει ένα LineString αντικείμενο από τα σημεία που εισήχθησαν σαν input. Η συνάρτηση θα ελέγχει αν η παράμετρος είναι λίστα ενώ αν δεν είναι, θα βγάζει το μήνυμα λάθους: `Input should be a list`. Επίσης θα ελέγχει αν η λίστα που εισάγεται έχει τουλάχιστον δύο τιμές. Σε διαφορετική περίπτωση θα πρέπει να επιστρέφει το μήνυμα λάθους: `LineString object requires at least two Points!`. Τέλος καλέστε την συνάρτηση ώστε να αποδείξετε τη λειτουργία της.

In [ ]:
```
# Your code here...
```

5) Δημιουργήστε μία συνάρτηση με το όνομα `create_poly` όπου θα δέχεται την παράμετρο `coords`. Η παράμετρος αυτή θα είναι μία λίστα από tuples. Η συνάρτηση θα δημιουργεί και θα επιστρέφει ένα Polygon αντικείμενο βασισμένο στις συντεταγμένες της παραμέτρου `coords`. Θα ελέγχει αν η παράμετρος είναι λίστα ενώ αν δεν είναι, θα βγάζει το μήνυμα λάθους: `Input should be a list`. Επίσης θα ελέγχει αν η λίστα που εισάγεται έχει τουλάχιστον τρεις τιμές. Σε διαφορετική περίπτωση θα πρέπει να επιστρέφει το μήνυμα λάθους: `Polygon object requires at least three Points!`. Τέλος καλέστε την συνάρτηση ώστε να αποδείξετε τη λειτουργία της.

In [ ]:
```
# Your code here...
```

6) Δημιουργήστε μία κλάση η οποία να δημιουργεί ένα Polygon αντικείμενο ενώ με τις μεθόδους `get_area()` και `get_centroid()` που θα περιλαμβάνει, θα υπολογίζει την επιφάνεια του και το κέντρο βάρους του (centroid) αντίστοιχα. Αποδείξτε την λειτουργία της, δημιουργώντας το αντικείμενο `myPolygon` και υπολογίστε την επιφάνεια και το κέτρο βάρους του μέσω της κλάσης που φτιάξατε.
Προαιρετικά: Προσθέστε περιορισμούς μέσω της λέξης κλειδί `assert`, ώστε να εμφανίζονται τα σφάλματα που αναφέρθηκαν στις παραπάνω ασκήσεις (`Input should be a list`, `Polygon object requires at least three Points!`) στις ανάλογες περιπτώσεις.

In [ ]:
```
# Your code here...
```

7 α) Γράψτε ένα πρόγραμμα σε Python, το οποίο να διαβάζει το αρχείο oikismoi2000 (το οποίο θα κατεβάσετε από το e-class του μαθήματος), και θα υπολογίζει τα εμβαδά των τελευταίων 10 οντοτήτων του αρχείου ενώ θα εμφανίζει στην οθόνη το σύστημα αναφοράς του (EPSG). Στη συνέχεια δημιουργήστε ένα αντίγραφο του αρχείου χρησιμοποιώντας την συνάρτηση `copy()` που θα λειτουργεί σαν backup και μετασχηματίστε το σύστημα αναφοράς του αρχείου που έχει φορτωθεί στη μνήμη στο σύστημα WGS84. Στη συνέχεια γράψτε το αποτέλεσμα στο δίσκο δημιουργώντας ένα νέο shapefile με το όνομα `oikismoi2000_wgs84.shp`.
β) Γράψτε μία for loop που να ελέγχει αν το σημείο (509699, 3929906) βρίσκεται εντός κάποιου από τους 1252 οικισμούς που βρίσκονται στο αρχικό σας αρχείο και εμφανίστε τις συντεταγμένες του αντίστοιχου πολυγώνου.

In [ ]:
```
# Your code here...
```

Tip: Για να επιβεβαιώνεται ότι κάποιο στοιχείο που χρησιμοποιεί το πρόγραμμα είναι στη μορφή που θέλει ο προγραμματιστής χρησιμοποιείστε τη λέξη-κλειδί assert:

In [1]:
```
posNumber = input('insert a positive number: ')
assert int(posNumber) > 0, 'please insert a positive number'
print(posNumber)
```

```
insert a positive number: -4
---------------------------------------------------------------------------
AssertionError                            Traceback (most recent call last)
<ipython-input-1-5cc6be2e0861> in <module>
      1 posNumber = input('insert a positive number: ')
----> 2 assert int(posNumber) > 0, 'please insert a positive number'
      3 print(posNumber)

AssertionError: please insert a positive number
```

Απαραίτητες οδηγίες για την επίλυση και υποβολή των ασκήσεων

Κάτω απο κάθε εκφώνηση βρίσκεται ένα πεδίο (cell) (# your code here...) που θα γράψετε τον κώδικα σας ώστε να επιλυθεί η άσκηση. Για να εκτελεστεί ο κώδικας σε κάθε cell θα πρέπει να επιλέγετε shift-enter. Τα αποτελέσματα ή τα σφάλματα που θα εξάγονται, θα εμφανίζονται ακριβώς από κάτω.

Πριν την υποβολή επιλέξτε με το ποντίκι το όνομα του αρχείου: Exercises_Python_all_final που βρίσκεται πάνω αριστερά της οθόνης και μετονομάστε το σε: Ex_python_epitheto_onoma

Επίσης στο κενό cell που υπάρχει κάτω από το "Ονομα / Επίθετο", πληκτρολογείστε το όνομα και το επίθετο σας. Στη συνέχεια επιλέξτε shift-enter.

Η υποβολή της άσκησης θα γίνεται αποδεκτή ΜΟΝΟ μέσω του e-class. Κατά την υποβολή της σειράς ασκήσεων Python θα παραδώσετε ΜΟΝΟ το αρχείο .ipynb που περιέχει τις εκφωνήσεις με συμπληρωμένο τον κώδικα που γράψατε για κάθε άσκηση.

Η προθεσμία υποβολής της σειράς ασκήσεων Python είναι την Τετάρτη 18 / 11 / 2020 και ώρα 00:00 το βράδυ.