Κλάσεις και Κληρονομικότητα στην Python

Στον προγραμματισμό γενικά προσπαθούμε να αποφεύγουμε τις επαναληπτικές εργασίες. Προσπαθούμε να γράφουμε κώδικα μία φορά και να τον χρησιμοποιούμε ξανά. Η αποφυγή της επανάληψης κώδικα επιτυγχάνεται με την χρήση συναρτήσεων. Οι συναρτήσεις τυποποιούν λειτουργίες. Όποτε το πρόγραμμα μας πρέπει να εκτελέσει αυτές τις λειτουργίες, καλούμε τις συναρτήσεις αυτές, περνώντας τις κατάλληλες παραμέτρους. Η χρήση συναρτήσεων δεν μας δίνει την δυνατότητα να αποθηκεύσουμε μέσα τους πληροφορίες. Οι τιμές των τοπικών τους μεταβλητών χάνονται μετά την επιστροφή εκτός αν περαστούν σαν παράμετροι, ή τις επιστρέψουν, οπότε η καλούσα συνάρτηση μπορεί να κρατήσει τις τιμές.

Με τον αντικειμενοστραφή προγραμματισμό και την χρήση κλάσεων δίνεται η δυνατότητα διαχείρισης δεδομένων και συναρτήσεων με πολύ ολοκληρωμένο τρόπο.

Στον αντικειμενοστραφή προγραμματισμό προτεραιότητα δίνεται στα δεδομένα και όχι στις συναρτήσεις όπως γίνεται στον διαδικαστικό προγραμματισμό (Procedural Programming).

Η Python, σε αντίθεση με τη Java επιτρέπει δυναμικούς τύπους.

Δημιουργία κλάσεων

Μία κλάση είναι ένα πρότυπο. Δεν έχει υπόσταση από μόνη της. Είναι όπως τα σχέδια κατασκευής για αντικείμενα. Περιγράφει πως μπορεί να λειτουργήσει ένα ή περισσότερα αντικείμενα, που έχει ή έχουν δημιουργηθεί υλοποιώντας την κλάση.

Παράδειγμα 1: Κώδικας για τον ορισμό μίας κλάσης

```
# Defining a class

class class_name:
        [statement 1]
        [statement 2]
        [statement 3]
        [etc.]
```

Οι κλάσεις περιέχουν:

- πεδία (δεδομένα)
- μεθόδους (συναρτήσεις)

Παράδειγμα 2: Κώδικας για τη δημιουργία της κλάσης Shape:

```
#An example of a class

class Shape:
    def __init__(self,x,y):
        self.x = x
        self.y = y
        description = "This shape has not been described yet"
        author = "Nobody has claimed to make this shape yet"

    def area(self):
        return self.x * self.y

    def perimeter(self):
        return 2 * self.x + 2 * self.y

    def describe(self,text):
        self.description = text

    def authorName(self,text):
        self.author = text

    def scaleSize(self,scale):
        self.x = self.x * scale
        self.y = self.y * scale
```

Έχουμε δημιουργήσει μια περιγραφή ενός σχήματος όπως και τις μεταβλητές και τις ενέργειες που μπορούμε να κάνουμε με το Shape. Όμως δεν έχουμε κατασκευάσει ένα πραγματικό σχήμα, αλλά απλά έχουμε ορίσει την περιγραφή του τι είναι ένα Shape. Το Shape έχει πλάτος (x), ύψος (y), και μας δίνει την δυνατότητα να υπολογίσουμε το εμβαδόν και την περίμετρο του.

Η συνάρτηση __init__ εκτελείται όταν δημιουργήσουμε ένα αντικείμενο (instance ή “στιγμιότυπο”) της Shape, δηλαδή, όταν δημιουργήσουμε ένα πραγματικό Shape, σε αντίθεση με το “σχέδιο” που έχουμε εδώ, τότε η __init__ εκτελείται αυτόματα.

Το self δείχνει ότι αναφερόμαστε σε στοιχεία της ίδιας της κλάσης. Αποτελεί την πρώτη παράμετρο σε όλες τις λειτουργίες που ορίζονται μέσα σε μια τάξη. Κάθε λειτουργία ή μεταβλητή που δημιουργήθηκε στο πρώτο επίπεδο της εσοχής, δηλαδή γραμμές κώδικα που ξεκινούν ένα tab δεξιά της θέσης όπου βάζουμε την κλάση Shape. Για πρόσβαση σε αυτές τις λειτουργίες και τις μεταβλητές

αλλού μέσα στην τάξη, το όνομά τους θα πρέπει να προηγείται η λέξη self ακολουθούμενη από μία τελεία (π.χ. self.variable_name).

Χρήση μίας κλάσης

Ενώ η κλάση είναι τα σχέδια, ή οι οδηγίες για την κατασκευή αντικειμένων, η δημιουργία των αντικειμένων (instances) είναι αυτή που δεσμεύει μνήμη για την αποθήκευση των πεδίων τους. (Ο κώδικας είναι κοινός για όλα τα αντικείμενα μίας κλάσης).

Παράδειγμα 3: Κώδικας για τη δημιουργία αντικειμένου της κλάσης

```
#creating an object

rectangle = Shape(100, 45)
```

Η συνάρτηση __init__ καλείται αυτόματα. Δημιουργούμε ένα αντικείμενο (“στιγμιότυπο”) της κλάσης δίνοντας το όνομά του (σε αυτή την περίπτωση, το Shape) και, στη συνέχεια, σε παρένθεση, οι τιμές για να περάσει στη λειτουργία __init__. Η συνάρτηση init εκτελείται (χρησιμοποιώντας τις παραμέτρους που δόθηκαν σε παρένθεση) και, στη συνέχεια, κατασκευάζει ένα αντικείμενο της κλάσης, η οποία στην περίπτωση αυτή αποδίδεται στη μεταβλητή rectangle.

Το αντικείμενο rectangle αποτελεί αυτοτελή συλλογή μεταβλητών και συναρτήσεων. Με τον ίδιο τρόπο που χρησιμοποιήσαμε την self για να αποκτήσουμε πρόσβαση στις συναρτήσεις και τις μεταβλητές της κλάσης μέσω της self, χρησιμοποιούμε το όνομα που της έχει ανατεθεί τώρα (rectangle) για την πρόσβαση στις λειτουργίες και τις μεταβλητές της κλάσης από έξω από την κλάση.

Παράδειγμα 4: Πρόσβαση χαρακτηριστικών εκτός κλάσης

```
#creating an object
rectangle=Shape(100,45)

#finding the area of your rectangle:
print(rectangle.area())

#finding the perimeter of your rectangle:
print(rectangle.perimeter())

#describing the rectangle
rectangle.describe('A wide rectangle')

#making the rectangle 50% smaller
```

```
rectangle.scaleSize(0.5)

#re-printing the new area of the rectangle
print(rectangle.area())
```

Παράδειγμα 5: Δημιουργία περισσότερων του ενός αντικειμένων

```
long_rectangle = Shape (120,10)
fat_rectangle = Shape (130,120)
```

Κληρονομικότητα

Παράδειγμα 6: Η κλάση Shape μπορεί να χρησιμοποιηθεί σαν βάση για τη δημιουργία παράγωγων κλάσεων με πιο εξειδικευμένα χαρακτηριστικά.

```
#An example of a class

class Shape:
    def __init__(self,x,y):
        self.x = x
        self.y = y
        description = "This shape has not been described yet"
        author = "Nobody has claimed to make this shape yet"

    def area(self):
        return self.x * self.y

    def perimeter(self):
        return 2 * self.x + 2 * self.y

    def describe(self,text):
        self.description = text

    def authorName(self,text):
        self.author = text

    def scaleSize(self,scale):
        self.x = self.x * scale
        self.y = self.y * scale
```

Για να ορίσουμε μια νέα κλάση, π.χ. την κλάση Square, δηλαδή τετράγωνο, με βάση την προηγούμενη κλάση Shape, κάνουμε αυτό:

Παράδειγμα 7: Κληρονομικότητα

```
#An inherited class

class Square(Shape):
    def __init__(self,x):
        self.x = x
        self.y = x
```

Ο ορισμός της υποκλάσης γίνεται ακριβώς όπως ορίζεται μια κλάση, με την διαφορά ότι βάζουμε σε παρένθεση μετά το όνομα, το όνομα της μητρικής κλάση από την οποία κληρονομήσαμε. Έτσι έχουμε περιγράψει ένα τετράγωνο πολύ γρήγορα, χρησιμοποιώντας την κλάση Shape. Αυτό συμβαίνει γιατί έχουμε κληρονομήσει τα πάντα, από την κλάση Shape και αλλάξαμε μόνο ότι χρειάζεται να αλλάξει. Σε αυτή την περίπτωση, έχουμε επαναπροσδιορίσει την λειτουργία __init__ του Shape, έτσι ώστε οι τιμές x και y να είναι πάντα ίδιες.

Παράδειγμα 8 - DoubleSquare κατηγορία

```
# The shape looks like this:
# __________
#|    |    |
#|    |    |
#|____|____|

class DoubleSquare(Square):
    def __init__(self,y):
        self.x = 2 * y
        self.y = y
    def perimeter(self):
        return 2 * self.x + 2 * self.y
```

Στο παράδειγμα αυτό ορίζουμε ξανά την μέθοδο που υπολογίζει την περίμετρο, έτσι ώστε να μπορέσει να υπολογιστεί και η κοινή πλευρά εύκολα.

Λεξικά

Τα λεξικά είναι συλλογές από δεδομένα, αταξινόμητα, τα οποία αποθηκεύουν ζευγάρια τιμών και χρησιμοποιούν σαν δείκτη την πρώτη.

```
>>> d={}
>>> d['Nikos']='2810-323322'
>>> d['Makis']='2810-226676'
>>> d['Maria']='2810-334229'
>>> d
{'Nikos': '2810-323322', 'Makis': '2810-226676', 'Maria': '2810-
334229'}
>>> print(d['Makis'])
2810-226676
>>> for x in d:
     print(x)

Nikos
Makis
Maria
>>> for x in d:
     print(d[x])

2810-323322
2810-226676
2810-334229
>>> for x, y in d.items():
     print(x, y)

Nikos 2810-323322
Makis 2810-226676
Maria 2810-334229
>>> for x in d.values():
     print(x)

2810-323322
2810-226676
2810-334229
>>> if 'Maria' in d:
     print('Maria is in the d dictionary')
```

```
Maria is in the d dictionary
>>>
```

Παράδειγμα 9: Λεξικό

```
# First, create a dictionary:
dictionary = {}

# Then, create some instances of classes in the dictionary:
dictionary["DoubleSquare1"] = DoubleSquare(5)
dictionary["long_rectangle"] = Shape(600,45)
dictionary["DoubleSquare1"].authorName("The Gingerbread Man")

#You can now use them like a normal class:
print(dictionary["long_rectangle"].area())
print(dictionary["DoubleSquare1"].author)
```

Από την ιστοσελίδα “A Beginner's Python Tutorial/Classes”, που βρίσκεται στο link:

https://en.wikibooks.org/wiki/A_Beginner%27s_Python_Tutorial/Classes