Tartalomjegyzék

Az el?z? gyakorlat példájának megoldása

```
import sys
import math

def reader(fileName):
        dataSet =
    open(fileName,'r')
for line in f:
            rstrip("\n")line.
         split("L")= line.
       []       data =
for l in L:
     append(float(l))   data.
        append(data)dataSet.
return dataSet

def learn(trainingData):
        averages =
for i in range(0,10):
         append(0)averages.
for user in trainingData:
for i in range(0,10):
        [i]+=user[i+1]  averages
for i in range(10):
        [i] = averagesaverages[i] / float(len(trainingData))
return averages

def predict(averages,testingData):
        predictions =
for test in testingData:
                aboveAverage = 0
for i in range(0,10):
if(test[i+1]>averages[i]):
                                aboveAverage+=1
id = test[0]
        int(test[0])truth =
if aboveAverage > len(averages)*0.5:
            append([id,truth,1])predictions.
else:
            append([id,truth,0])predictions.
return predictions

def accuracy(predictions):
        acc = 0
for prediction in predictions:
if prediction[1] == prediction[2]:
                        acc += 1
return acc/float(len(predictions))

def learnAndPredict():
        trainingData = sys.argv[1])reader(
        testingData = sys.argv[2])reader(
```

```
        averages(trainingData)
        predictions =(averages,testingData)
print "Accuracy: ", accuracy(predictions)

learnAndPredict()
```

Osztályok

- Az osztályok egységbe zárnak és a felesleges információt elrejtik.
- Az osztályok metódusokat és változókat tartalmaznak. Metódusokat tekintsük osztályokon belüli függvényeknek.
- Tekintsünk erre úgy, hogy a kód egy részét (mind tárolókat, mind a függvényeket nézve) összefogjuk.
- Egy objektum egy osztály konkrét megvalósítása.

Egy metódust az alábbi módon definiálunk pythonban:

```
class Brick:
   def setx(self, x):
      self.x = x
   def bar(self):
      print self.x
```

- Itt a lényeges dolog a self változó, ami mindig az els? argumentuma a metódusnak. Ez mindig a konkrét megvalósított objektumot jelöli. Tehát külön értékadás nélkül self.x az osztályon (objektumon) belüli x változó, aminek az értékét állítjuk be a setx függvénnyel.
- Az alábbi módon hozhatunk létre objektumokat és hívhatjuk meg azok metódusait:

```
f = Brick()
f.setx(5)
f.bar()
Brick.setx(f,5)
Brick.bar(f)

>>> class A:
...   pass
...
>>> A.v1 = 10
>>> x=A() # létrehozok két példányt
>>> y=A()
>>> x.v1  # példányon belül is elérem az osztály szint? változót
10
>>> x.v1 = 34 # itt viszont létrehozok egy példány szint?t aminek a neve felülírja az osztály szin
>>> y.v1
10
>>> x.v1
34
>>> A.v1
10
>>> A.v1 = 12 # itt az osztályszint? változót változtatjuk
>>> x.v1 # mivel felülírtuk a v1-t az x példányban így ezen már nincs hatása
34
>>> y.v1
12
>>> A.v1
12
```

Feladatok

1. feladat

Ismételd meg a fenti kódrészleteket.

2. feladat

- Írj mátrix osztályt az alábbi metódusokkal:

```
class Matrix:
    def const(self,n,x):
      ...
    def zeros(self,n):
      ...
    def printer(self):
      ...
    def set(self,i,j,x):
      ...
    def get(self,i,j):
      ...
    def hasValue(self,i,j):
      ...
    def getLine(self,i):
      ...
    def pow(self):
      ...
    def product(self,v)
```

- próbálj meg hibakezelést beépíteni a függvényekbe (try --- except)