Regular Expressions / Κανονικές Εκφράσεις

Κεφάλαιο 11

Python για Όλους

www.py4e.com

Κανονικές Εκφράσεις

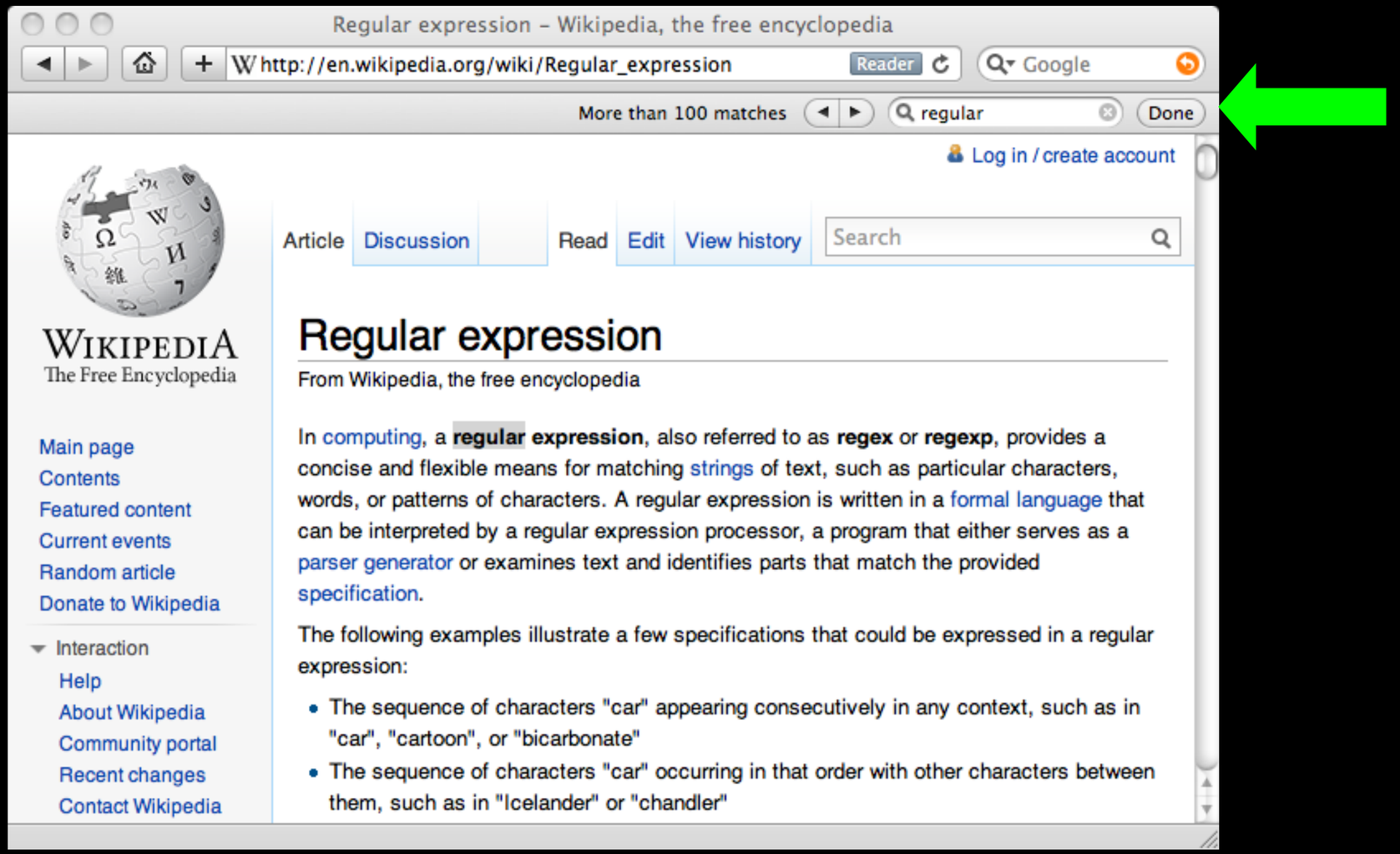
Στους υπολογιστές, μια κανονική έκφραση, που αναφέρεται επίσης ως «regex» ή «regexp», παρέχει ένα συνοπτικό και ευέλικτο μέσο για την αντιστοίχιση συμβολοσειρών, όπως συγκεκριμένους χαρακτήρες, λέξεις ή μοτίβα χαρακτήρων. Μια κανονική έκφραση γράφεται σε μια επίσημη γλώσσα που μπορεί να ερμηνευτεί από έναν επεξεργαστή κανονικών εκφράσεων.

http://en.wikipedia.org/wiki/Regular_expression

Κανονικές Εκφράσεις

Πραγματικά έξυπνες εκφράσεις με «μπαλαντέρ» για αντιστοίχιση και ανάλυση συμβολοσειρών

http://en.wikipedia.org/wiki/Regular_expression



Πραγματικά έξυπνη «Εύρεση» ή «Αναζήτηση»

Κατανόηση των Κανονικών Εκφράσεων

- Πολύ ισχυρές και αρκετά αινιγματικές
- Διασκεδαστικές άπαξ τις καταλάβετε
- Οι κανονικές εκφράσεις είναι μια γλώσσα από μόνες τους
- Μια γλώσσα «χαρακτήρων δεικτών» - προγραμματισμός με χαρακτήρες
- Είναι μια γλώσσα «παλιάς σχολής» - συμπαγής

http://xkcd.com/208/

Σύντομος Οδηγός Κανονικών Εκφράσεων

^	Ταιριάζει την αρχή μιας γραμμής
$	Ταιριάζει το τέλος μιας γραμμής
.	Ταιριάζει οποιονδήποτε χαρακτήρα
\s	Ταιριάζει κενό διάστημα (μη ορατό χαρακτήρα)
\S	Ταιριάζει οποιοδήποτε μη-κενό διάστημα (ορατό χαρακτήρα)
*	Επαναλαμβάνει ένα χαρακτήρα καμία ή περισσότερες φορές
*?	Επαναλαμβάνει ένα χαρακτήρα καμία ή περισσότερες φορές (μη-άπληστα)
+	Επαναλαμβάνει ένα χαρακτήρα μία ή περισσότερες φορές
+?	Επαναλαμβάνει ένα χαρακτήρα μία ή περισσότερες φορές (μη-άπληστα)
[aeiou]	Ταιριάζει έναν μόνο χαρακτήρα από το δοθέν σύνολο
[^XYZ]	Ταιριάζει έναν μόνο χαρακτήρα που δεν περιέχεται στο δοθέν σύνολο
[a-z0-9]	Το σύνολο των χαρακτήρων μπορεί να περιλαμβάνει ένα εύρος/διάστημα
(	Υποδεικνύει από πού θα ξεκινήσει η εξαγωγή της συμβολοσειράς
)	Υποδεικνύει πού θα τελειώσει η εξαγωγή της συμβολοσειράς

https://www.py4e.com/lectures3/Pythonlearn-11-Regex-Handout.txt

Η Ενότητα Κανονική Έκφραση

- Πριν χρησιμοποιήσετε τις κανονικές εκφράσεις στο πρόγραμμά σας, πρέπει να εισαγάγετε την αντίστοιχη βιβλιοθήκη χρησιμοποιώντας το «import re»
- Μπορείτε να χρησιμοποιήσετε το re.search() για να δείτε αν μια συμβολοσειρά ταιριάζει με μια κανονική έκφραση, παρόμοια με τη χρήση της μεθόδου find() για συμβολοσειρές
- Μπορείτε να χρησιμοποιήσετε re.findall() για την εξαγωγή τμημάτων μιας συμβολοσειράς που ταιριάζει με κανονική έκφραση σας, παρόμοια με ένα συνδυασμό find() και τεμαχισμού: var[5:10]

Χρήση του re.search() Αντί του find()

```
hand = open('mbox-short.txt')
for γραμμή in hand:
    γραμμή = γραμμή.rstrip()
    if γραμμή.find('From:') >= 0:
        print(γραμμή)
```

```
import re

hand = open('mbox-short.txt')
for γραμμή in hand:
    γραμμή = γραμμή.rstrip()
    if re.search('From:', γραμμή) :
        print(γραμμή)
```

Χρήση του re.search() Αντί του startswith()

```
hand = open('mbox-short.txt')
for γραμμή in hand:
    γραμμή = γραμμή.rstrip()
    if γραμμή.startswith('From:') :
        print(γραμμή)
```

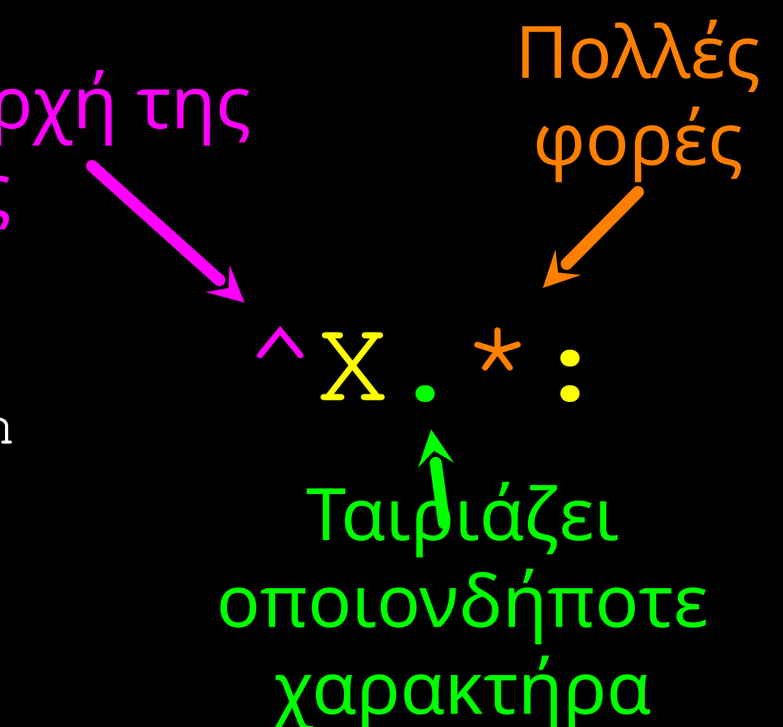
```
import re

hand = open('mbox-short.txt')
for γραμμή in hand:
    γραμμή = γραμμή.rstrip()
    if re.search('^From:', γραμμή) :
        print(γραμμή)
```

Ρυθμίζουμε με ακρίβεια αυτό που ταιριάζει προσθέτοντας ειδικούς χαρακτήρες στη συμβολοσειρά

Χαρακτήρες Μπαλαντέρ

- Ο χαρακτήρας τελεία ταιριάζει με οποιονδήποτε χαρακτήρα
- Αν προσθέσετε τον αστερίσκο, ο χαρακτήρας είναι «όσες φορές»

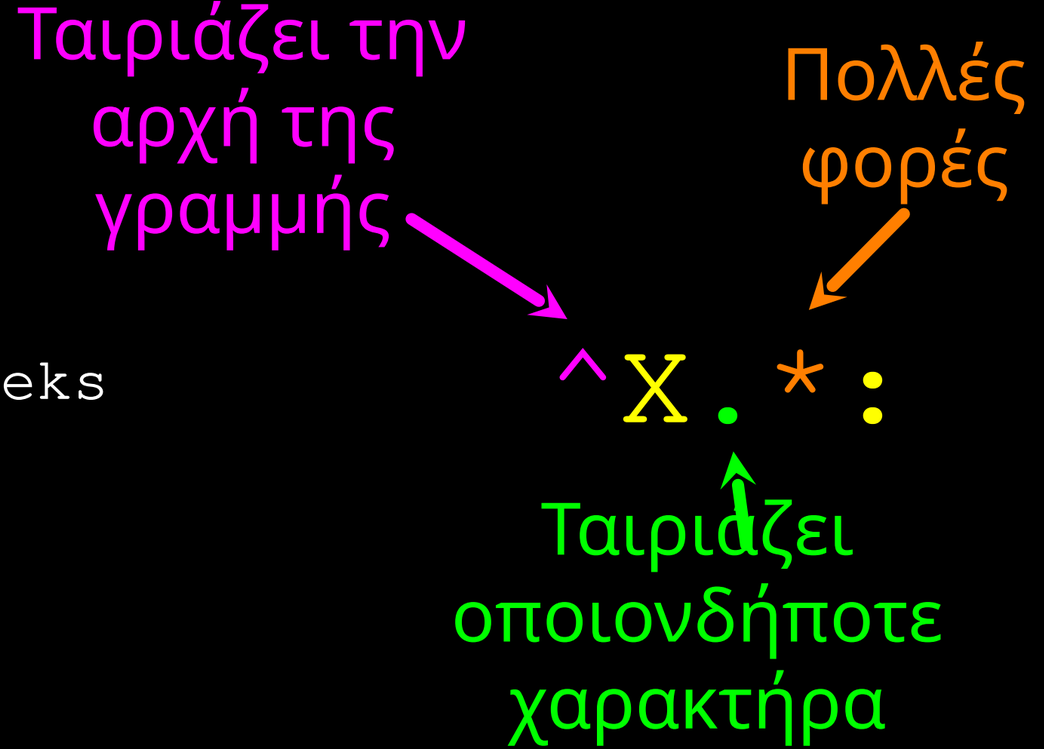
```
X-Sieve: CMU Sieve 2.3
X-DSPAM-Result: Innocent
X-DSPAM-Confidence: 0.8475
X-Content-Type-Message-Body: text/plain
```



Βελτιστοποιήστε το Ταίριασμα

Ανάλογα με το πόσο «καθαρά» είναι τα δεδομένα σας και τον σκοπό της εφαρμογής σας, μπορεί να θέλετε να περιορίσετε λίγο την αντιστοίχιση

```
X-Sieve: CMU Sieve 2.3
X-DSPAM-Result: Innocent
X-Plane is behind schedule: two weeks
X-: Very short
```



Βελτιστοποιήστε το Ταίριασμα

Ανάλογα με το πόσο «καθαρά» είναι τα δεδομένα σας και τον σκοπό της εφαρμογής σας, μπορεί να θέλετε να περιορίσετε λίγο την αντιστοίχιση

```
X-Sieve: CMU Sieve 2.3
X-DSPAM-Result: Innocent
X-: Very Short
X-Plane is behind schedule: two weeks
```

Ταιριάζει την αρχή της γραμμής

Μία ή περισσότερες φορές

```
^X-\S+:
```

Ταιριάζει οποιονδήποτε μη-κενό χαρακτήρα

Ταίριασμα και Εξαγωγή Δεδομένων

- re.search() επιστρέφει Αληθές/Ψευδές ανάλογα με το αν η συμβολοσειρά ταιριάζει με την κανονική έκφραση
- Στην πραγματικότητα εάν θέλουμε να εξάγονται οι συμβολοσειρές που ταιριάζουν, χρησιμοποιούμε το re.findall()

```
[0-9]+
```

```
>>> import re
>>> x = 'Τα 2 αγαπημένα μου νούμερα είναι το 19 και το 42'
>>> y = re.findall('[0-9]+',x)
>>> print(y)
['2', '19', '42']
```

Ένα ή περισσότερα ψηφία

Αντιστοίχιση και Εξαγωγή Δεδομένων

Όταν χρησιμοποιούμε το re.findall(), μας επιστρέφει μια λίστα με καμία ή περισσότερες υπο-συμβολοσειρές που ταιριάζουν με την κανονική έκφραση

```
>>> import re
>>> x = 'Τα 2 αγαπημένα μου νούμερα είναι το 19 και το 42'
>>> y = re.findall('[0-9]+',x)
>>> print(y)
['2', '19', '42']
>>> y = re.findall('[AEIOU]+',x)
>>> print(y)
[]
```

Προειδοποίηση: Άπληστο Ταίριασμα

Οι χαρακτήρες επανάληψης (* και +) ωθούν προς τα έξω και προς τις δύο κατευθύνσεις (άπληστοι) για να ταιριάξουν με τη μεγαλύτερη δυνατή συμβολοσειρά

```
>>> import re
>>> x = 'From: Using the : character'
>>> y = re.findall('^F.+:', x)
>>> print(y)
['From: Using the :']
```

Γιατί όχι 'From:' ;

Ένας ή περισσότεροι χαρακτήρες

^F.+:

Ο πρώτος χαρακτήρας που θα ταιριάξει είναι το F

Ο τελευταίος χαρακτήρας που θα ταιριάξει είναι το :

Μη-Άπληστο Ταίριασμα

Δεν είναι άπληστοι όλοι οι κωδικοί επανάληψης των κανονικών εκφράσεων! Αν προσθέσετε ένα χαρακτήρα ?, το + και το * χαλαρώνουν λίγο...

```
>>> import re
>>> x = 'From: Using the : character'
>>> y = re.findall('^F.+?:', x)
>>> print(y)
['From:']
```

Ένας ή περισσότεροι χαρακτήρες αλλά μη άπληστα

```
^F.+?:
```

Ο πρώτος χαρακτήρας που θα ταιριάξει είναι το F

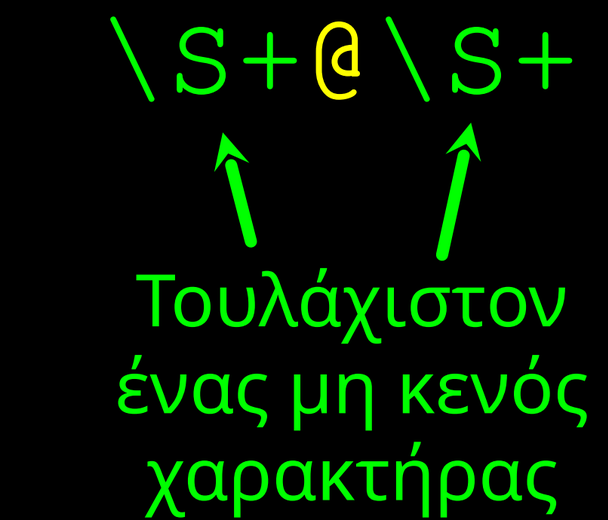
Ο τελευταίος χαρακτήρας που θα ταιριάξει είναι το :

Καλύτερη Ρύθμιση Εξαγωγής Συμβολοσειρών

Μπορείτε να βελτιώσετε την αντιστοίχιση του re.findall() και να καθορίσετε ξεχωριστά ποιο τμήμα της αντιστοίχισης πρόκειται να εξαχθεί χρησιμοποιώντας παρενθέσεις

```
From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008
```

```
>>> y = re.findall('\S+@\S+',x)
>>> print(y)
['stephen.marquard@uct.ac.za']
```

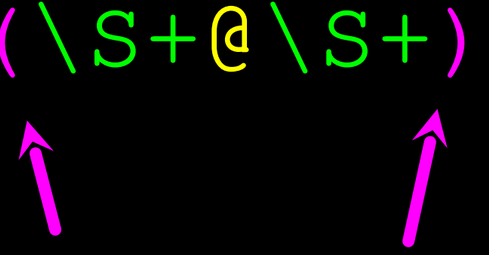


Καλύτερη Ρύθμιση Εξαγωγής Συμβολοσειρών

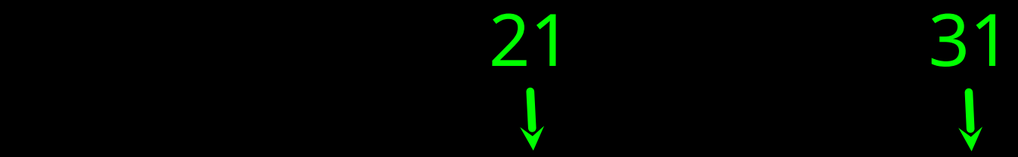
Οι Παρενθέσεις δεν αποτελούν τμήμα της αντιστοίχισης – αλλά λένε από να αρχίσει και που να σταματήσει η εξαγόμενη συμβολοσειρά

```
From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008
```

```
>>> y = re.findall('\S+@\S+',x)
>>> print(y)
['stephen.marquard@uct.ac.za']
>>> y = re.findall('^From (\S+@\S+)',x)
>>> print(y)
['stephen.marquard@uct.ac.za']
```



Παραδείγματα Ανάλυσης Συμβολοσειρών …




```
From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008
```

```
>>> data = 'From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008'
>>> atpos = data.find('@')
>>> print(atpos)
21
>>> sppos = data.find(' ',atpos)
>>> print(sppos)
31
>>> host = data[atpos+1 : sppos]
>>> print(host)
uct.ac.za
```

Εξαγωγή ονόματος κεντρικού υπολογιστή - χρησιμοποιώντας εύρεση και κατάτμηση συμβολοσειράς

Το Μοτίβο Διπλής Διάσπασης

Μερικές φορές χωρίζουμε μια γραμμή με έναν τρόπο και στη συνέχεια παίρνουμε ένα από τα κομμάτια που προέκυψαν και το χωρίζουμε ξανά

```
From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008
```

```
λέξεις = γραμμή.split()
email = λέξεις[1]
κομμάτια = email.split('@')
print(κομμάτια[1])
```

```
stephen.marquard@uct.ac.za
['stephen.marquard', 'uct.ac.za']
'uct.ac.za'
```

Η Έκδοση με Κανονική Έκφραση

```
From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008
```

```
import re
γραμμή = 'From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008'
y = re.findall('@([^ ]*)',γραμμή)
print(y)

['uct.ac.za']
```

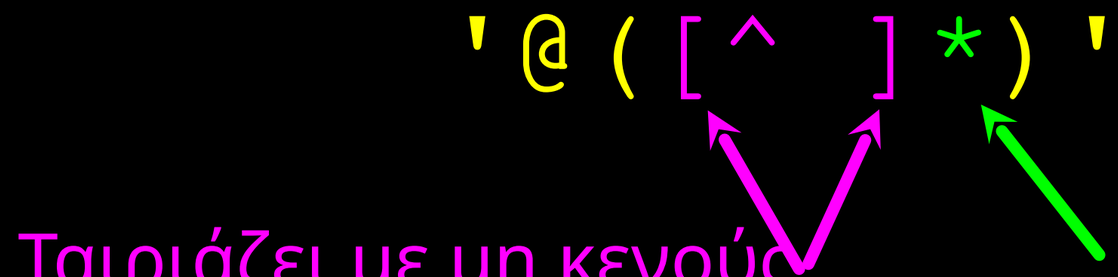
```
'@([^ ]*)'
```

Ψάξε στην συμβολοσειρά μέχρι να βρεις το σύμβολο @

Η Έκδοση με Κανονική Έκφραση

```
From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008
```

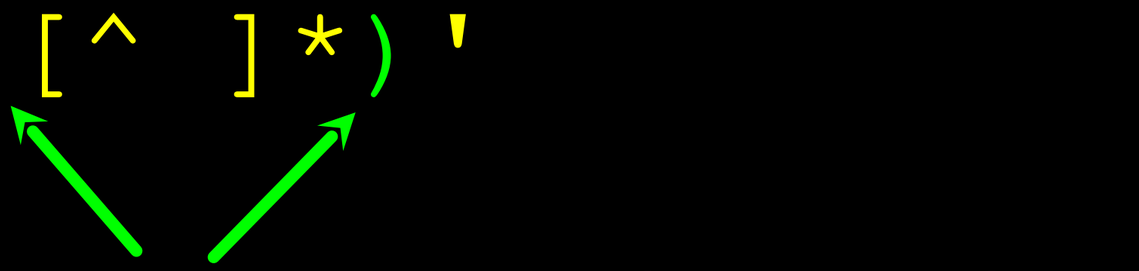
```
import re
γραμμή = 'From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008'
y = re.findall('@([^ ]*)',γραμμή)
print(y)

['uct.ac.za']
```



Η Έκδοση με Κανονική Έκφραση

```
From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008
```

```
import re
γραμμή = 'From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008'
y = re.findall('@([^ ]*)',γραμμή)
print(y)

['uct.ac.za']
```



Ακόμη Καλύτερη Regex

```
From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008
```

```
import re
γραμμή = 'From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008'
y = re.findall('^From .*@([^ ]*)',γραμμή)
print(y)

['uct.ac.za']
```

```
'^From .*@([^ ]*)'
```

Ξεκινά στην αρχή της γραμμής, ψάχνει για τη συμβολοσειρά 'From '

Ακόμη Καλύτερη Regex

```
From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008
```

```
import re
γραμμή = 'From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008'
y = re.findall('^From .*@([^ ]*)',γραμμή)
print(y)

['uct.ac.za']
```

```
'^From .*@([^ ]*)'
```

Προσπερνά ένα σωρό χαρακτήρες, ψάχνει για το σύμβολο @

Ακόμη Καλύτερη Regex

```
From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008
```

```
import re
γραμμη = 'From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008'
y = re.findall('^From .*@([^ ]*)',γραμμή)
print(y)

['uct.ac.za']
```

```
'^From .*@([^ ]*)'
```

Αρχή της εξαγωγής

Ακόμη Καλύτερη Regex

From stephen.marquard@uct.ac.za Sat Jan 5 09:14:16 2008

```
import re
γραμμή = 'From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008'
y = re.findall('^From .*@([^ ]*)',γραμμή)
print(y)

['uct.ac.za']
```

```
'^From .*@([^ ]+)'
```

Μη-κενοί χαρακτήρες

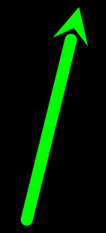
Ταιριάζει οποιονδήποτε από αυτούς

Ακόμη Καλύτερη Regex

```
From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008
```

```
import re
γραμμή = 'From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008'
y = re.findall('^From .*@([^ ]*)',γραμμή)
print(y)

['uct.ac.za']
```

```
'^From .*@([^ ]+)'
```



Τέλος εξαγωγής

Spam Confidence

```
import re
hand = open('mbox-short.txt')
λίσταααριθμών = list()
for γραμμή in hand:
    γραμμή = γραμμή.rstrip()
    διάφορα = re.findall('^X-DSPAM-Confidence: ([0-9.]+)', γραμμή)
    if len(διάφορα) != 1 :  continue
    αριθμός = float(διάφορα[0])
    λίσταααριθμών.append(αριθμός)
print('Μεγαλύτερο:', max(λίσταααριθμών))
```

```
X-DSPAM-Confidence: 0.8475
```

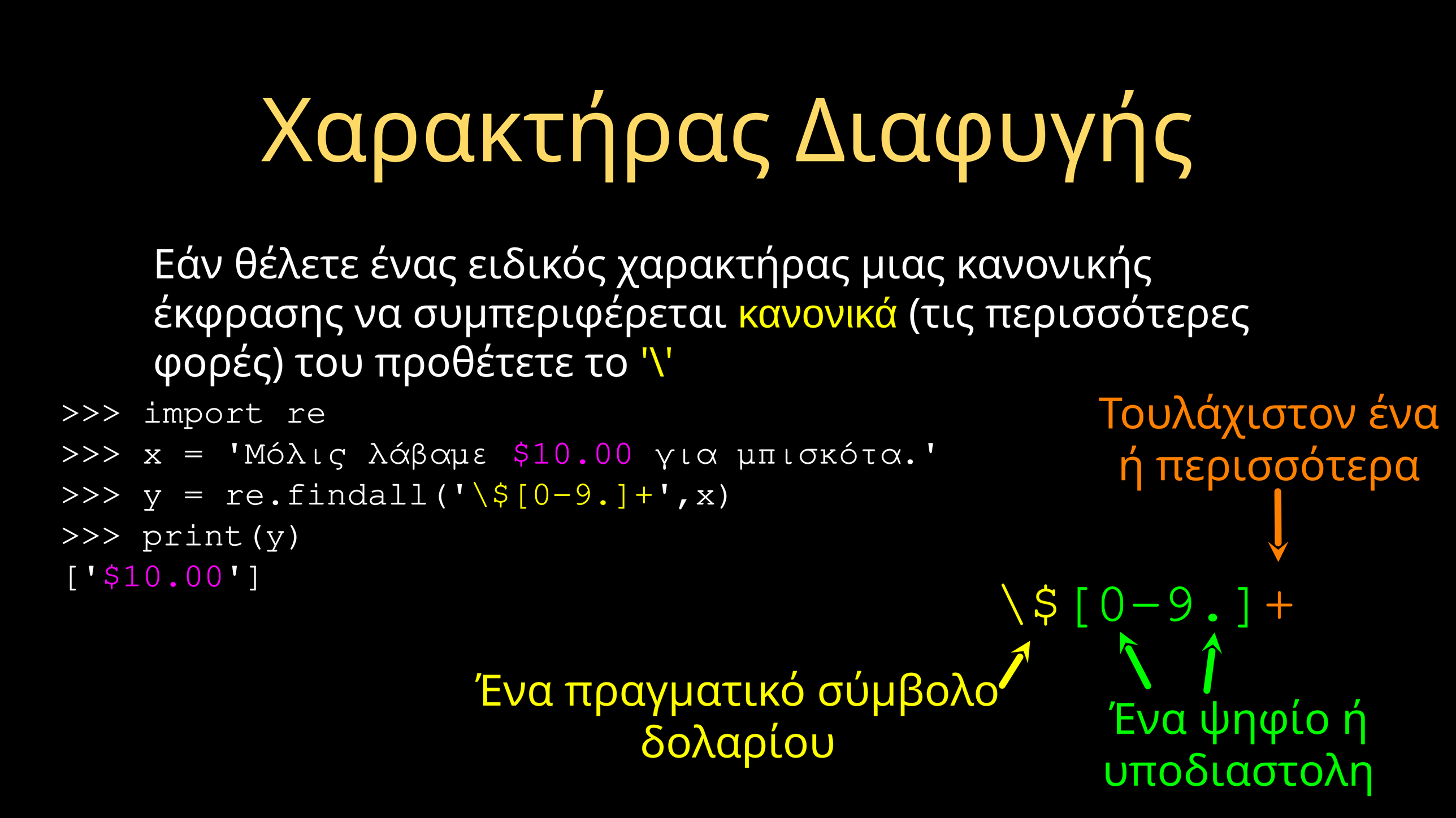
```
python ds.py
Μεγαλύτερο:
0.9907
```

Χαρακτήρας Διαφυγής

Εάν θέλετε ένας ειδικός χαρακτήρας μιας κανονικής έκφρασης να συμπεριφέρεται κανονικά (τις περισσότερες φορές) του προθέτετε το '\'

```
>>> import re
>>> x = 'Μόλις λάβαμε $10.00 για μπισκότα.'
>>> y = re.findall('\$[0-9.]+',x)
>>> print(y)
['$10.00']
```

Τουλάχιστον ένα ή περισσότερα



Σύνοψη

- Οι κανονικές εκφράσεις είναι μια αινιγματική αλλά ισχυρή γλώσσα για την αντιστοίχιση συμβολοσειρών και την εξαγωγή στοιχείων από αυτές τις συμβολοσειρές
- Οι κανονικές εκφράσεις έχουν ειδικούς χαρακτήρες που υποδηλώνουν πρόθεση

Ευχαριστίες / Συνεισφορές

Αρχική ανάπτυξη : Charles Severance, University of Michigan School of Information

Απόδοση στα Ελληνικά: Κιουρτίδου Δ. Κωνσταντία

… Εισαγάγετε νέους Μεταφραστές και άτομα που έχουν συνεισφέρει εδώ

Συνέχεια…