Σημείωσεις για το μάθημα "Προγραμματισμός σε python"

Αλέξανδρος Καντεράκης kantale@ics.forth.gr (mailto:kantale@ics.forth.gr)

Διάλεξη 7η, 1 Δεκεμβρίου 2016

Σε αυτή τη διάλεξη θα ασχοληθούμε με τη δημιουργία γραφικών παραστάσεων μέσα από τη βιβλιοθήκη matplotlib (http://matplotlib.org/)

Ο συνηθισμένος τρόπος για να εισάγουμε (import) τη matplotlib είναι:

In [3]:
```
import matplotlib.pyplot as plt
```

Στη συνέχεια για κάθε νέο plot πρέπει να δημιουργούμε τα εξής δύο αντικείμενα το fig (http://matplotlib.org/api/figure_api.html) και το ax (http://matplotlib.org/api/axes_api.html)

In [4]:
```
fig, ax = plt.subplots()
```

Γενικότερα: η matplotlib έχει τρία βασικά αντικείμενα για τον χειρισμό των plots:

- ax : Χειρισμός των αξόνων, τις περισσότερες φορές θα ασχολούμαστε με αυτό το αντικείμενο
- fig : Χειρισμός του plot ως εικόνα.
- plt : Αυτό είναι είναι το pyplot αντικείμενο το οποίο έχουμε κάνει import. Περιέχει τις βασικές μεθόδους του ax και του fig αντικειμένου. Υπάρχει σαν βοηθητικό αντικείμενο, απλούστευσης ώστε να μην χρειάζεται κάποιος να "παίζει" με δύο αντικείμενα. π.χ. μπορεί κάποιος να γράψει: `ax.plot()` ή `plt.plot()` παρομοίως μπορεί να γράψει `fig.show()` ή `plt.show()` . Παρόλα αυτά υπάρχουν μέθοδοι που είναι μέρος του ax αντικειμένου και δεν είναι του plt αντικειμένου. Και δεδομένου ότι η python είναι μία γλώσσα "There should be one-- and preferably only one --obvious way to do it." η ύπαρξη αυτών των επιλογών θεωρώ ότι είναι έξω από το πνεύμα της γλώσσας (είναι κυρίως ιστορικοί λόγοι που γίνεται αυτό). Για αυτό και δεν θα (πολύ) ασχοληθούμε με τη plt!

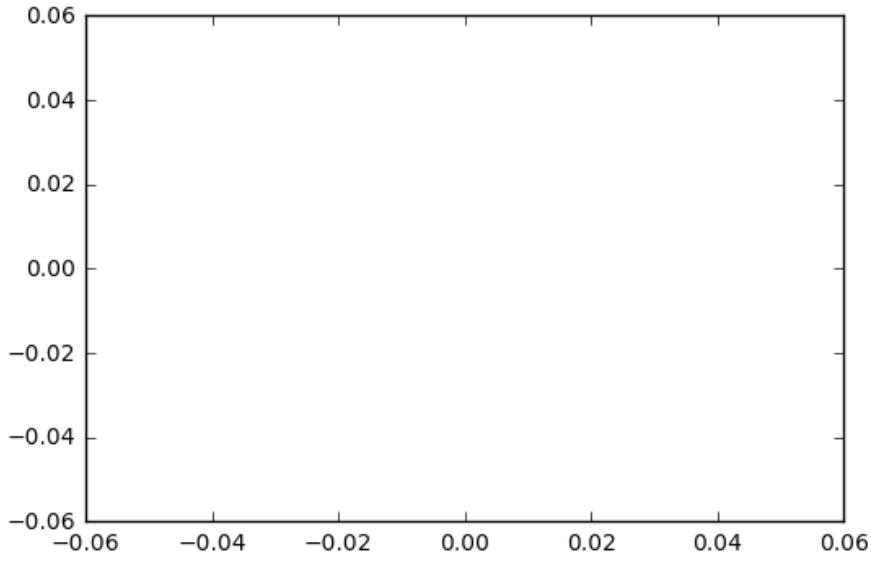
Ας φτιάξουμε ένα άδειο plot!

In [5]:
```
ax.plot()
```

Out[5]:
```
[]
```

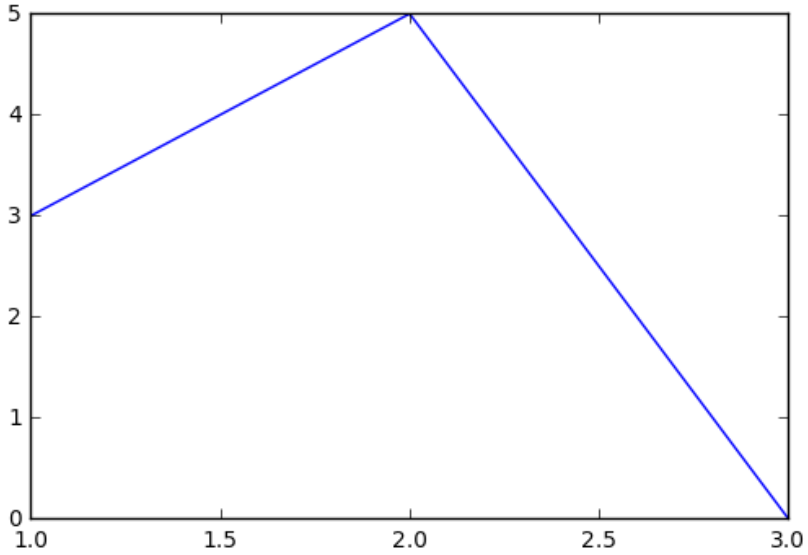
Για να εμφανίσουμε ένα plot χρησιμοποιούμε την εντολή `show()` της plt

In [7]:
```
plt.show()
```



Η `ax.plot` δέχεται μία μεγάλη ποικιλία από ορίσματα. Τα δύο πρώτα ορίσματα είναι δύο λίστες. Η πρώτη περιέχει τις συντεταγμένες στον άξονα X των στοιχείων που θέλουμε να κάνουμε plot, και η δεύτερη τις συντεταγμένες στον άξονα Y. π.χ. Για να εμφανίσουμε μία τεθλασμένη γραμμή που περνάει από τα σημεία: (1,3), (2,5), (3,0) :

In [11]:
```
fig, ax = plt.subplots()
ax.plot([1,2,3], [3,5,0])
plt.show()
```



Η plot δέχεται και ένα τρίτο όρισμα το οποίο είναι το "στυλ" της γραμμής. Αποτελείται από δύο μέρη: χρώμα και στυλ. Το χρώμα μπορεί να είναι (http://matplotlib.org/api/colors_api.html (http://matplotlib.org/api/colors_api.html)):

- b: blue
- g: green
- r: red
- c: cyan
- m: magenta
- y: yellow
- k: black
- w: white

Σε περίπτωση που θέλουμε να κάνουμε plot κάποια γραμμή τότε το στυλ μπορεί να είναι είτε ένα από (http://matplotlib.org/api/lines_api.html#matplotlib.lines.Line2D.set_linestyle (http://matplotlib.org/api/lines_api.html#matplotlib.lines.Line2D.set_linestyle)):

- '-' or 'solid' solid line
- '--' or 'dashed' dashed line
- '-.' or 'dashdot' dash-dotted line
- ':' or 'dotted' dotted line
- 'None' draw nothing
- ' ' draw nothing
- '' draw nothing

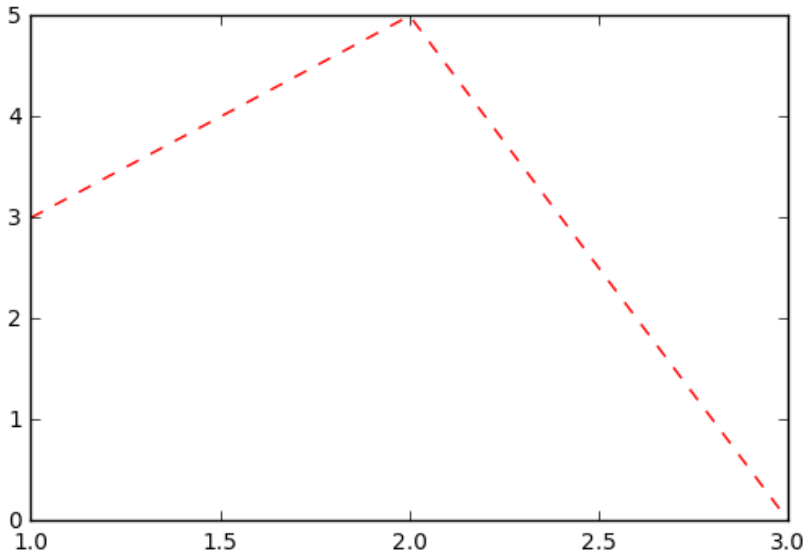
Σε περίπτωση που θέλουμε να κάνουμε plot μόνο τα σημεία (και όχι γραμμές) τότε οι επιλογές είναι (http://matplotlib.org/api/markers_api.html (http://matplotlib.org/api/markers_api.html)):

- ”.” point
- ”,” pixel
- “o” circle
- “v” triangle_down
- “^” triangle_up
- “<” triangle_left
- “>” triangle_right
- “1” tri_down
- “2” tri_up
- “3” tri_left
- “4” tri_right
- “8” octagon
- “s” square
- “p” pentagon
- “*” star
- “h” hexagon1
- “H” hexagon2
- “+” plus
- “x” x
- “D” diamond
- “d” thin_diamond
- “|” vline
- “_” hline
- TICKLEFT tickleft
- TICKRIGHT tickright
- TICKUP tickup
- TICKDOWN tickdown
- CARETLEFT caretleft
- CARETRIGHT caretright
- CARETUP caretup
- CARETDOWN caretdown
- “None” nothing
- None nothing
- ” “ nothing
- “” nothing

Υπάρχουν περισσότερες επιλογές και δυνατότητες για "καστομιές"¨: http://matplotlib.org/api/markers_api.html (http://matplotlib.org/api/markers_api.html)

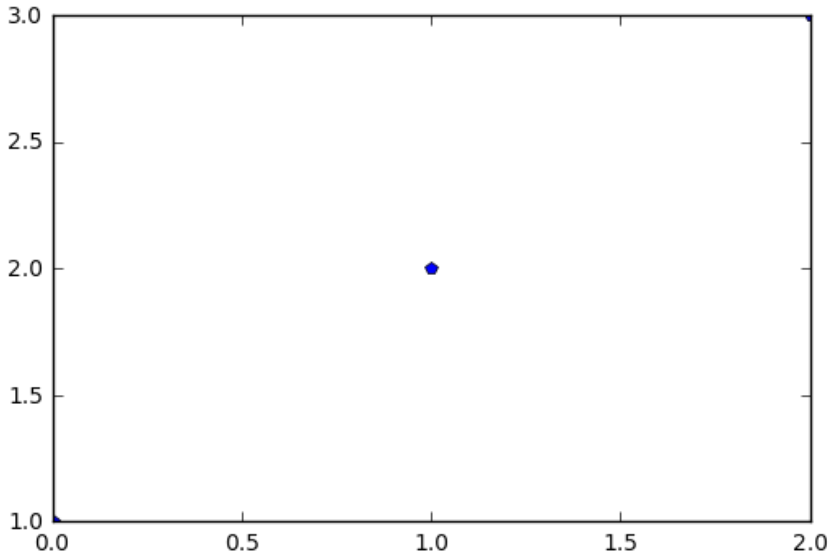
Οπότε για να πλοτάρουμε μία κόκκινη διακεκομένη γραμμή:

In [14]:
```
fig, ax = plt.subplots()
ax.plot([1,2,3], [3,5,0], 'r--')
plt.show()
```



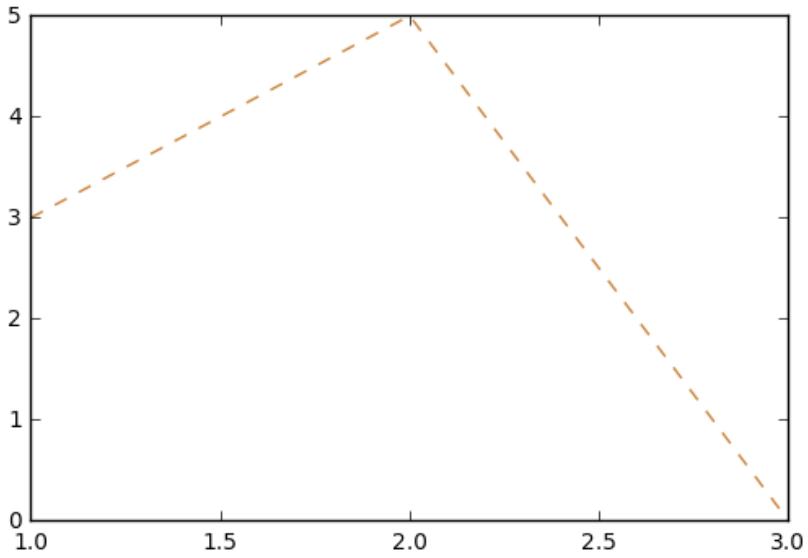
Για να πλοτάρουμε μπλε πεντάγωνα:

In [18]:
```
fig, ax = plt.subplots()
ax.plot([1,2,3], 'bp') # ΠΡΟΣΟΧΗ! εδώ πλοτάρουμε (1,1), (2,2), (3,3). bp: b=blu
e, p=pentagon
plt.show()
```



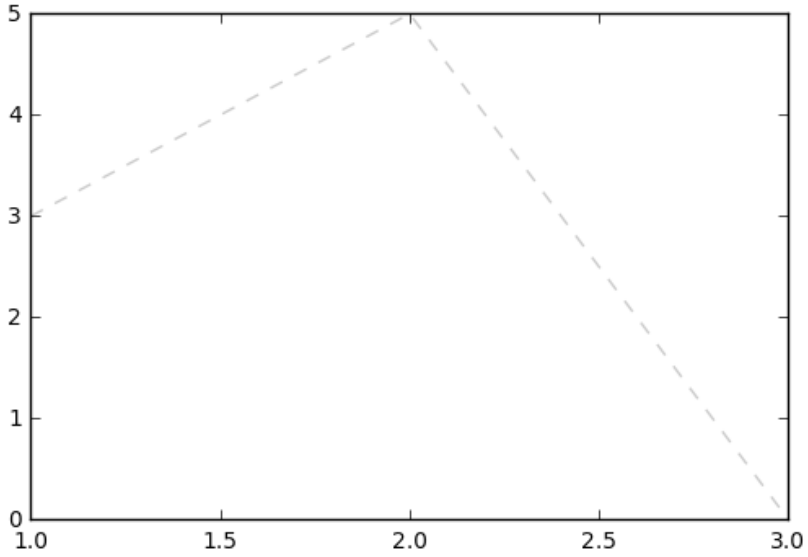
Φυσικά υπάρχει πιο πλούσιος τρόπος να ορίσουμε ένα χρώμα με την παράμετρο "c"

In [19]:
```
fig, ax = plt.subplots()
ax.plot([1,2,3], [3,5,0], '--', c="peru")
plt.show()
```



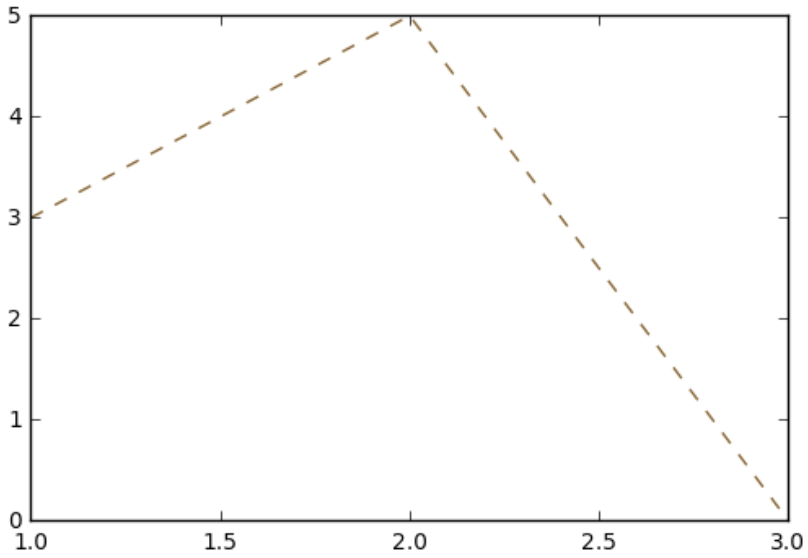
Ναι, υπάρχει χρώμα που λέγεται "peru". Πλήρη λίστα με ονόματα υπάρχει εδώ: (http://matplotlib.org/examples/color/named_colors.html (http://matplotlib.org/examples/color/named_colors.html)) . Εναλλακτικά μπορείτε να χρησιμοποιείσετε μία τιμή από το 0.0 μέχρι το 1.0 για να τυπώσετε σε ένα "grayscale" όπου το 0.0 είναι το μαύρο και το 1.0 είναι το άσπρο:

In [20]:
```
fig, ax = plt.subplots()
ax.plot([1,2,3], [3,5,0], '--', c="0.8")
plt.show()
```



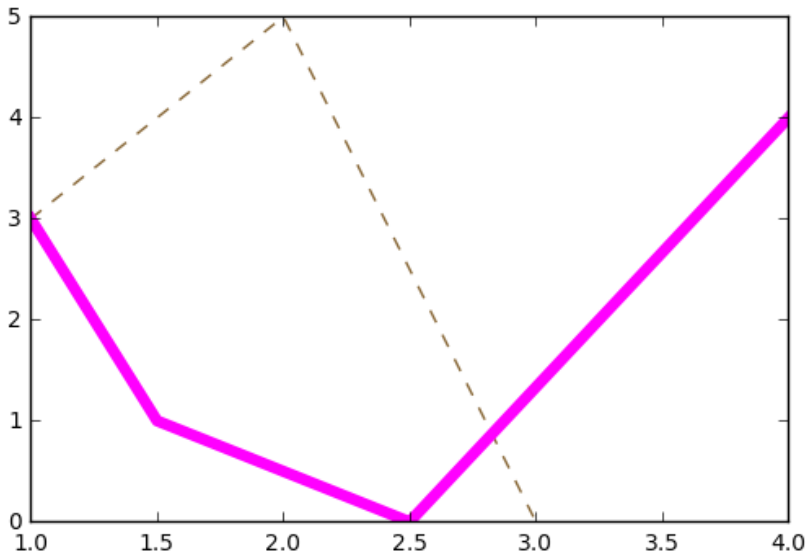
Φυσικά μπορείτε να χρησιμοποιήσετε ένα οποιοδίποτε RGB χρώμα (https://en.wikipedia.org/wiki/RGB_color_model). Υπάρχουν πολλά sites που μπορείς να επιλέξεις ένα χρώμα π.χ: http://htmlcolorcodes.com/ (http://htmlcolorcodes.com/)

In [21]:
```
fig, ax = plt.subplots()
ax.plot([1,2,3], [3,5,0], '--', c="#876635")
plt.show()
```



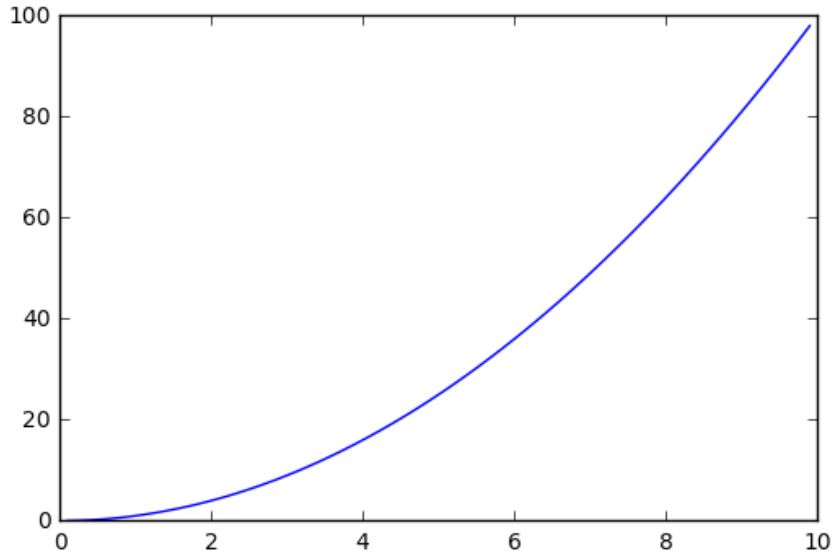
Μπορούμε να χρησιμοποιήσουμε πολλές φορές τη plot:

In [24]:
```
fig, ax = plt.subplots()
ax.plot([1,2,3], [3,5,0], '--', c="#876635")
ax.plot([1, 1.5, 2.5, 4], [3,1, 0, 4], '-', c="magenta", linewidth=5) # Paxos = 5
plt.show()
```



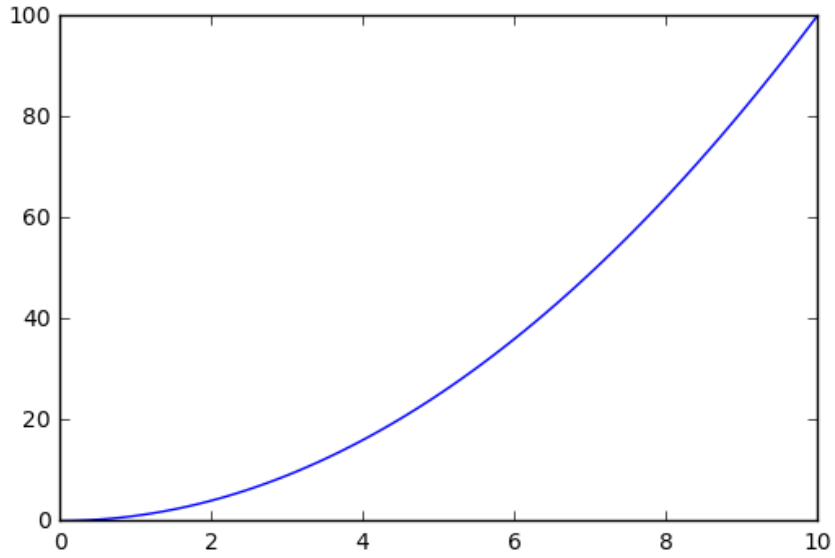
In [ ]:
```
Μπορούμε επίσης να κάνουμε plot μία συνάρτηση υπολογίζοντας τα X και τα Y της:
```

In [25]:
```
def f(x):
    return x**2 # X square

X = [x/10.0 for x in range(1,100)]
Y = [f(x) for x in X]

fig, ax = plt.subplots()
ax.plot(X,Y)
plt.show()
```



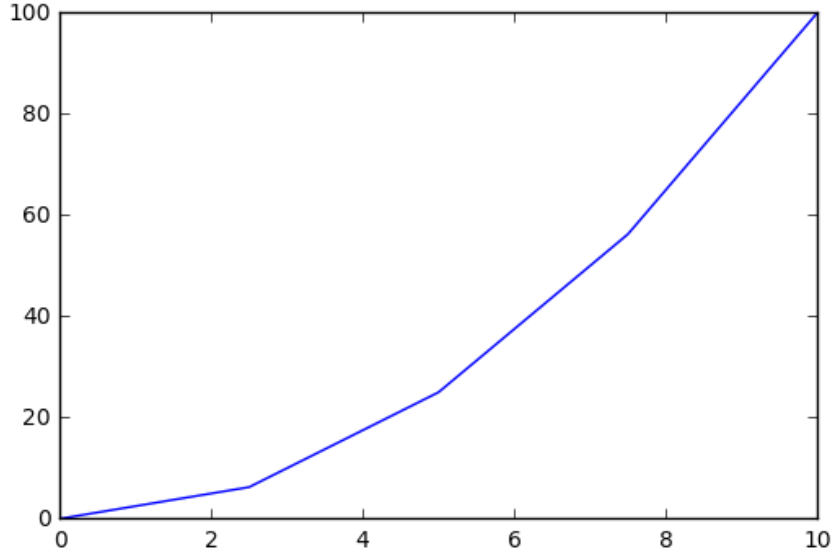
Ένας καλύτερος τρόπος για να πλοτάρουμε συναρτήσεις είναι να χρησιμοποιήσουμε τη linspace (https://docs.scipy.org/doc/numpy/reference/generated/numpy.linspace.html) της numpy. Η linspace(a,b,c) δημιουργεί μία αριθμητική πρόοδο από a μέχρι b, έτσι ώστε να υπάρχουν συνολικά c στοιχεία.

In [27]:
```
import numpy as np

X = np.linspace(0, 10, 100) # 0 = min X, 10 = max X, 100 = resolution...
Y = [f(x) for x in X]

fig, ax = plt.subplots()
ax.plot(X,Y)

plt.show()
```



Το 100 στη linspace μπορούμε να το δούμε και ως την "ανάλυση της γραφικής παράστασης"

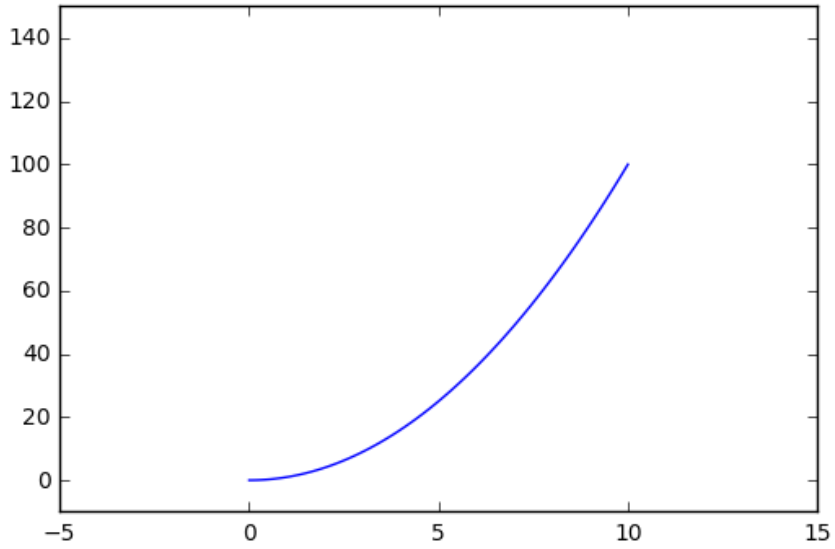
In [30]:
```
X = np.linspace(0, 10, 5) # 5 = resolution
Y = [f(x) for x in X]

fig, ax = plt.subplots()
ax.plot(X,Y)

plt.show()
```



Παρατηρήστε πόσο "σπαστή" είναι η γραφική παράσταση

Με τη συνάρτηση ax.set_xlim, ax.set_ylim αλλάζουμε τα όρια των αξόνων

In [32]:
```
fig, ax = plt.subplots()

X = np.linspace(0, 10, 100) # 0 = min X, 10 = max X, 100 = resolution...
Y = [f(x) for x in X]

ax.set_xlim(-5, 15)
ax.set_ylim(-10, 150)
ax.plot(X,Y)

plt.show()
```



Μπορούμε επίσης να βάλουμε labels στους άξονες και σε όλο το πλοτ:

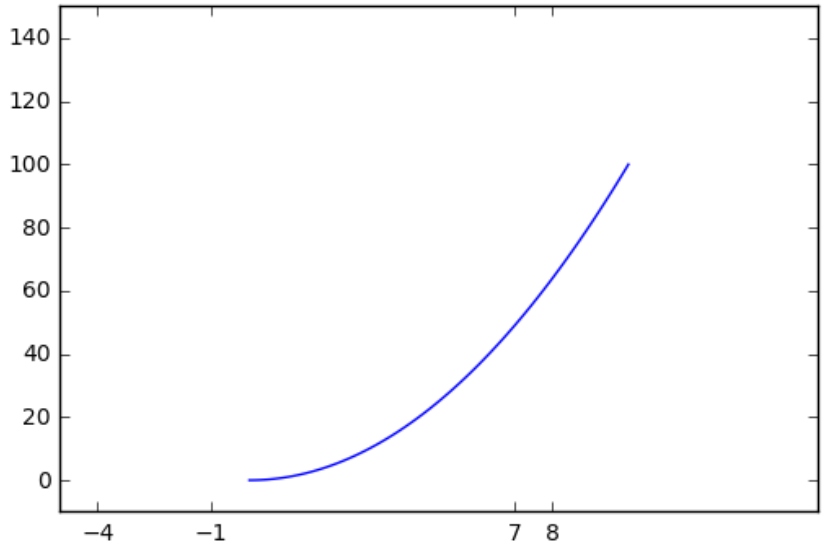
In [33]:
```
fig, ax = plt.subplots()

X = np.linspace(0, 10, 100) # 0 = min X, 10 = max X, 100 = resolution...
Y = [f(x) for x in X]

ax.set_xlim(-5, 15)
ax.set_ylim(-10, 150)
ax.plot(X,Y)

ax.set_xlabel("Genes")
ax.set_ylabel("Unicorns")
ax.set_title("Number of Unicorns in Genes")


plt.show()
```

Μπορείτε να ορίσετε μέγεθος, γραμματοσειρά και στυλ στα labels

In [36]:
```
fig, ax = plt.subplots()

X = np.linspace(0, 10, 100) # 0 = min X, 10 = max X, 100 = resolution...
Y = [f(x) for x in X]

ax.set_xlim(-5, 15)
ax.set_ylim(-10, 150)
ax.plot(X,Y)

ax.set_xlabel("Genes")
ax.set_ylabel("Unicorns")
ax.set_title("Number of Unicorns in Genes", fontname="Comic Sans MS", fontsize=
20)


plt.show()
```

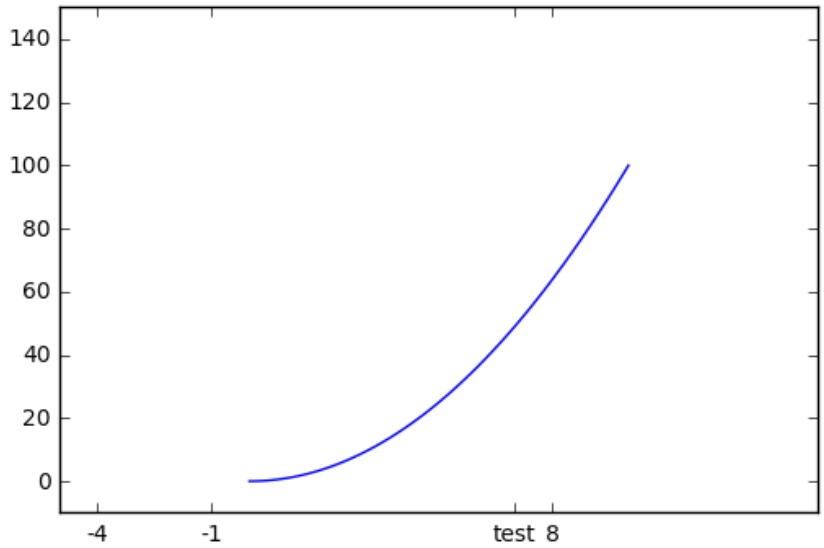
Με τη ax.set_xticks μπορείτε να ορίσεττε εσείς ποια ticks θα φαίνονται σε κάθε άξονα

In [38]:
```
fig, ax = plt.subplots()
ax.set_xlim(-5, 15)
ax.set_ylim(-10, 150)
ax.plot(X,Y)
ax.set_xticks([-4,-1,7,8])
plt.show()
```



Μπορείτε να αλλάξετε και το label του tick:

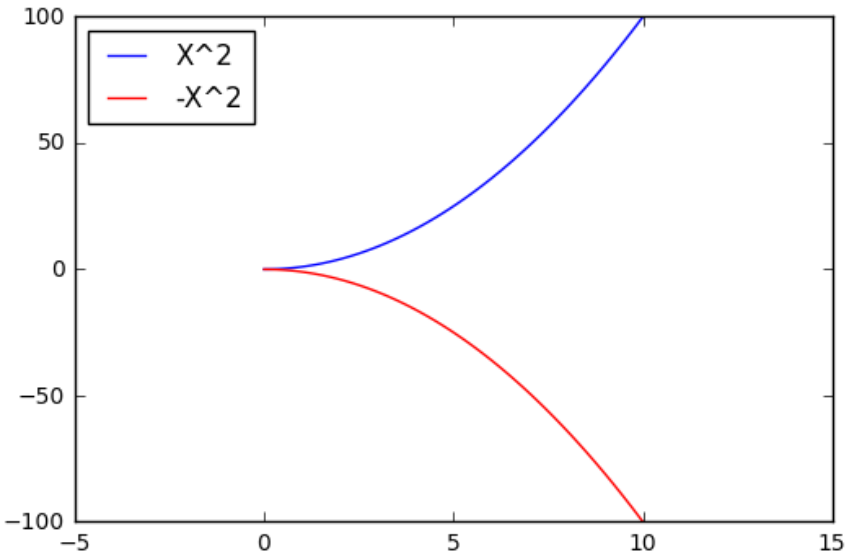
In [45]:
```
fig, ax = plt.subplots()
ax.set_xlim(-5, 15)
ax.set_ylim(-10, 150)
ax.plot(X,Y)
ax.set_xticks([-4,-1,7,8])

ticks = ax.get_xticks().tolist()
ticks[2] = "test"
ax.set_xticklabels(ticks)

plt.show()
```



Η συνάρτηση plot επιστρέφει έναν πίνακα από legends. Αυτά τα legends μπορούμε αν θέλουμε να να τα προσθέσουμε στο plot

In [62]:
```
fig, ax = plt.subplots()
ax.set_xlim(-5, 15)
ax.set_ylim(-100, 100)
legends = ax.plot(X,Y, 'b', X, [-y for y in Y], 'r')
print (legends)
plt.legend(legends, ["X^2", "-X^2"], loc=2) # loc=2 shmainei panw aristera
plt.show()
```

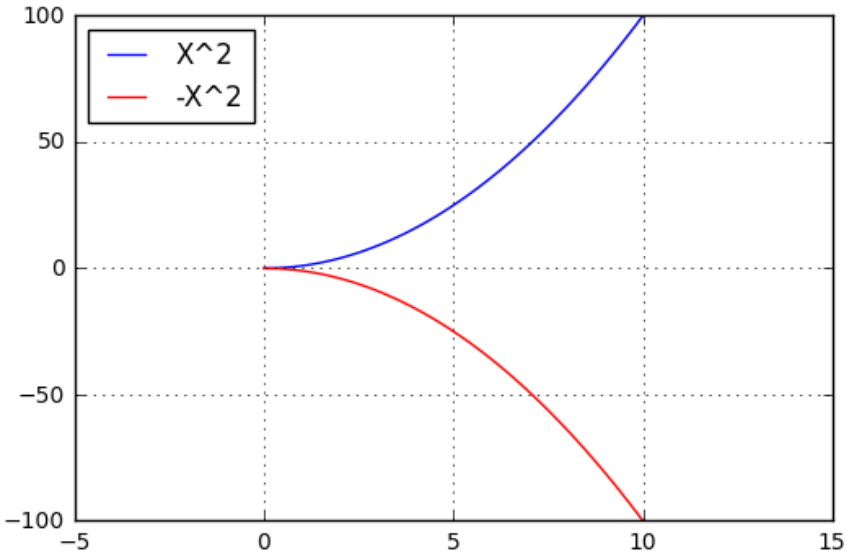
```
[<matplotlib.lines.Line2D object at 0x10e3c2208>, <matplotlib.lines.Line2D obj
ect at 0x10df87d68>]
```



Περισσότερα για το loc (location του legend δείτε εδώ: http://matplotlib.org/api/legend_api.html#matplotlib.legend.Legend (http://matplotlib.org/api/legend_api.html#matplotlib.legend.Legend))

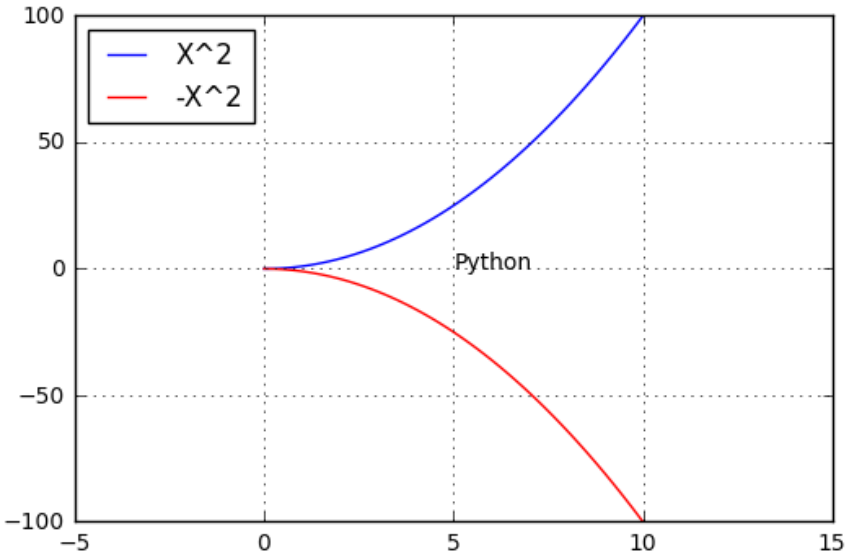
Επίσης μπορούμε να προσθέσουμε ενα grid:

In [64]:
```
fig, ax = plt.subplots()
ax.set_xlim(-5, 15)
ax.set_ylim(-100, 100)
legends = ax.plot(X,Y, 'b', X, [-y for y in Y], 'r')

ax.grid(True)

plt.legend(legends, ["X^2", "-X^2"], loc=2) # loc=2 shmainei panw aristera
plt.show()
```



Προσθέστε ένα κείμενο στο σημείο X,Y :

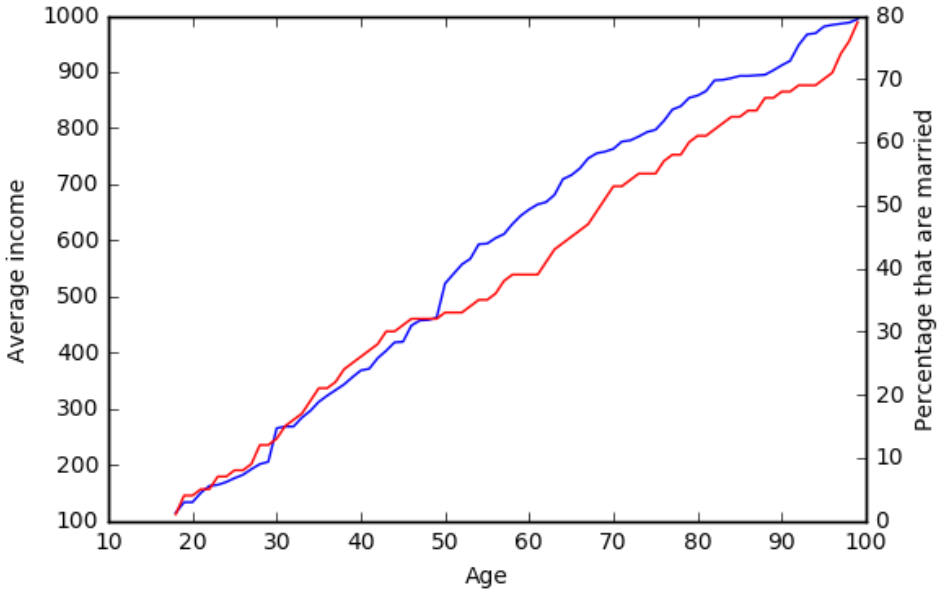
In [65]:
```
fig, ax = plt.subplots()
ax.set_xlim(-5, 15)
ax.set_ylim(-100, 100)
legends = ax.plot(X,Y, 'b', X, [-y for y in Y], 'r')
ax.grid(True)

ax.text(5,0,"Python")

plt.legend(legends, ["X^2", "-X^2"], loc=2) # loc=2 shmainei panw aristera
plt.show()
```



Επίσης υπάρχει η annotate (http://matplotlib.org/users/annotations_guide.html#plotting-guide-annotation) με την οποία μπορείτε να βάλετε βελάκια

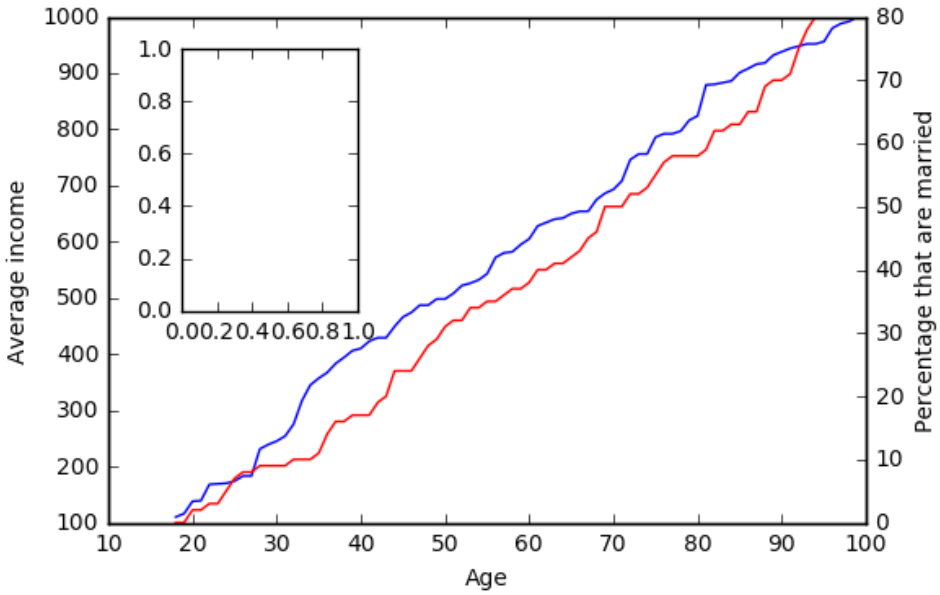
Πολλές φορές θέλουμε να τυπώσουμε δύο plots τα οποία να μοιράζονται τον ίδιο άξονα. Ας υποθέσουμε π.χ. ότι θέλουμενα μοιράζονται το άξονα X :

In [70]:
```
fig, ax = plt.subplots()
import random

age = [x for x in range(18,100)]
income = sorted([random.randint(100,1000) for x in age])
percentage_married = sorted([random.randint(0, 80) for x in age])

legends_income = ax.plot(age, income, 'b')

# Dhmiourgoume ena antigrafo tou axona X
ax_new = ax.twinx()

# Plotaroume to deutero plot panw se auto
legends_married = ax_new.plot(age, percentage_married, 'r')

ax.set_xlabel("Age")
ax.set_ylabel("Average income")
ax_new.set_ylabel("Percentage that are married")

plt.show()
```

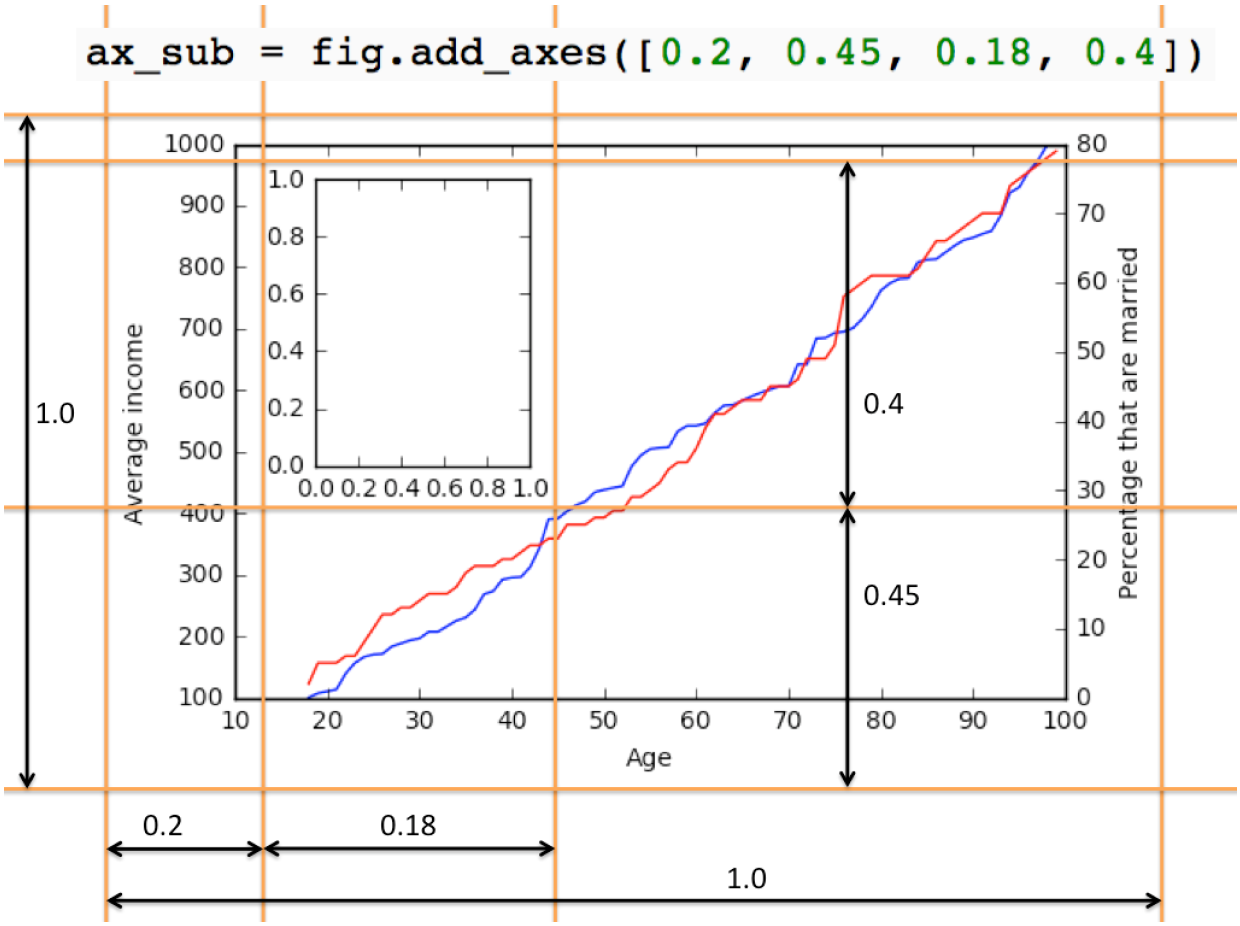


Επίσης μπορούμε να προσθέσουμε ένα ολόκληρο νέο πλότ μέσα σε ένα παλιό! Αυτό το κάνουμε με την εντολή fig.add_axes() (http://matplotlib.org/api/figure_api.html#matplotlib.figure.Figure.add_axes). Η add_axes **ΑΓΝΟΕΙ** το μέγεθος των αξόνων (π.χ. ο X έχει μέγεθος από 10 μέχρι 100 παραπάνω). Αντίθετα θεωρεί ότι ΟΛΟ το πλοτ είναι ένα καρτεσιανό γινόμενο [0,1]X[0,1] . Με την add_axes ορίζουμε τις διαστάσεις του νέου plot πάνω στο παλιό. Παράδειγμα

In [82]:
```
fig, ax = plt.subplots()
import random

age = [x for x in range(18,100)]
income = sorted([random.randint(100,1000) for x in age])
percentage_married = sorted([random.randint(0, 80) for x in age])

legends_income = ax.plot(age, income, 'b')

# Dhmiourgoume ena antigrafo tou axona X
ax_new = ax.twinx()

# Plotaroume to deutero plot panw se auto
legends_married = ax_new.plot(age, percentage_married, 'r')

ax.set_xlabel("Age")
ax.set_ylabel("Average income")
ax_new.set_ylabel("Percentage that are married")


# Φτιάχνουμε νέο υπο-πλοτ:
ax_sub = fig.add_axes([0.2, 0.45, 0.18, 0.4])


plt.show()
```

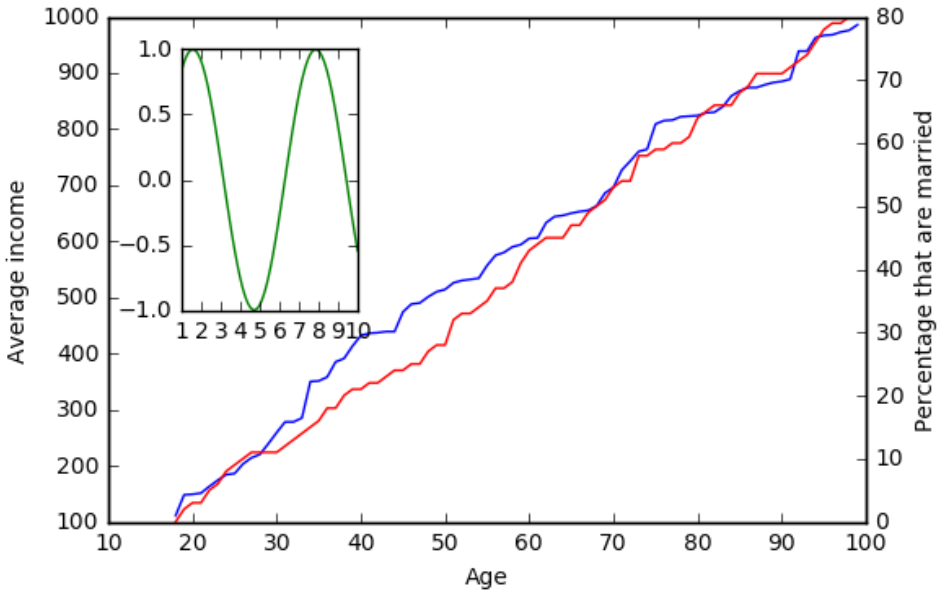


Η σημασιολογία των παραμέτρων της `add_axes` φαίνεται στο παρακάτω σχήμα:



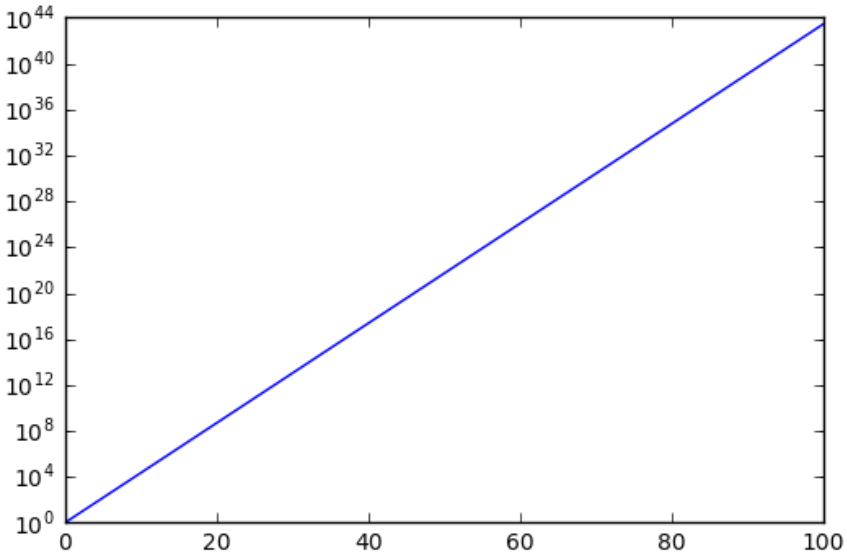
Ας πλοτάρουμε μέσα στο sub-plot:

In [85]:
```
fig, ax = plt.subplots()
import random

age = [x for x in range(18,100)]
income = sorted([random.randint(100,1000) for x in age])
percentage_married = sorted([random.randint(0, 80) for x in age])

legends_income = ax.plot(age, income, 'b')

# Dhmiourgoume ena antigrafo tou axona X
ax_new = ax.twinx()

# Plotaroume to deutero plot panw se auto
legends_married = ax_new.plot(age, percentage_married, 'r')

ax.set_xlabel("Age")
ax.set_ylabel("Average income")
ax_new.set_ylabel("Percentage that are married")


# Φτιάχνουμε νέο υπο-πλοτ:
ax_sub = fig.add_axes([0.2, 0.45, 0.18, 0.4])
sub_X = np.linspace(1,10,100)
sub_Y = np.sin(sub_X)
ax_sub.plot(sub_X,sub_Y, 'g')

plt.show()
```



Μπορούμε επίης να αλλάξουμε τη κλίμακα των αξόνων σε λογαριθμική:

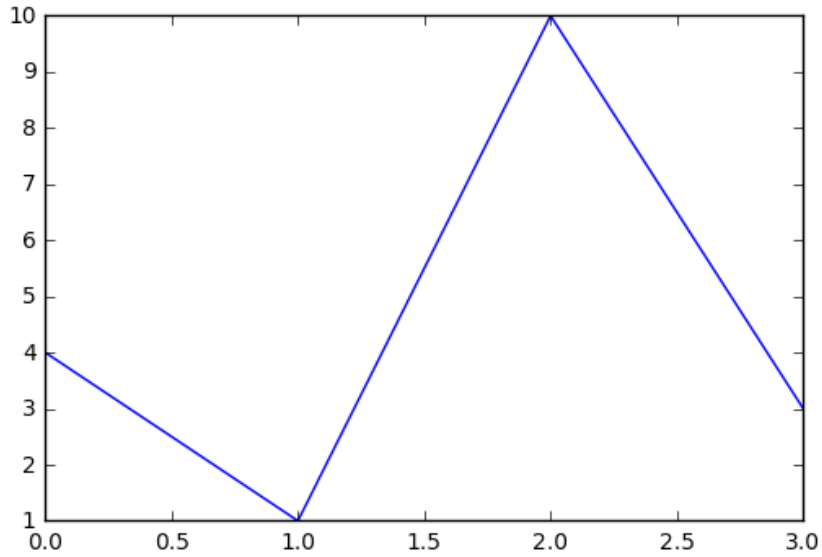
In [88]:
```
fig, ax = plt.subplots()

X = np.linspace(0,100,1000)
Y = np.exp(X)

ax.plot(X,Y)

ax.set_yscale("log")
plt.show()
```



Μπορούμε να σώσουμε στο δίσκο ένα πλοτ μέσω της "plt.savefig()". Ανάλογα με την κατάληξη που θα βάλετε στο όνομα του αρχείου, θα το σώσει και σε διαφορετικό format. **ΠΡΟΣΟΧΗ!** τη savefig πρέπει να τη καλείτε ΠΡΙΝ τη plt.show()

In [87]:
```
fig, ax = plt.subplots()

ax.plot([4,1,10,3])

plt.savefig("figure.png")
plt.savefig("figure.jpg")
plt.savefig("figure.eps")
plt.savefig("figure.tiff")
plt.savefig("figure.pdf")

plt.show()
```

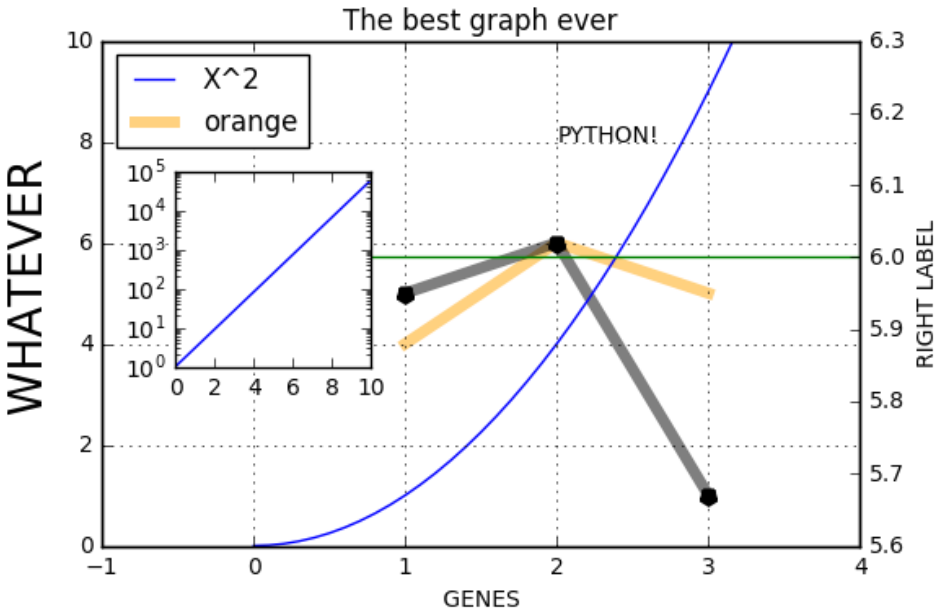
```
<matplotlib.figure.Figure at 0x10e54dd68>
```



Και το παράδειγμα που δουλέψαμε στη διάλεξη:

In [89]:
```
def x2(x):
    return x*x

fig, ax = plt.subplots()
leg_orange, = ax.plot([1,2,3], [4,6,5,], '-', color="orange", lw="5", alpha=0.
5)
ax.plot([1,2,3], [5,6,1,], '-', color="black", lw="5", alpha=0.5)
ax.plot([1,2,3], [5,6,1,], '*', color="black", lw="5", mew=5)
X = [x/10.0 for x in range(0,100)]
Y = [x2(x) for x in X]
leg_x2, = ax.plot(X, Y)

ticks = ax.get_xticks()
print (ticks)
#ticks = [str(x) + " a " for x in ticks]
# ax.set_xticklabels(ticks)
#ax.set_xticks([x for x in range(-1,5, 2)])
ax.set_xlabel(" GENES ")
ax.set_ylabel(" WHATEVER", fontsize=20)
ax.set_title("The best graph ever")
ax.grid(True)

ax_new = ax.twinx()
ax_new.plot([0,4], [6,6], color="green")
ax_new.set_ylabel("RIGHT LABEL")

ax.set_xlim(-1,4)
ax.set_ylim(0,10)

ax.text(2, 8, "PYTHON!")

plt.legend([leg_x2, leg_orange], ["X^2", "orange"], loc=2)
ax2 = fig.add_axes([0.2, 0.4, 0.2, 0.3])
X2 = [x/10.0 for x in range(0,100)]
Y2 = [3**x for x in X2]
ax2.set_yscale("log")

ax2.plot(X2, Y2)

plt.savefig("figure.png")
plt.savefig("figure.jpg")
plt.savefig("figure.eps")
plt.savefig("figure.tiff")
plt.show()
```

```
[  0.   2.   4.   6.   8.  10.]
```



Κάποιες τελευταίες σημειώσεις:

- Η matplotlib είναι τεράστια! Ελέγξτε πόσο διαφορετικά plots μπορείτε να φτιάξετε: http://matplotlib.org/gallery.html (http://matplotlib.org/gallery.html) .
- Υπάρχουν βιβλιοθήκες για plot-άρισμα συγκεκριμμένου τύπου δεδομένων οι οποίες βασίζονται στη matplotlib. Μία από τις καλύτερες (IMHO) είναι η seaborn (http://seaborn.pydata.org/) η οποία προσανατολίζεται σε πλοτάρισμα κατανομών.
- Η matplotlib έχει "κατηγορηθεί" ότι είναι καλή μεν αλλά δεν παράγει publication ready plots. Δηλαδή αισθητικά θέλουν λίγο δουλειά ακόμα πριν μπουν σε ένα paper. Διαβάστε σχετικά: https://github.com/jbmouret /matplotlib_for_papers (https://github.com/jbmouret/matplotlib_for_papers) , http://blog.dmcdougall.co.uk /publication-ready-the-first-time-beautiful-reproducible-plots-with-matplotlib/ (http://blog.dmcdougall.co.uk /publication-ready-the-first-time-beautiful-reproducible-plots-with-matplotlib/) , http://www.reddit.com /r/bioinformatics/comments/2cnv9g/polishing_your_python_matplotlib_plots_for/ (http://www.reddit.com /r/bioinformatics/comments/2cnv9g/polishing_your_python_matplotlib_plots_for/) , http://nipunbatra.github.io /2014/08/latexify/ (http://nipunbatra.github.io/2014/08/latexify/)
- Η matplotlib ανοίκει στη κατηγορία των μη-interactive plots. Δλδ δεν μπορείς να αλληλεπιδράσεις με το plot (αν και στη τελευταία έκδοση έχουν προσθέσει κάποιες βασικές δυνατότητες). Φτιάχνει plots για παρουσιάσεις / papers και όχι για visual exploration! Στη 2η κατηγορία ανοίκουν άλλες βιβλιοθήκες όπως η bokeh (http://bokeh.pydata.org /en/latest/) και η plot.ly (https://plot.ly/). Αν υπάρξει λαϊκή απαίτηση μπορούμε να κάνουμε μία διάλεξη για αυτές!
- Το plot-άρισμα είναι τέχνη! Όσο καλή και αν είναι μία βιβλιοθήκη πρέπει κάποιος να γνωρίζει βασικά θέματα γραφιστικής / αισθητικής για να κάνει ένα καλό plot: http://journals.plos.org/ploscompbiol/article?id=10.1371 /journal.pcbi.1003833 (http://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1003833)

In [ ]: