Python kurssi 24.8.-30.9.2015 Kaisa Sarén

Harkat:

Harjoitus 2.

Kuvaa yleisellä tasolla ohjelma, joka lukee kaksi genomia ja tulostaa parhaiten keskenään samanlaiset avoimet lukukehykset .
– Mitä tietoja ohjelma tarvitsee syötteenään?
Ohjelma tarvitsee kaksi genomia, aloituskodonit
– Minkälaisen tuloksen ohjelman pitäisi tuottaa / mitä ohjelman pitäisi tehdä?
1. Ohjelma etsii molemmista genomeista avoimet lukukehykset ja tallentaa ne esim. listaksi
2. Ohjelma ottaa genomista A avoimen lukukehyksen ja vertaa sitä B:n ensimmäiseen ORF:iin. Rinnastus pisteytetään kertomaan vastaavuudesta jollakin sopivalla tavalla (taulukkoon, Mappiin?)
– Mitä erillisiä (toistettavia) kokonaisuuksia ohjelmassa on?
3. Kohta 2 tehdään jokaiselle A:n ORF:lle, kunnes listan kaikki ORF:t on käyty läpi.
– Mitä ohjelmassa voi mennä pieleen? Miten asiaan pitää suhtautua?
Virheitä esim.:
- indeksointivirheet (sekoittavat tuloksen tai ylittävät rajat->kaatuu)
- ikuiset silmukat
- syötteet väärin muotoiltu
- väärät/puutteelliset orf:t
- huonosti sopiva pisteytysmatriisi

Harjoitus3.

```
# Kaisa Saren, Aug 26th, 2015
# Python-course; make a following drawing using print-commands.
# *-------*
# |       |
# |       |
# |       |
# *-------*

print('*-------*')
i = 0
while i < 3:
    print('|\t|')
    i += 1
print('*-------*')
```

Harjoitus 4.

```
# Kaisa Saren, Aug 26th, 2015
# Python-course
# Tee ohjelma, joka kysyy lukua ja tulostaa sen neliön
```

```
# Muista muuttaa merkkijono luvuksi float-komennolla

luku = float(input('Syötä luku josta haluat neliön: '))
print(luku**2)
```

Harjoitus 5.

```
# Kaisa Saren, Aug 26th, 2015
# Python-course, Helsinki Open University
# Tee ohjelma, joka kysyy kahta lukua ja tulostaa
# lukujen keskiarvon. Kysy lukuja yksi kerrallaan.

lukujen_summa = 0;
count = 0;
lukujen_summa += (float(input('Tulostan lukujen keksiarvon. Anna ensimmäinen luku: ')))
count += 1
lukujen_summa += (float(input('Anna toinen luku: ')))
count += 1
print('Keskiarvo on ' + str(lukujen_summa/count))
```

Harjoitus 6.

```
# Kaisa Saren, Aug 26th, 2015
# Python-course, Helsinki Open University
# Tee ohjelma, joka kysyy käyttäjältä kolme lukua (voivat olla
# missä järjestyksessä tahansa) ja tulostaa kahden suurimman luvun
# summan. Tulosteet luvuilla 1, 2, 3 ja 10, 11, 11.

luvut = []
luvut.append(float(input('Anna kolme lukua, tulostan kahden '
                + 'suurimman summan. Ensimmäinen luku: ')))
luvut.append(float(input('Toinen luku: ')))
luvut.append(float(input('Kolmas luku: ')))
jarj_luvut = (sorted(luvut))
print(jarj_luvut[-2] + jarj_luvut[-1])
```

Harjoitus 7.

```
# Kaisa Saren, Aug 26th, 2015
# Python-course, Helsinki Open University
# Tee ohjelma, joka kysyy käyttäjältä numeroa ja kertoo, onko luku
# parillinen vai pariton. Käytä if- ja else-komentoja, ”%”-operaattori
# antaa kahden luvun jakojäännöksen.
```

```
luku = int(input('Parillisuustesti, anna kokonaisluku :'))
jakojaannos = luku % 2
if jakojaannos == 0:
    print('Luku ' + str(luku) + ' on parillinen')
elif jakojaannos == 1:
    print('Luku ' + str(luku) + ' on pariton')
else: print('Jotain meni pieleen...')
```

Harjoitus 8.

```
# Kaisa Saren, Aug 26th, 2015
# Python-course, Helsinki Open University
# Tee ohjelma, joka kysyy käyttäjältä kolme lukua (voivat olla
# missä järjestyksessä tahansa),vähentää suurimmasta 5 ja lisää pienimpään
# lukuun 8 ja lopulta tulostaa muuttuneet luvut pienimmästä suurimpaan.
# Mitä ohjelma tulostaa luvuilla: 2, 4, 8 ja 10, 11, 11?

luvut = []
luvut.append(float(input('Anna kolme lukua, tulostan kahden '
            + 'suurimman summan. Ensimmäinen luku: ')))
luvut.append(float(input('Toinen luku: ')))
luvut.append(float(input('Kolmas luku: ')))
jarj_luvut = (sorted(luvut))
print(jarj_luvut)
jarj_luvut[-1] -= 5
jarj_luvut[0] += 8
print(sorted(jarj_luvut))       #2,4,8 -> [3.0, 4.0, 10.0]
                                #10,11,11 -> [6.0, 11.0, 18.0]
```

Harjoitus 9.

```
# Kaisa Saren, Aug 27th, 2015
# Python-course, Helsinki Open University

eksoni = 'TATTGCCC' + 'ATGCCAATTG'
print(len('TATTGCCCATGCCAATTG'))
print('TATTGCCCATGCCAATG'.find('ATG'))
if 'ATG' in 'TATTGCCCATGCCAATTG' :
    print('Löytyi!')
# 18 (pituus)
# 8 (alk. indeksistä 8)
# Löytyi!

# Note: Jos löytöjä on useita, tulostaa ensimmäisen osuman sijainnin...
```

Harjoitus 10.

```
# Kaisa Saren, Aug 27th, 2015
# Python-course, Helsinki Open University
# Tee ohjelma, joka kysyy käyttäjältä kahta
# sekvenssiä ja tulostaa sitten sekvenssin
# pituuden, jos molemmat sekvenssit ovat samoja.

print('''Tämä ohjelma tarkistaa kahden sekvenssin identtisyyden
ja tulostaa niiden pituuden, jos ovat samoja.''')
sekv1 = input('Anna ensimmäinen sekvenssi tekstimuotoisena: ')
sekv2 = input('Anna toinen sekvenssi: ')

if sekv1 == sekv2: print(len(sekv1))
else: print('Sekvenssit eivät ole samat.')
```

Harjoitus 11.

```
# Kaisa Saren, Aug 27th, 2015
# Python-course, Helsinki Open University
# Tee ohjelma, joka lukee vähintään viisi merkkiä
# pitkän DNA-sekvenssin ja tulostaa siitä kolme
# viimeistä merkkiä.

# Huom. Harkkaohjelma, ei tarkista mitään, esim. merkkien oikeellisuutta

sekv = ''
while len(sekv) < 5:
    sekv = input('Syötä sekvenssi, jonka loppua haluat tutkia (min. 5bp): ')
print('Kolme viimeistä syöttämäsi sekvenssin emästä on: ' + sekv[-3:])
```

Harjoitus 12.

```
# Kaisa Saren, Aug 27th, 2015
# Python Course, Helsinki Open University
# Tee ohjelma, joka lukee syötteenä sekvenssin ja sitten tulostaa
# sen sekä alusta loppuun että lopusta alkuun. Esim:
# Syöte: ”TAGGGAGAAATTT”, Tuloste: TAGGGAGAAATTT ja TTTAAAGAGGGAT
# Tarkistaa että sekvenssi koostuu DNA:n emäksistä, jotka listattu listaan DNA_bases.

print('Ohjelma tulostaa syöttämäsi sekvenssin molemmista lukusuunnista luettuna.')
merkkij = input('Syötä sekvenssi: ')
DNA_bases = ['A','T','G','C']
sekv = [ch for ch in merkkij if ch in DNA_bases]
str_sekv = ''
for base in sekv:
```

```
    str_sekv += base
if str_sekv != merkkij or merkkij == '':
    print('"Sekvenssi oli viallinen. Tarkista sekvenssi ja
käynnistä ohjelma uudelleen."')
else:
    rev_sekv = str_sekv[::-1]
    print(str_sekv + ' ja ' + rev_sekv)
```

Harjoitus 13.

```
# Kaisa Saren, Aug 27th, 2015
# Python-course, Helsinki Open University
# Tee ohjelma, joka lukee syötteenä DNA-sekvenssin ja sitten
# tulostaa montako kappaletta kutakin nukleotidia on.

print('Ohjelma tulostaa syöttämäsi sekvenssin nukleotidimäärät.')
merkkij = input('Syötä sekvenssi: ')
DNA_bases = ['A','T','G','C']
sekv = [ch for ch in merkkij if ch in DNA_bases]
str_sekv = ''
for base in sekv:
    str_sekv += base
if str_sekv != merkkij or merkkij == '':
    print('"Sekvenssi oli viallinen. Tarkista sekvenssi ja
käynnistä ohjelma uudelleen."')
else:
    count_A = 0
    count_T = 0
    count_C = 0
    count_G = 0
    for base in sekv:
        if base == 'A': count_A += 1
        elif base == 'T': count_T += 1
        elif base == 'C': count_C += 1
        else: count_G += 1
    print('A : ' + str(count_A))
    print('T : ' + str(count_T))
    print('C : ' + str(count_C))
    print('G : ' + str(count_G))
```

Kokeillaan... (sisäkkäiset loopit)

“

Mitä tulostaa:

```
for i in range(10):
for j in range(10):
print(i*j)
```

”

Vastaus: Sisennysten korjaamisen jälkeen tulostaa pötkönä tuloja i*j , jossa 0<=i,j<10... Pienellä fiksailulla tulostaa nätin kertotaulun ;)

Siis näin:

```
for i in range(10):
    print()
    for j in range(10):
        print((i+1)*(j+1), end='\t')
```

tulostaa:

1	2	3	4	5	6	7	8	9	10
2	4	6	8	10	12	14	16	18	20
3	6	9	12	15	18	21	24	27	30
4	8	12	16	20	24	28	32	36	40
5	10	15	20	25	30	35	40	45	50
6	12	18	24	30	36	42	48	54	60
7	14	21	28	35	42	49	56	63	70
8	16	24	32	40	48	56	64	72	80
9	18	27	36	45	54	63	72	81	90
10	20	30	40	50	60	70	80	90	100

Harjoitus 14.

```
# Kaisa Saren, Aug 27th, 2015
# Python Course, Helsinki Open University
# Tee ohjelma, joka lukee syötteenä DNA-sekvenssejä yksi
# kerrallaan, kunnes käyttäjä syöttää merkkijonon ”STOP”.
# Lopuksi ohjelma tulostaa luetut sekvenssit (pituusjärjestyksessä).

DNA_bases = ['A','T','G','C']
sekvenssit = []
print('''Ohjelma lukee syöttämiäsi sekvenssejä. "STOP" lopettaa ja tulostaa
syöttämäsi sekvenssit''')
merkkij = input('Syötä ensimmäinen sekvenssi: ')
while merkkij != 'STOP':
    str_sekv = ''   #Alustetaan str_sekv tyhjäksi
    sekv = [ch for ch in merkkij if ch in DNA_bases]
    for base in sekv:
        str_sekv += base
    #testataan että kaikki merkit kelpasivat ja että niitä on...
    if str_sekv != merkkij or merkkij == '':
        print('Sekvenssi oli viallinen. Tarkista sekvenssi ja yritä uudelleen.')
    else:
        sekvenssit.append(str_sekv)
    merkkij = input('Syötä sekvenssi ("STOP" lopettaa): ')
jarj_sekvenssit = sorted(sekvenssit, key=len)
for s in jarj_sekvenssit:
```

```
    print(s)
```

Harjoitus 16.

Edellinen tehtävä muutettuna niin että DNA:n oikeellisuus testataan funktiolla.

```
def test_DNA(sekvenssi):
    ok = True
    DNA_bases = ['A','T','G','C']
    for ch in sekvenssi:
        if ch not in DNA_bases:
            ok = False
            break
    return ok

sekvenssit = []
print('''Ohjelma lukee syöttämiäsi sekvenssejä. "STOP" lopettaa ja tulostaa
syöttämäsi sekvenssit''')
merkkij = input('Syötä ensimmäinen sekvenssi: ')
while merkkij != 'STOP':
    str_sekv = ''   #Alustetaan str_sekv tyhjäksi
    #testataan että kaikki merkit kelpasivat ja että niitä on...
    if test_DNA(merkkij)== False or merkkij == '':
        print('Sekvenssi oli viallinen. Tarkista sekvenssi ja yritä uudelleen.')
    else:
        sekvenssit.append(merkkij)
    merkkij = input('Syötä sekvenssi ("STOP" lopettaa): ')
jarj_sekvenssit = sorted(sekvenssit, key=len)
for s in jarj_sekvenssit:
    print(s)
```

Harjoitus 17.

```
# Kaisa Saren, Aug 27th, 2015
# Python Course, Helsinki Open University
# Tee ohjelma, joka lukee DNA-sekvenssin, mutatoi satunnaisen
# nukleotidin ja tulostaa alkuperäisen ja mutantin allekkain.
# Aja ohjelmaa useamman kerran => muuttuuko tulos?

# Huom! Harjoitusversio, yli rivin pituisilla sekvensseillä rivit eivät tulostu rinnakkain, yli
kymmenen numeroilla viivain ei skaalaudu...

import random
DNA_bases = ['A','T','G','C']
```

```
def test_DNA(sekvenssi):
    ok = True
    if sekvenssi == '':
        ok = False
    else:
        for ch in sekvenssi:
            if ch not in DNA_bases:
                ok = False
                break
    return ok

def mutatoi(sekvenssi):
    mut_loc = random.randint(0,(len(merkkij)-1))
    mut_base = DNA_bases[random.randint(0,(len(DNA_bases)-1))]
    while mut_base == sekvenssi[mut_loc]:   #Uusitaan mutatointi jos arvottiin sama emäs
        # print('uusitaan arvonta...') #testi
        mut_base = DNA_bases[random.randint(0,(len(DNA_bases)-1))]
    print('Mutatoidaan emäs sijainnissa ' + str(mut_loc + 1) + ' emäksellä ' + mut_base)
    mutantti = sekvenssi[:mut_loc] + mut_base + sekvenssi[(mut_loc +1):]
    return mutantti

#Ohjelma alkaa tästä...

print('''Ohjelma tulostaa syöttämäsi sekvenssin ja siitä
luodun satunnaisen pistemutantin.''')
merkkij = input('Syötä sekvenssi: ')

if test_DNA(merkkij) == True:
    mutantti = mutatoi(merkkij)
    viivain = ''
    for num in range(1,(len(merkkij)+1)):
        viivain += str(num)
    print(3*'\t' + viivain)
    print('Syötetty sekvenssi:\t' + merkkij)
    print('Satunnainen mutantti:\t' + mutantti)
else:
    print('Sekvenssi ei kelpaa.')
```

Harjoitus 18.

```
# Kaisa Saren, Aug 31th, 2015
# Python Course, Helsinki Open University
# Ohjelma, joka lukee DNA-sekvenssin ja tulostaa sen komplementin käyttämällä
# sanakirjaa emäspareille (esim. komplementti[”A”] = ”T”)
# Esim. tulostus:
# TAAAGATCC
# ATTTCTAGG

# Huom! Harjoitusversio, yli rivin pituisilla sekvensseillä rivit eivät tulostu rinnakkain.
```

```
DNA_bases = ['A','T','G','C']
komplementti = {'A':'T','T':'A','C':'G','G':'C'}

def test_DNA(sekvenssi):
    ok = True
    if sekvenssi == '':
        ok = False
    else:
        for ch in sekvenssi:
            if ch not in DNA_bases:
                ok = False
                break
    return ok

def hae_vastin(sekvenssi):
    vastinpari = ''
    for base in sekvenssi:
        vastinpari += komplementti[base]
    return vastinpari

#Ohjelma alkaa tästä...

print('''Ohjelma tulostaa syöttämäsi sekvenssin ja siitä
generoidun vastinjuosteen.''')
merkkij = input('Syötä sekvenssi: ')

if test_DNA(merkkij) == True:
    vastinpari = hae_vastin(merkkij)
    print('Syötetty sekvenssi:\t' + merkkij)
    print('Vastinjuoste:\t\t' + vastinpari)
else:
    print('Sekvenssi ei kelpaa.')
```

Harjoitus 19.

parsimonia.txt:n sisältö:

Mikä on parsimoniamenetelmä?
Parsimoniapuun muodostamisen periaate
Hennigin argumentaatio
Wagnerin menetelmä
Wagnerin kaava
Optimaalisuuskriteeri
Wagnerin optimaalisuuskriteeri
Fitchin optimaalisuuskriteeri
Dollon optimaalisuuskriteeri
Camin-Sokalin optimaalisuuskriteeri
Yleistetty optimaalisuuskriteeri

Lyhyimmän mahdollisen puun etsintä
Muodostettujen puiden kuvailu ja vertailu
Puun pituus
Yhdenmukaisuusindeksi
Synapomorfiaindeksi
Muokattu yhdenmukaisuusindeksi
Indeksien ongelmista

```
# Kaisa Saren, Aug 31th, 2015
# Python Course, Helsinki Open University
# Ohjelma, joka kysyy käyttäjältä tiedoston nimen ja sitten tulostaa
# jokaisen rivin (numeroituna) erikseen.

f = open('parsimonia.txt', 'r')
rivi_nro = 1
for line in f:
    print('rivi ' + str(rivi_nro) + ': ' + line)
    rivi_nro += 1
f.close()   #turha näin ohjelman lopussa, mutta harjoituksena silti tässä.
```

Harjoitus 20.

```
# Kaisa Saren, Aug 31th, 2015
# Python Course, Helsinki Open University
# Ohjelma, joka lukee kaksi sekvenssiä konsolilta, avaa tiedoston
# ”tmp.txt” kirjoittamista varten ja kirjoittaa sekvenssit muodossa:
# > sekvenssi_1
# TAAGAGAGGTT
# > sekvenssi_2
# GCGCGTTGGAGAGT

DNA_bases = ['A','T','G','C']

def test_DNA(sekvenssi):
    ok = True
    if sekvenssi == '':
        ok = False
    else:
        for ch in sekvenssi:
            if ch not in DNA_bases:
                ok = False
                break          #Jos viallinen emäs, ei jatketa tarkistusta.
    return ok

t_nimi = 'tmp.txt'
f = open(t_nimi, 'w')
sekv_nro = 1
sekv = input('''Tällä ohjelmalla voit kirjoittaa sekvenssejä tiedostoon.
```

```
Anna ensimmäinen sekvenssi: '')
while sekv != 'STOP':
    if test_DNA(sekv) == True:
        f.write('> sekvenssi_' +str(sekv_nro)+ '\n')
        f.write(sekv + '\n')
        sekv_nro += 1
    else: print('Sekvenssi ei kelpaa. Tarkista ja yritä uudelleen.')
    sekv = input('Syötä seuraava sekvenssi ("STOP" lopettaa: ')
f.close()
valinta = input('''Ohjelma on päättynyt.
Haluatko vielä tulostaa syötetyt sekvenssit? (Y/N)''')
if valinta == 'Y':
    f = open(t_nimi, 'r')
    print(t_nimi)
    for line in f:
        print(line)
```

Harjoitus 21.

```
# Kaisa Saren, Aug 31th, 2015
# Python Course, Helsinki Open University
# Lue ja pilko multi-FASTA-tiedosto yksittäisiksi
# tiedostoiksi. Keksi järkevät tiedostojen nimet.

print('Ohjelmaa pilkkoo multi-FASTA -tiedoston erillisiksi FASTA-tiedostoiksi.')
tiedosto_multi = input('Syötä multi-FASTA -tiedoston nimi. ')
fr = open(tiedosto_multi, 'r')
FASTAt = ''
for line in fr:
    FASTAt += line
FASTA_lista = FASTAt.split('>')

for fasta in FASTA_lista:
    nimi = fasta[:25] + '.fasta'
    fw = open(nimi, 'w')
    fw.write('>')
    fw.write(fasta)
    fw.close()
```

Harjoitus 22.

```
# Kaisa Saren, Aug 31th, 2015
# Python Course, Helsinki Open University
# Tee ohjelma, joka laskee annetusta DNA-sekvenssistä
# nukleotidien lukumäärän. Esim:
```

```
# syöte: TAAAGCCCTTTGGGG
# tuloste: T=4, A=3, C=3 ja G=5 kpl

DNA_bases = ['A','T','G','C']

def test_DNA(sekvenssi):
    ok = True
    if sekvenssi == '':
        ok = False
    else:
        for ch in sekvenssi:
            if ch not in DNA_bases:
                ok = False
                break
    return ok

print('''Ohjelma tulostaa syöttämäsi sekvenssin nukleotidimäärät.''')
merkkij = input('Syötä sekvenssi: ')
count_T = 0
count_A = 0
count_G = 0
count_C = 0
if test_DNA(merkkij) == True:
    for base in merkkij:
        if base == 'T': count_T += 1
        elif base == 'A': count_A += 1
        elif base == 'G': count_G += 1
        elif base == 'C': count_C += 1
    print('T=' + str(count_T) + ', A=' + str(count_A) + ', C=' + str(count_C) + ', G=' + str(count_G) + '
kpl')
else:
    print('Sekvenssi ei kelpaa.')
```

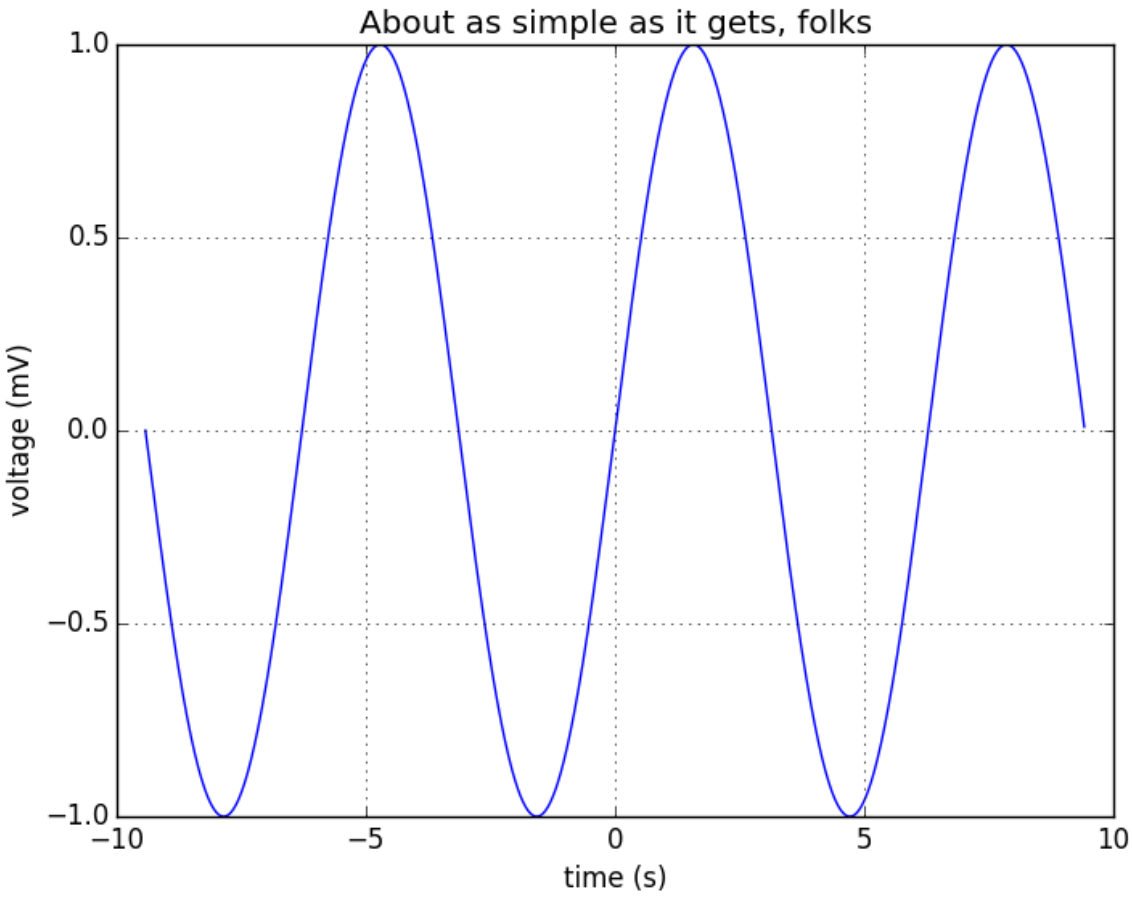
Harjoitus “Matplot”

```
# Kopioi ohjelma alla olevalta sivulta ja muuttele kokeeksi
# muuttujan s arvoa

from pylab import *

t = arange(-3*pi, 3*pi, 0.01)
s = sin(t)
plot(t, s)

xlabel('time (s)')
ylabel('voltage (mV)')
title('About as simple as it gets, folks')
grid(True)
savefig("test.png")
show()
```


About as simple as it gets, folks
1.0
0.5
0.0
−0.5
−1.0
voltage (mV)
−10
−5
0
5
10
time (s)