Exercise 1:

```
>>> def product(seq):

        def product(x, y):

                return x * y

        return reduce(product,seq)

>>> product(range(1,11))

3628800
```

Exercise 2:

```
>>> def delPrime(a):

          for x in range(2, len(a)-1):

                for y in range(2, x):

                  if x % y == 0:

                        break

                else:

                  del a[x]

>>> a = [-1, 1, 66.25, 333, 333, 1234.5]

>>> delPrime(a)

>>> a

[-1, 1, 333]
```

Exercise 3:

```
>>> t = 'Hillary Clinton', 65, 'The White House, Pennsylvania Avenue, Washington D.C., U.S.A.', 'President of the good ol\' U.S.A'

>>> t

('Hillary Clinton', 65, 'The White House, Pennsylvania Avenue, Washington D.C., U.S.A.', "President of the good ol' U.S.A")

>>> name, age, address, occupation = t

>>> name

'Hillary Clinton'

>>> age

65

>>> address

'The White House, Pennsylvania Avenue, Washington D.C., U.S.A.'

>>> occupation

"President of the good ol' U.S.A"
```

Exercise 4:

```
>>> superSet = set('supercalifragilisticexpialidocious')

>>> superSet

set(['a', 'c', 'e', 'd', 'g', 'f', 'i', 'l', 'o', 'p', 's', 'r', 'u', 't', 'x'])

>>> superSet2 = set('Mary Poppins')

>>> superSet2

set(['a', ' ', 'p', 'i', 'M', 'o', 'n', 'P', 's', 'r', 'y'])

>>> superSet2 - superSet

set(['y', ' ', 'M', 'P', 'n'])
```

```
>>> superDict = {'Singer' : 'Mary Poppins', 'Song' : 'supercalifragilisticexpialidocious'}

>>> superDict

{'Singer': 'Mary Poppins', 'Song': 'supercalifragilisticexpialidocious'}

>>> superDict.keys()

['Singer', 'Song']

>>> superDict.has_key('Singer')

True

>>> superDict.has_key('Artist')

False
```

Exercise 5:

```
>>> presDict2 = [{'Name : ' : 'Hilary Clinton', 'Age : ' : 65, 'Office : ' : 2016}, {'Name : ' : 'George W. Bush', 'Age : ' : 68, 'Office : ' : 2001}, {'Name : ' : 'Bill Clinton', 'Age : ' : 68, 'Office : ' : 1993}, {'Name : ' : 'George Bush', 'Age : ' : 90, 'Office : ' : 1989}]

>>> presDict2

[{'Office : ': 2016, 'Name : ': 'Hilary Clinton', 'Age : ': 65}, {'Office : ': 2001, 'Name : ': 'George W. Bush', 'Age : ': 68}, {'Office : ': 1993, 'Name : ': 'Bill Clinton', 'Age : ': 68}, {'Office : ': 1989, 'Name : ': 'George Bush', 'Age : ': 90}]


>>> for x in presDict2:

    print 'Name : ', x['Name : '] ,'\tAge : ', x['Age : '],'\tOffice : ', x['Office : ']



Name :  Hilary Clinton          Age :  65        Office :  2016

Name :  Bill Clinton            Age :  68        Office :  1993

Name :  George W. Bush          Age :  68        Office :  2001

Name :  George Bush             Age :  90        Office :  1989
```

Exercise 6:

times.py

```
>> def times(x, y):
        x = x * y
        return x


>>> import times
>>> times.times(2,3)
6
>>> times = times.times
>>> times(5,5)
25
```

Exercise 7:

pres.txt

```
Hilary Clinton
65
2016
George W. Bush
68
2001
Bill Clinton
68
1993
George Bush
```

```
90

1989

>>> f=open('pres.txt', 'r')

>>> for line in f:

        name = line

        age = f.next()

        office = f.next()

        print 'Name : %sAge : %sOffice : %s' % (name, age, office)



Name : Hilary Clinton

Age : 65

Office : 2016


Name : George W. Bush

Age : 68

Office : 2001


Name : Bill Clinton

Age : 68

Office : 1993


Name : George Bush

Age : 90

Office : 1989
```

Exercise 8:

```
>>> import pickle

>>> pickle.dump( presDict2, open( "pickle.p", "wb" ) )
```

```
pickle.p

(lp0

(dp1

S'Office : '

p2

I2016

sS'Name : '

p3

S'Hilary Clinton'

p4

sS'Age : '

p5

I65

sa(dp6

g2

I2001

sg3

S'George W. Bush'

p7

sg5

I68

sa(dp8
g2

I1993

sg3

S'Bill Clinton'

p9

sg5

I68

sa(dp10

g2

I1989

sg3

S'George Bush'

p11

sg5

I90

sa.


>>> presDict2

[{'Office : ': 2016, 'Name : ': 'Hilary Clinton', 'Age : ': 65}, {'Office : ': 2001, 'Name : ': 'George W. Bush', 'Age : ': 68}, {'Office : ': 1993, 'Name : ': 'Bill Clinton', 'Age : ': 68}, {'Office : ': 1989, 'Name : ': 'George Bush', 'Age : ': 90}]


>>> presDict3 = pickle.load( open( "pickle.p", "rb" ) )

>>> presDict3
```

```
[{'Age : ': 65, 'Name : ': 'Hilary Clinton', 'Office : ': 2016}, {'Age : ': 68, 'Name : ': 'George W. Bush',
'Office : ': 2001}, {'Age : ': 68, 'Name : ': 'Bill Clinton', 'Office : ': 1993}, {'Age : ': 90, 'Name : ': 'George
Bush', 'Office : ': 1989}]
```

The pickled file stores the data in a completely different way to how it is viewed on the screen, each line of the pickled file is used for each defined character such as open parenthesis and close parenthesis so when the file is unpicked it knows how to separate out each dictionary key. In addition to having data with additional characters representing other string elements such as “: and ' ”. This is interpreted when unpickled to restore the real dictionary.