# List

Example (save as `ds_using_list.py`):

```
# This is my shopping list
shoplist = ['apple', 'mango', 'carrot', 'banana']

print('I have', len(shoplist), 'items to purchase.')

print('These items are:', end=' ')
for item in shoplist:
    print(item, end=' ')

print('\nI also have to buy rice.')
shoplist.append('rice')
print('My shopping list is now', shoplist)

print('I will sort my list now')
shoplist.sort()
print('Sorted shopping list is', shoplist)

print('The first item I will buy is', shoplist[0])
olditem = shoplist[0]
del shoplist[0]
print('I bought the', olditem)
print('My shopping list is now', shoplist)
```

Output:

```
$ python ds_using_list.py
I have 4 items to purchase.
These items are: apple mango carrot banana
I also have to buy rice.
My shopping list is now ['apple', 'mango', 'carrot', 'banana', 'rice']
I will sort my list now
Sorted shopping list is ['apple', 'banana', 'carrot', 'mango', 'rice']
The first item I will buy is apple
I bought the apple
My shopping list is now ['banana', 'carrot', 'mango', 'rice']
```

# Tuple

Example (save as `ds_using_tuple.py` ):

```
# I would recommend always using parentheses
# to indicate start and end of tuple
# even though parentheses are optional.
# Explicit is better than implicit.
zoo = ('python', 'elephant', 'penguin')
print('Number of animals in the zoo is', len(zoo))

new_zoo = 'monkey', 'camel', zoo
print('Number of cages in the new zoo is', len(new_zoo))
print('All animals in new zoo are', new_zoo)
print('Animals brought from old zoo are', new_zoo[2])
print('Last animal brought from old zoo is', new_zoo[2][2])
print('Number of animals in the new zoo is',
      len(new_zoo)-1+len(new_zoo[2]))
```

Output:

```
$ python ds_using_tuple.py
Number of animals in the zoo is 3
Number of cages in the new zoo is 3
All animals in new zoo are ('monkey', 'camel', ('python', 'elephant', 'penguin'))
Animals brought from old zoo are ('python', 'elephant', 'penguin')
Last animal brought from old zoo is penguin
Number of animals in the new zoo is 5
```

# Dictionary

Example (save as `ds_using_dict.py`):

```
# 'ab' is short for 'a'ddress'b'ook

ab = {
    'Swaroop': 'swaroop@swaroopch.com',
    'Larry': 'larry@wall.org',
    'Matsumoto': 'matz@ruby-lang.org',
    'Spammer': 'spammer@hotmail.com'
}

print("Swaroop's address is", ab['Swaroop'])

# Deleting a key-value pair
del ab['Spammer']

print('\nThere are {} contacts in the address-book\n'.format(len(ab)))

for name, address in ab.items():
    print('Contact {} at {}'.format(name, address))

# Adding a key-value pair
ab['Guido'] = 'guido@python.org'

if 'Guido' in ab:
    print("\nGuido's address is", ab['Guido'])
```

Output:

```
$ python ds_using_dict.py
Swaroop's address is swaroop@swaroopch.com

There are 3 contacts in the address-book

Contact Swaroop at swaroop@swaroopch.com
Contact Matsumoto at matz@ruby-lang.org
Contact Larry at larry@wall.org

Guido's address is guido@python.org
```

# Sequence

Example (save as `ds_seq.py`):

```
shoplist = ['apple', 'mango', 'carrot', 'banana']
name = 'swaroop'

# Indexing or 'Subscription' operation #
print('Item 0 is', shoplist[0])
print('Item 1 is', shoplist[1])
print('Item 2 is', shoplist[2])
print('Item 3 is', shoplist[3])
print('Item -1 is', shoplist[-1])
print('Item -2 is', shoplist[-2])
print('Character 0 is', name[0])

# Slicing on a list #
print('Item 1 to 3 is', shoplist[1:3])
print('Item 2 to end is', shoplist[2:])
print('Item 1 to -1 is', shoplist[1:-1])
print('Item start to end is', shoplist[:])

# Slicing on a string #
print('characters 1 to 3 is', name[1:3])
print('characters 2 to end is', name[2:])
print('characters 1 to -1 is', name[1:-1])
print('characters start to end is', name[:])
```

Output:

```
$ python ds_seq.py
Item 0 is apple
Item 1 is mango
Item 2 is carrot
Item 3 is banana
Item -1 is banana
Item -2 is carrot
Character 0 is s
Item 1 to 3 is ['mango', 'carrot']
Item 2 to end is ['carrot', 'banana']
Item 1 to -1 is ['mango', 'carrot']
Item start to end is ['apple', 'mango', 'carrot', 'banana']
characters 1 to 3 is wa
characters 2 to end is aroop
characters 1 to -1 is waroo
characters start to end is swaroop
```

```
>>> shoplist = ['apple', 'mango', 'carrot', 'banana']
>>> shoplist[::1]
['apple', 'mango', 'carrot', 'banana']
>>> shoplist[::2]
['apple', 'carrot']
>>> shoplist[::3]
['apple', 'banana']
>>> shoplist[::-1]
['banana', 'carrot', 'mango', 'apple']
```

You can also provide a third argument for the slice, which is the *step* for the slicing (by default, the step size is 1):

# Set

```
>>> bri = set(['brazil', 'russia', 'india'])
>>> 'india' in bri
True
>>> 'usa' in bri
False
>>> bric = bri.copy()
>>> bric.add('china')
>>> bric.issuperset(bri)
True
>>> bri.remove('russia')
>>> bri & bric # OR bri.intersection(bric)
{'brazil', 'india'}
```

# References

`mylist = shoplist` `mylist = shoplist[:]`

Example (save as `ds_reference.py`):

```
print('Simple Assignment')
shoplist = ['apple', 'mango', 'carrot', 'banana']
# mylist is just another name pointing to the same object!
mylist = shoplist

# I purchased the first item, so I remove it from the list
del shoplist[0]

print('shoplist is', shoplist)
print('mylist is', mylist)
# Notice that both shoplist and mylist both print
# the same list without the 'apple' confirming that
# they point to the same object

print('Copy by making a full slice')
# Make a copy by doing a full slice
mylist = shoplist[:]
# Remove first item
del mylist[0]

print('shoplist is', shoplist)
print('mylist is', mylist)
# Notice that now the two lists are different
```

Output:

```
$ python ds_reference.py
Simple Assignment
shoplist is ['mango', 'carrot', 'banana']
mylist is ['mango', 'carrot', 'banana']

Copy by making a full slice
shoplist is ['mango', 'carrot', 'banana']
mylist is ['carrot', 'banana']
```

# Strings

Example (save as `ds_str_methods.py`):

```
# This is a string object
name = 'Swaroop'

if name.startswith('Swa'):
    print('Yes, the string starts with "Swa"')

if 'a' in name:
    print('Yes, it contains the string "a"')

if name.find('war') != -1:
    print('Yes, it contains the string "war"')

delimiter = '_*_'
mylist = ['Brazil', 'Russia', 'India', 'China']
print(delimiter.join(mylist))
```

Output:

```
$ python ds_str_methods.py
Yes, the string starts with "Swa"
Yes, it contains the string "a"
Yes, it contains the string "war"
Brazil_*_Russia_*_India_*_China
```

I want a program which creates a backup of all my important files.

backup_ver1.py :

```
import os
import time

# 1. The files and directories to be backed up are
# specified in a list.
# Example on Windows:
# source = ['"C:\\My Documents"', 'C:\\Code']
# Example on Mac OS X and Linux:
source = ['/Users/swa/notes']
# Notice we had to use double quotes inside the string
# for names with spaces in it.

# 2. The backup must be stored in a
# main backup directory
# Example on Windows:
# target_dir = 'E:\\Backup'
# Example on Mac OS X and Linux:
target_dir = '/Users/swa/backup'
# Remember to change this to which folder you will be using

# 3. The files are backed up into a zip file.
# 4. The name of the zip archive is the current date and time
target = target_dir + os.sep + \
         time.strftime('%Y%m%d%H%M%S') + '.zip'

# Create target directory if it is not present
if not os.path.exists(target_dir):
    os.mkdir(target_dir)  # make directory

# 5. We use the zip command to put the files in a zip archive
zip_command = 'zip -r {0} {1}'.format(target,
                                      ' '.join(source))
```

```
# Run the backup
print('Zip command is:')
print(zip_command)
print('Running:')
if os.system(zip_command) == 0:
    print('Successful backup to', target)
else:
    print('Backup FAILED')
```

```
$ python backup_ver1.py
Zip command is:
zip -r /Users/swa/backup/20140328084844.zip /Users/swa/notes
Running:
  adding: Users/swa/notes/ (stored 0%)
  adding: Users/swa/notes/blah1.txt (stored 0%)
  adding: Users/swa/notes/blah2.txt (stored 0%)
  adding: Users/swa/notes/blah3.txt (stored 0%)
Successful backup to /Users/swa/backup/20140328084844.zip
```

# I want a program which creates a backup of all my important files.

backup_ver2.py :

```
import os
import time

# 1. The files and directories to be backed up are
# specified in a list.
# Example on Windows:
# source = ['"C:\\My Documents"', 'C:\\Code']
# Example on Mac OS X and Linux:
source = ['/Users/swa/notes']
# Notice we had to use double quotes inside the string
# for names with spaces in it.

# 2. The backup must be stored in a
# main backup directory
# Example on Windows:
# target_dir = 'E:\\Backup'
# Example on Mac OS X and Linux:
target_dir = '/Users/swa/backup'
# Remember to change this to which folder you will be using

# Create target directory if it is not present
if not os.path.exists(target_dir):
    os.mkdir(target_dir)  # make directory

# 3. The files are backed up into a zip file.
# 4. The current day is the name of the subdirectory
# in the main directory.
today = target_dir + os.sep + time.strftime('%Y%m%d')
# The current time is the name of the zip archive.
now = time.strftime('%H%M%S')
# The name of the zip file
target = today + os.sep + now + '.zip'

# Create the subdirectory if it isn't already there
if not os.path.exists(today):
    os.mkdir(today)
    print('Successfully created directory', today)

# 5. We use the zip command to put the files in a zip archive
zip_command = 'zip -r {0} {1}'.format(target,
                                      ' '.join(source))

# Run the backup
print('Zip command is:')
print(zip_command)
print('Running:')
if os.system(zip_command) == 0:
    print('Successful backup to', target)
else:
    print('Backup FAILED')
```

```
$ python backup_ver2.py
Successfully created directory /Users/swa/backup/20140329
Zip command is:
zip -r /Users/swa/backup/20140329/073201.zip /Users/swa/notes
Running:
  adding: Users/swa/notes/ (stored 0%)
  adding: Users/swa/notes/blah1.txt (stored 0%)
  adding: Users/swa/notes/blah2.txt (stored 0%)
  adding: Users/swa/notes/blah3.txt (stored 0%)
Successful backup to /Users/swa/backup/20140329/073201.zip
```

# I want a program which creates a backup of all my important files.

backup_ver3.py :

```
import os
import time

# 1. The files and directories to be backed up are
# specified in a list.
# Example on Windows:
# source = ['"C:\\My Documents"', 'C:\\Code']
# Example on Mac OS X and Linux:
source = ['/Users/swa/notes']
# Notice we had to use double quotes inside the string
# for names with spaces in it.

# 2. The backup must be stored in a
# main backup directory
# Example on Windows:
# target_dir = 'E:\\Backup'
# Example on Mac OS X and Linux:
target_dir = '/Users/swa/backup'
# Remember to change this to which folder you will be using

# Create target directory if it is not present
if not os.path.exists(target_dir):
    os.mkdir(target_dir)  # make directory

# 3. The files are backed up into a zip file.
# 4. The current day is the name of the subdirectory
# in the main directory.
today = target_dir + os.sep + time.strftime('%Y%m%d')
# The current time is the name of the zip archive.
now = time.strftime('%H%M%S')
# Take a comment from the user to
# create the name of the zip file
comment = input('Enter a comment --> ')
# Check if a comment was entered
if len(comment) == 0:
    target = today + os.sep + now + '.zip'
else:
    target = today + os.sep + now + '_' +
        comment.replace(' ', '_') + '.zip'

# Create the subdirectory if it isn't already there
if not os.path.exists(today):
    os.mkdir(today)
    print('Successfully created directory', today)

# 5. We use the zip command to put the files in a zip archive
zip_command = "zip -r {0} {1}".format(target,
                                      ' '.join(source))

# Run the backup
print('Zip command is:')
print(zip_command)
print('Running:')
if os.system(zip_command) == 0:
    print('Successful backup to', target)
else:
    print('Backup FAILED')
```

```
$ python backup_ver3.py
  File "backup_ver3.py", line 39
    target = today + os.sep + now + '_' +
                                          ^
SyntaxError: invalid syntax
```

# I want a program which creates a backup of all my important files.

backup_ver4.py :

```
import os
import time

# 1. The files and directories to be backed up are
# specified in a list.
# Example on Windows:
# source = ['"C:\\My Documents"', 'C:\\Code']
# Example on Mac OS X and Linux:
source = ['/Users/swa/notes']
# Notice we had to use double quotes inside the string
# for names with spaces in it.

# 2. The backup must be stored in a
# main backup directory
# Example on Windows:
# target_dir = 'E:\\Backup'
# Example on Mac OS X and Linux:
target_dir = '/Users/swa/backup'
# Remember to change this to which folder you will be using

# Create target directory if it is not present
if not os.path.exists(target_dir):
    os.mkdir(target_dir)  # make directory

# 3. The files are backed up into a zip file.
# 4. The current day is the name of the subdirectory
# in the main directory.
today = target_dir + os.sep + time.strftime('%Y%m%d')
# The current time is the name of the zip archive.
now = time.strftime('%H%M%S')
# Take a comment from the user to
# create the name of the zip file
comment = input('Enter a comment --> ')
# Check if a comment was entered
if len(comment) == 0:
    target = today + os.sep + now + '.zip'
else:
    target = today + os.sep + now + '_' + \
        comment.replace(' ', '_') + '.zip'

# Create the subdirectory if it isn't already there
if not os.path.exists(today):
    os.mkdir(today)
    print('Successfully created directory', today)

# 5. We use the zip command to put the files in a zip archive
zip_command = 'zip -r {0} {1}'.format(target,
                                      ' '.join(source))

# Run the backup
print('Zip command is:')
print(zip_command)
print('Running:')
if os.system(zip_command) == 0:
    print('Successful backup to', target)
else:
    print('Backup FAILED')
```

```
$ python backup_ver4.py
Enter a comment --> added new examples
Zip command is:
zip -r /Users/swa/backup/20140329/074122_added_new_examples.zip /Users/swa/notes
Running:
  adding: Users/swa/notes/ (stored 0%)
  adding: Users/swa/notes/blah1.txt (stored 0%)
  adding: Users/swa/notes/blah2.txt (stored 0%)
  adding: Users/swa/notes/blah3.txt (stored 0%)
Successful backup to /Users/swa/backup/20140329/074122_added_new_examples.zip
```