Types

```
a = 'this is a string'
if str(type(a)) == "<type 'str'>":
  print 'a string'
else:
  print 'not a string'
print type(a)
```

Repeat the above for

```
a = 1.2
a = 1
```

(string)
https://docs.python.org/2.0/ref/strings.html
escape character : backslash \ , raw string r(preceeding the string)
* and + operator on string

docstring triple quotes

(float, integer) also support 'complex'

String

```
str_text = 'This is a string'
len(str_text)
```

Copy slice of str_text

```
str_text[0]
str_text[-1]
str_text[:]
str_text[1:]
str_text[1:4]
str_text[:-2]
str_text[2:-2]
```

Methods and properties of str_text

```
help(str)
```

```
str_text.count('i')
str_text.upper()
str_text
```

List

```
list_var = [ 1, 2.3, 4, 'a string', 5, 1, 4, 5,
6, 'another string' ,[2,3,"d"] ]
len(list_var)

list_var[0]
list_var[1]
list_var[-1]
list_var[1:]
list_var[:-2]

list_var.index(5)

list_var.append(8)
list_var.extend([ 'some string', -6 ])
          *(check how append and extend are different)
list_var
list_var.pop()
list_var

list_var.remove(2.3)
list_var

list_var.reverse()
list_var
list_var.sort()
list_var

for elmt in list_var:
  print elmt, list_var.index(elmt)
```

Set

```
set_var = set(list_var)
set_var
del(set_var)

set_var1 | set_var2
```
*(union set operator)

Dictionary

```
dict_var = { 1 : 2, 3 : 'a', 'b' : 4, 'c' : 'd',
5 : 6, 'e' : [5, 6, 7] }
len(dict_var)
dict_var['f'] = 8
dict_var['d'] = { 'x': 20 , 'y' : 21 }
dict_var.keys()
dict_var.values()

del(dict_var[5])
dict_var.pop('c')
dict_var

for key,val in dict_var.iteritems():
  print key,val
```

key is unique in the dictionary, try var={1:2,1:3} and see

Tuple

```
tuple_var = (1, 2, 3, 4, 5, 'a string', 3, 2, 3,
'another string')
len(tuple_var)
tuple_var[5]
tuple_var[2:5]
tuple_var.count(3)
tuple_var.index(3)
```

Object Identity

```
d = 1
e = 2.3
f = 1
id(d)
id(e)
id(f)

f = 2
id(f)

f = e
id(f)
e,d = d,e
id(d)
id(e)
print d
print e
```

Object

Everything is an object

```
two = 2
type(two)

three = 3.4
type(three)
```

Notes:

Examples on using double quotes “ or quotes ‘

```
mystring="a string with 'quotes'"

mystring2='a string with "quotes"'

mystring3="""a string with "quotes" and more 'quotes'"""

mystring4='''a string with 'quotes' and more "quotes"'''

(triple quotes called “docstrings”)-will be covered later on

mystring5='a string with \"quotes\"'

mystring6='a string with \042quotes\042'

mystring7='a string with \047quotes\047'
```

try to print the aboves