Type Conversion

- Sometimes it's necessary to perform conversions between the built-in types.
- To convert between types you simply use the type name as a function.
- In addition, several built-in functions are supplied to perform special kinds of conversions. All of these functions return a new object representing the converted value.

Function	Description
int(x [,base])	Converts x to an integer. base specifies the base if x is a string.
long(x [,base])	Converts x to a long integer. base specifies the base if x is a string.
float(x)	Converts x to a floating-point number.
complex(real [,imag])	Creates a complex number.
str(x)	Converts object x to a string representation.
eval(str)	Evaluates a string and returns an object.
tuple(s)	Converts s to a tuple.
list(s)	Converts s to a list.
set(s)	Converts s to a set.
dict(d)	Creates a dictionary. d must be a sequence of (key,value) tuples.
frozenset(s)	Converts s to a frozen set.
hex(x)	Converts an integer to a hexadecimal string.
oct(x)	Converts an integer to an octal string.

We can convert between different data types by using different type conversion functions like int(), float(), str() etc.

```
In [8]: float(5)

Out[8]: 5.0
```

Conversion from float to int will truncate the value (make it closer to zero).

```
In [9]: int(10.6666)

Out[9]: 10
```

```
In [10]: int(-10.6)
```
Out[10]: -10

Conversion to and from string must contain compatible values.

```
In [11]: float('2.5')
```
Out[11]: 2.5

```
In [12]: str(25)
```
Out[12]: '25'

```
In [13]: int('1p')
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-13-106779fd957a> in <module>()
----> 1 int('1p')

ValueError: invalid literal for int() with base 10: '1p'
```

```
In [14]: set([1,2,3])
```
Out[14]: {1, 2, 3}

```
In [15]: list('hello')
```
Out[15]: ['h', 'e', 'l', 'l', 'o']

```
In [16]: tuple({5,6,7})
```
Out[16]: (5, 6, 7)

To convert to dictionary, each element must be a pair

```
In [17]: dict([[1,2],[3,4]])
```
Out[17]: {1: 2, 3: 4}

```
In [18]:  dict([(3,26),(4,44)])
```
Out[18]: {3: 26, 4: 44}

```
In [19]: int("A2",16)
```
Out[19]: 162

```
In [20]: int("22",8)
```
Out[20]: 18