함수

함수를 선언할 때에는 def 사용.
파이썬에서 함수를 만들 때 중요한 점은 함수의 입력, 출력, 함수의 동작을 파악하는 것.

```
>>> def cal_upper(price):
     increment = price * 0.3
     upper_price = price + increment
     return upper_price

>>> cal_upper(10000)
13000.0
```

두 개의 값 반환하기
파이썬은 두개 이상 리턴이 가능하다.
단 튜플 형태로 묶어주어야 한다.

```
>>> def cal_upper_lower(price):
        offset = price * 0.3
        upper = price + offset
        lower = price - offset
        return (upper, lower)   # 튜플로 리턴

>>> (upper, lower) = cal_upper_lower(10000)
>>> upper
13000.0
>>> lower
7000.0
```

튜플은 수정이 불가능함

모듈

기능이 분리되어있는 하나의 파일.
파이썬에서는 모듈단위로 실행이 된다.
파이썬의 모듈을 사용하는 경우에는 함수가 정의돼 있는 파일 자체를 복사한 후 모듈을

임포트(import)하기만 하면 해당 파일(모듈)에 구현된 모든 함수 및 자료구조를 사용할 수 있다.
파이썬에는 다양한 기능을 수행하는 모듈이 기본적으로 제공.

모듈 만들기

파이썬 IDLE 에서 새 파일 열기
새 창에 다음과 같이 코드를 작성

```
def cal_upper(price):
    increment = price * 0.3
    upper_price = price + increment
    return upper_price

def cal_lower(price):
    decrement = price * 0.3
    lower_price = price - decrement
    return lower_price

author = "pystock"

print(cal_upper(10000))
print(cal_lower(10000))
print(__name__)
```

File 메뉴에서 Save 메뉴를 클릭해 파일로 저장 (stock.py)

모듈 임포트

```
>>> import stock
13000.0
7000.0
stock

>>> print(stock.author)
pystock

>>> stock.cal_upper(10000)
13000.0
>>> stock.cal_lower(10000)
7000.0
```

아래 코드를 추가하면 임포트 했을때 문구가 출력되지 않는다.

```
if __name__ == "__main__":
    print(cal_upper(10000))
    print(cal_lower(10000))
    print(__name__)
```

추가된 코드의 의미를 살펴보면 __name__이라는 파이썬 변수가 바인딩하고 있는 값이 __main__이라는 문자열이라면 들여쓰기 된 세 개의 print 문을 실행한다. 즉, stock.py 파일이 직접 실행이 된 경우라면 세 개의 print 문이 실행된다. 만약 다른 파일에 stock.py 파일(모듈)이 임포트 된다면 __name__은 모듈의 이름인 stock 을 바인딩하므로 if 문의 조건식을 만족하지 않고 따라서 세 개의 print 문도 실행되지 않는다.

시간

파이썬에서 시간을 사용하려면 모듈을 이용하면 된다.
time 모듈 사용
time 모듈을 사용하려면 먼저 time 모듈을 임포트

```
>>> import time
>>> time.time()
1586479377.1422153
>>> time.ctime()
'Fri Apr 10 09:43:21 2020'
>>> type(_)
<class 'str'>
```

파이썬에서도 split으로 문자열 분리가 가능함

```
>>> cur_time = time.ctime()
>>> print(cur_time.split(' ')[-1])
2020
```

dir() 내장 함수를 호출하면 해당 모듈의 구성 요소를 확인할 수 있다.

```
>>> import time
>>> dir(time)
['_STRUCT_TM_ITEMS', '__doc__', '__loader__', '__name__', '__package__', '_
 'altzone', 'asctime', 'ctime', 'daylight', 'get_clock_info', 'gmtime', 'lo
 'mktime', 'monotonic', 'monotonic_ns', 'perf_counter', 'perf_counter_ns',
 'process_time_ns', 'sleep', 'strftime', 'strptime', 'struct_time', 'thread
 'thread_time_ns', 'time', 'time_ns', 'timezone', 'tzname']

언더바(_)가 있는 함수들은 수정이 불가능하다.
```

OS모듈

os 모듈은 Operating System 의 약자로 운영체제에서 제공되는 여러 기능을 파이썬에서 수행할 수 있게 해준다.
예를 들어, 파이썬을 이용해 파일을 복사하거나 디렉터리를 생성하고 특정 디렉터리 내의 파일 목록을 구하고자 할 때 os 모듈을 사용하면 된다.

```
>>> import os
>>> os.getcwd()
'C:\\Users\\USER\\AppData\\Local\\Programs\\Python\\Python38-32'
```

특정 경로에 존재하는 파일과 디렉터리 목록을 구하는 함수인 listdir()

```
>>> os.listdir()
['DLLs', 'Doc', 'include', 'Lib', 'libs', 'LICENSE.txt', 'NEWS.txt', 'pytho
'python3.dll', 'python38.dll', 'pythonw.exe', 'Scripts', 'stock.py', 'tcl',
'vcruntime140.dll', '__pycache__']

>>> os.listdir('c:/')
['$Recycle.Bin', 'AMTAG.BIN', 'Aomei', 'DB_Drivers', 'Documents and Setting
jee', 'eclipse-jee.zip', 'filetest', 'git_clone', 'git_test', 'hiberfil.sys
'JAVA_test', 'JDK_8', 'Nexon', 'OkBootConfig.dat', 'OKTAG.BIN', 'oraclexe',
'OracleXE112_Win64', 'pagefile.sys', 'PerfLogs', 'Program Files', 'Program
'ProgramData', 'R', 'RealEstate', 'Recovery', 'RStudio', 'sqldeveloper-19.1
x64', 'swapfile.sys', 'System Volume Information', 'Users', 'Windows', 'YKS
'YKSstudy']
```

디렉터리 안의 폴더와 파일 갯수

```
>>> len(os.listdir('c:/windows/'))
109
```

아래는 'c:/windows'이라는 경로에 있는 파일 중 확장자가 'exe'로 끝나는 파일만 출력

```
>>> for x in os.listdir('c:/windows/'):
if x.endswith('exe'):
    print(x)


bfsvc.exe
explorer.exe
fs_shortcut.exe
HelpPane.exe
hh.exe
notepad.exe
py.exe
pyw.exe
regedit.exe
splwow64.exe
winhlp32.exe
write.exe
```

모듈 안에 특정 함수 하나만 임포트 하고자 할때에는 아래와 같이 사용

```
>>> from os import listdir      # os모듈로부터 listdir함수만 임포트 함
>>> dir()
['__annotations__', '__builtins__', '__doc__', '__file__', '__loader__', '_
'__package__', '__spec__', 'author', 'cal_lower', 'cal_upper', 'cur_time',
'os', 'time', 'x']
```

모듈명이 긴 경우에는 아래와 같이 모듈명을 바꿔서 임포트가 가능함

```
>>> import os as winos      # os 모듈을 winos 로 임포트하라
>>> dir()
['__annotations__', '__builtins__', '__doc__', '__file__', '__loader__', '_
'__package__', '__spec__', 'author', 'cal_lower', 'cal_upper', 'cur_time',
'os', 'time', 'winos', 'x']
```

파이썬 내장 함수

파이썬은 여러 함수를 기본 제공.

abs()	dict ()	help ()	min ()	setattr()
all()	dir()	hex()	next()	slice()
any()	divmod()	id()	object()	sorted()
ascii()	enumerate()	input()	oct()	staticmethod()
bin()	eval()	int()	open()	str()
bool()	exec()	isinstance()	ord()	sum()
bytearray()	filter()	issubclass()	pow()	super()
bytes()	float()	iter()	print()	tuple()
callable()	format()	len()	property()	type()
chr()	frozenset()	list()	range()	vars()
classmethod()	getattr()	locals()	repr()	zip()
compile()	globals()	map()	reversed()	import()
complex()	hasattr()	max()	round()	
delattr()	hash()	memoryview()	set()	

chr(i) 내장 함수는 유니코드 값을 입력받은 후 해당 값에 해당하는 문자열을 반환합니다. 예를 들어, 아스키(ASCII) 코드에서 알파벳 대문자 A 는 65, 소문자 a 는 97 에 해당하는데, 해당하는 정숫값을 chr() 함수의 인자로 전달하면 문자열이 반환되는 것을 확인할 수 있습니다.

```
>>> chr(97)
'a'
>>> chr(65)
'A'
```

len(s) 내장 함수는 리스트, 튜플, 문자열, 딕셔너리등을 입력받아 그 객체의 원소 개수를 반환합니다.

```
>>> len(['SK', 'naver'])
2
>>> len('SK hynix')
8
>>> len({1:'SK', 2:'Naver'})
2
```

list() 내장 함수는 문자열이나 튜플을 입력받은 후 리스트 객체로 만들고 해당 리스트를 반환합니다. 다음과 같이 튜플을 리스트로 변환할 때 자주 사용됩니다.

```
>>> list('hello')
['h', 'e', 'l', 'l', 'o']
>>> list((1,2,3))
[1, 2, 3]
```

max() 내장 함수는 입력값 중 최댓값을 반환합니다. 반대로 min() 내장 함수는 입력값 중 최솟값을 반환합니다.

```
>>> max(1,2,3)
3
>>> min([1,2,3])
1
```

sorted() 내장 함수는 입력값을 정렬한 후 정렬된 결괏값을 '리스트'로 반환합니다.

```
>>> sorted((4,3,1,0))
[0, 1, 3, 4]
>>> sorted([5, 4, 3, 2, 1])
[1, 2, 3, 4, 5]
>>> sorted(['c', 'b', 'a'])
['a', 'b', 'c']
>>> sorted(("가", "바", "나", "라"))
['가', '나', '라', '바']
```

int(x) 내장 함수는 문자열을 인자로 입력받아 해당 문자열을 정수형으로 변환한 후 반환합니다. 반대로 str(x) 내장 함수는 객체를 입력받아 문자열로 변환합니다. int(x)와 str(x) 내장 함수는 다음과 같이 문자열을 정수형으로, 정수형을 문자열로 변환할 때 자주 사용됩니다.

```
>>> int('3')
3
>>> str(3)
'3'
```