

# **KAIST MFE, 2024 Spring**

Kim Hyeonghwan

2024-04-29

# Table of contents

<b>Welcome!</b>	<b>10</b>
<b>I Python('24봄)</b>	<b>11</b>
<b>1 Python Basics</b>	<b>12</b>
1.1 Values and Variables . . . . .	12
1.1.1 - Values . . . . .	12
1.1.2 - Variables . . . . .	13
1.1.3 - Variable names . . . . .	14
1.2 Core Data Types . . . . .	14
1.2.1 - Floating Point(float) . . . . .	14
1.2.2 - Complex . . . . .	15
1.2.3 - Integer . . . . .	16
1.2.4 - Boolean(bool) . . . . .	16
1.3 Operators . . . . .	16
1.3.1 - Arithmetic Operators . . . . .	16
1.3.2 - Relational Operators . . . . .	17
1.3.3 - Assignment Operators . . . . .	17
1.3.4 - Logical Operators . . . . .	18
1.4 Python Operators Precedence . . . . .	19
1.4.1 - Python Operators Precedence . . . . .	19
1.5 Mathmatical Function . . . . .	19

1.6	Python I/O . . . . .	20
1.6.1	- Python Output . . . . .	20
1.6.2	- Python Input . . . . .	21
<b>2</b>	<b>Containers</b>	<b>22</b>
2.0.1	- Accessing Values in Strings . . . . .	22
2.0.2	- String operations . . . . .	24
2.0.3	- Built-in string function and methods . . . . .	24
2.0.4	- Immutable String . . . . .	26
2.1	Lists . . . . .	27
2.1.1	- Accessing Items in lists . . . . .	28
2.1.2	- Updating Lists . . . . .	30
2.1.3	- Basic List Operations . . . . .	31
2.1.4	- Built-in List Functions and Mothods . . . . .	32
2.1.5	- Coping List . . . . .	33
2.2	Tuples . . . . .	34
2.2.1	- Accesing values in tuples . . . . .	36
2.2.2	- Updating Tuples . . . . .	36
2.2.3	- Basic tuples operations . . . . .	37
2.2.4	- Built-in tuple functions and methods . . . . .	38
2.3	Dictionaries . . . . .	39
2.3.1	- Accesing values in dictionary . . . . .	40
2.3.2	- Updating dict . . . . .	40
2.3.3	- Basic dict operations . . . . .	41
2.3.4	- Bulit-in dict functions and methods . . . . .	41
2.3.5	- Coping Dictionary . . . . .	43
2.4	Sets . . . . .	44
2.4.1	- Accesing or Updating items in a Set . . . . .	45
2.4.2	- Basic sets operation . . . . .	45
2.4.3	Warning! . . . . .	45

<b>3 Control Flows</b>	<b>47</b>
3.1 Statements & Comments . . . . .	47
3.2 if Conditionals . . . . .	47
3.3 Loops - for loop . . . . .	49
<b>4 Functions</b>	<b>53</b>
<b>5 Numpy</b>	<b>55</b>
<b>금융공학프로그래밍3, Quiz1</b>	<b>62</b>
1. Calculate the following . . . . .	62
(1) 근의 공식 코드 작성 . . . . .	62
(2) 정규분포 확률밀도함수 코드 작성 . . . . .	62
2. Why the error occurs? . . . . .	63
(1) Input error . . . . .	63
(2) Range error . . . . .	63
(3) method error . . . . .	63
3. String . . . . .	64
(1) ‘+’를 이용해서 ‘ABCDFFFDCBA’ 만들기 . . . . .	64
(2) A의 개수 . . . . .	64
(3) ‘FFF’를 반환하는 4가지 다른 표현식 . . . . .	64
(4) ‘ABCDAAADCBA’로 수정하기 . . . . .	65
(5) 소문자로 모두 바꾸기 . . . . .	65
4. Why the error occurs? . . . . .	65
(1) slicing 반환 형식 . . . . .	65
(2) 단일 원소 tuple 표현 . . . . .	66
(3) 변수 할당 labeling vs. copy . . . . .	66
(4) Key값은 Immutable 해야함 . . . . .	66
(5) slicing&indexing for Dict . . . . .	67
5. List . . . . .	67
(1) indexing, slicing . . . . .	67
(2) ‘+’ 활용 . . . . .	67

(3) remove method . . . . .	68
(4) insert method . . . . .	68
6. list2 . . . . .	68
(1) append method . . . . .	68
(2) sorting list . . . . .	69
(3) slicing and replacing . . . . .	69
7. tuple . . . . .	69
(1) sorting tuple via translate to list . . . . .	69
(2) basic operation . . . . .	70
(3) concatenating tuple . . . . .	70

## **금융공학프로그래밍3, Quiz2** 71

1. Dictionary . . . . .	71
(1) . . . . .	71
(2) . . . . .	71
2. Dictionary2 . . . . .	72
(1) . . . . .	72
(2) . . . . .	72
(3) . . . . .	73
(4) . . . . .	73
3. if statement . . . . .	73
4. for loop . . . . .	74

## **금융공학프로그래밍3, Quiz3** 75

1. Calculate the following by using ‘while’ loop. . . . .	75
2. List comprehension . . . . .	75
(1) Cartesian product . . . . .	75
(2) Cartesian product with condition . . . . .	76
3. Function . . . . .	77
4. Explain why the error occurs . . . . .	78

<b>금융공학프로그래밍3, Quiz4</b>	<b>80</b>
1. Find out which one of following is wrong? . . . . .	80
2. . . . .	81
3. . . . .	81
4. . . . .	82
5. . . . .	82
<b>Python Midterm</b>	<b>83</b>
Problem 1. . . . .	83
(1) . . . . .	83
(2) . . . . .	85
(3) . . . . .	86
Problem 2. . . . .	88
(1) . . . . .	88
(2) . . . . .	90
<b>II 채권분석('24봄)</b>	<b>92</b>
<b>채권분석 과제1</b>	<b>93</b>
1. CB . . . . .	93
2. FRN . . . . .	93
3. YTM . . . . .	94
4. Continuous compounding . . . . .	95
5. Arbitrage . . . . .	96
<b>채권분석 과제2</b>	<b>97</b>
1. Callable bond . . . . .	97
2. Spot rate . . . . .	98
3. Forward rate . . . . .	102
4. Continuous compounding - Forward rate . . . . .	104
5. Pricing convention - full price . . . . .	105

<b>채권분석 과제3</b>	<b>107</b>
Question 1 . . . . .	107
Answer . . . . .	107
Question 2 . . . . .	108
Answer . . . . .	108
Question 3 . . . . .	109
Answer . . . . .	109
Question 4 . . . . .	111
Answer . . . . .	111
Question 5 . . . . .	112
Answer . . . . .	112
 <b>III 투자분석('24봄)</b>	 <b>114</b>
 <b>투자분석 과제1</b>	 <b>115</b>
<i>Question</i> . . . . .	115
문제풀이 . . . . .	115
(a) 주식 선정 . . . . .	115
(b) 무위험이자율 및 리스크프리미엄 . . . . .	116
(c) CAPM . . . . .	119
(d) PVGO . . . . .	123
(e) Find $V_0$ , $V_1$ . . . . .	127
(f) Find the expected rate of return for each stock . . . . .	128
(g) Which stock is the best? . . . . .	129
 <b>투자분석 과제2</b>	 <b>131</b>
Problems . . . . .	131
Answer . . . . .	132

<b>IV 재무회계('24봄)</b>	<b>145</b>
<b>재무회계 Ch1-3 과제</b>	<b>146</b>
(1) Quetion 1-31 . . . . .	146
<i>Answer</i> . . . . .	146
(2) Quetion 1-34 . . . . .	147
<i>Answer</i> . . . . .	147
(3) Quetion 2-10 . . . . .	149
<i>Answer</i> . . . . .	149
(4) Quetion 3-12 . . . . .	152
<i>Answer</i> . . . . .	152
<b>재무회계 Ch4 과제</b>	<b>154</b>
(1) Quetion 4-9 . . . . .	154
(2) Quetion 4-15 . . . . .	155
<b>재무회계 Ch5 과제</b>	<b>157</b>
(1) Question 5-12 . . . . .	157
(2) Question 5-18 . . . . .	158
<b>재무회계 Ch6 과제</b>	<b>160</b>
<b>V 경영통계분석('24봄)</b>	<b>165</b>
<b>경영통계분석 과제1</b>	<b>166</b>
1. Type of variables . . . . .	166
2. Discrete data . . . . .	167
3. Histogram . . . . .	170
4. Sample Statistics . . . . .	172
<b>경영통계분석 과제2</b>	<b>175</b>
Question 1 . . . . .	175
Question 2 . . . . .	176

Question 3 . . . . .	179
Question 4 . . . . .	179
Question 5 . . . . .	182
Question 6 . . . . .	183
Question 7 . . . . .	183
<b>경영통계분석 과제3</b>	<b>185</b>
Question 1 . . . . .	185
Answer . . . . .	185
Question 2 . . . . .	186
Answer . . . . .	186
Question 3 . . . . .	187
Question 4 . . . . .	188
Answer . . . . .	189
Question 5 . . . . .	190
Answer . . . . .	190
<b>경영통계분석 과제4</b>	<b>192</b>
Problem 1 . . . . .	192
Answer . . . . .	192
Problem 2 . . . . .	193
Answer . . . . .	193
Problem 3 . . . . .	193
Answer . . . . .	194

# Welcome!

안녕하세요, KAIST MFE 24년 봄학기에 이수한 과목의 과제 등을 정리해두었습니다.

## **Part I**

# **Python('24봄)**

# 1 Python Basics

## 1.1 Values and Variables

### 1.1.1 - Values

```
type("Hello, World")
```

str

```
type(())
```

tuple

```
type([])
```

list

```
type(set())
```

set

```
type({})
```

```
dict
```

```
# 파이썬에서는 복소수 i를 j로 표현함  
type(2+3j)
```

```
complex
```

### 1.1.2 - Variables

```
message="Hello, world!"  
n=42  
e=2.71  
print(n)  
print(message)
```

```
42
```

```
Hello, world!
```

```
length=4.2  
height=3.5  
area=length*height  
print(area)
```

```
14.70000000000001
```

```
# 변수는 방문앞에 이름만 바꾼것... 새로운 변수에 기존 변수를 할당하고, 기존 변수를 변경하면 새로운 변수도 영향을 미친다
a=[1,2,3]
b=a
a[0]=10
print(b)
```

[10, 2, 3]

### 1.1.3 - Variable names

- 숫자로 시작하면 안됨

```
x=1.0
X=1.0
X1=1.0
x1=1.0
dell=1.0
# not allowed
# x:, 1X, X1-1, for
```

```
x,y,z=1,3.14,'a'
print(x,y,z)
```

1 3.14 a

## 1.2 Core Data Types

### 1.2.1 - Floating Point(float)

```
x=1  
y=1.0  
z=float(1)  
xx=1.234e-5  
print(x,y,z,xx)  
print(type(z),type(float(True)))
```

```
1 1.0 1.0 1.234e-05  
<class 'float'> <class 'float'>
```

```
# float 타입은 근사치를 사용하기 때문에 사용에 유의  
1-0.9==0.1
```

```
False
```

```
float('inf')
```

```
inf
```

### 1.2.2 - Complex

```
x=1j  
y=2+3j  
z=complex(1)  
print(x,y,z,y.real,y.imag)  
print(type(y.imag))
```

```
1j (2+3j) (1+0j) 2.0 3.0  
<class 'float'>
```

### 1.2.3 - Integer

```
# Integer는 float와 달리 정확하다  
x=2**127+2**65  
x
```

170141183460469231768580791863303208960

### 1.2.4 - Boolean(bool)

```
print(bool(1),bool(1.2),bool(-1),bool(0.1))
```

True True True True

```
print(bool(0),bool(0.0),bool(0.0000),bool(None),bool(""),bool(()))
```

False False False False False False

```
3>4
```

False

## 1.3 Operators

### 1.3.1 - Arithmetic Operators

```
a=10  
b=20
```

```
# 나누기 연산은 항상 float로 반환됨을 유의  
print(a+b,a-b,a*b,b/a)
```

30 -10 200 2.0

```
# 나머지 연산자  
b%a
```

0

```
# 몫 연산자  
b//a
```

2

```
# 거듭제곱 연산자, 우선순위가 가장 높음  
a**b
```

10000000000000000000000000

### 1.3.2 - Relational Operators

```
print(a==b,a!=b,a>b,a<b,a>=b,a<=b)
```

False True False True False True

### 1.3.3 - Assignment Operators

```
t=0
```

```
t+=1
```

```
t
```

1

```
t/=2
```

```
t
```

0.5

```
t*=2
```

```
t
```

1.0

#### 1.3.4 - Logical Operators

```
print(True and False, True or False, not True)
```

False True False

```
# 논리연산자는 container가 bool이 아닌 경우 용법이 다름
```

```
# or은 가장 먼저나오는 True값을 반환, and는 가장 먼저나오는 false값을 반환
```

```
print('' or 'abc', 0 or 1, 'abc' or '', 'abc' or 'ab')
```

```
print(1 and 0, 'abc' and 0, None and '' and 0)
```

abc 1 abc abc

0 0 None

## 1.4 Python Operators Precedence

### 1.4.1 - Python Operators Precedence

```
# 거듭제곱 > 덧셈뺄셈 > 나눗셈, 둑, 나머지 > 부등호 > 논리  
1+3/2
```

2.5

```
2400//500*500+2000%500
```

2000

```
(2400//500)*500+(2000%500)
```

2000

## 1.5 Mathematical Function

```
# built-in  
print(abs(-1),round(1.111,1))
```

1 1.1

```
# functional in math module  
import math  
  
# factorial, floor, isfinite, isinf, isnan, trunc  
# exp, log, log10, sqrt, cos, sin, tan, degrees, radians, pi, e
```

```
print(  
    math.ceil(1.11),  
    math.log(10),  
    math.log(10,10),  
    math.exp(1),  
    math.e,  
    math.pi,  
    math.trunc(math.pi),  
    math.degrees(1),  
    math.cos(math.pi)  
)
```

2 2.302585092994046 1.0 2.718281828459045 2.718281828459045 3.141592653589793 3 57.295779513

## 1.6 Python I/O

### 1.6.1 - Python Output

```
# print(object1, object2, ... , sep=' ', end='\n',...)  
print(1,2,3,4)  
print(1,2,3,4,sep='*')  
print(1,2,3,4,end='END')  
print(1,2,3,4,sep='')
```

1 2 3 4  
1\*2\*3\*4  
1 2 3 4END1234

```
# formatting
x=5; y=10
print('The value of x is {} and y is {}'.format(x,y))
print('I love {2} and {1}, but hate {0}'.format('cucumber','butter','bread'))
```

```
The value of x is 5 and y is 10
I love bread and butter, but hate cucumber
```

### 1.6.2 - Python Input

```
# Input value is a string
num=input('Enter a number:')
print(num)
print(int(num))
```

```
10
10
```

## 2 Containers

- 기초적인 내용, String / Integer / Dictionary 뿐이 사용함 ## Strings

Strings : An immutable sequence of unicode characters

```
"You can put single quotes ' in a string with double quotes on the outside"
```

```
"You can put single quotes ' in a string with double quotes on the outside"
```

```
'and visa versa ("')'
```

```
'and visa versa ("')'
```

```
print(  
    """muti-line strings  
    with three quotes""")
```

```
muti-line strings  
with three quotes
```

### 2.0.1 - Accessing Values in Strings

```
# 잘 알아들것, 많이 사용
word='KAIST MFE Program'

# indexing
print(word[0],word[5],word[6],sep="/")

# slicing
# str[a:b:c] = start a, end b, jump c
# a is inclusive, b is exclusive, c can be minus
# default : a =0, b=n+1, c=1 but when c is minus then default : a=n, b=0(inclusive)
print(word[0:5],word[6:9],word[10:])
print(word[:2],word[-2:])
print(word[0:10:2])
print(word[::-1])
```

```
K/ /M
KAIST MFE Program
KA am
KITME
margorP EFM TSIAK
```

```
text='Python strings are sliceable.'
text[10]
```

```
'i'
length=len(text)
text[length]
```

```
IndexError: string index out of range
```

```
text[length-1]
```

```
'.'
```

### 2.0.2 - String operations

- +, -, in, not in

```
a='Hello'; b='Python'  
print(a+b, a*2+b*3, 'H' in a, 'm' not in a)
```

```
HelloPython HelloHelloPythonPythonPython True True
```

```
'123'+1
```

```
TypeError: can only concatenate str (not "int") to str
```

```
'123'+str(1)
```

```
'1231'
```

```
int('123')+1
```

```
124
```

### 2.0.3 - Built-in string function and methods

```
# len, .find, .count, .lower, .upper, .split, .replace, .startswith, .endswith
# .isalpha, .isnumeric, .isalnum

# len is function, others are methods

print(
    word,
    len(word),
    word.find('I'),
    word.count(' '),
    word.lower(),
    word.upper(),
    word.split(' '),
    word.replace(' ', '-'),
    sep='\n'
)
```

```
KAIST MFE Program
17
2
2
kaist mfe program
KAIST MFE PROGRAM
['KAIST', 'MFE', 'Program']
KAIST-MFE-Program
True
False
False
True
False
True
False
```

```
print(  
    word.startswith('K'),  
    word.endswith('M'),  
    word.isalpha(),  
    word.replace(' ', '').isalpha(),  
    word.isnumeric(),  
    '123'.isnumeric(),  
    word.isalnum(),  
    word.replace(' ', '1').isalnum(),  
    sep='\n'  
)
```

```
True  
False  
False  
True  
False  
True  
False  
True  
True
```

#### 2.0.4 - Immutable String

```
# string = immutable, not possible to modify  
# method or operations on string != modify but make new string  
sequence="ACGT"  
sequence.replace("A","G")
```

```
'GCGT'
```

```
sequence
```

```
'ACGT'
```

```
id(sequence)
```

```
2053186997616
```

```
sequence=sequence.replace("A","G")
did(sequence)
```

```
2053183781744
```

## 2.1 Lists

```
# a mutable sequence of objects, important!!
# To create List object, use [] or list()
x1=[1,2,3,4]
x2=[1,1.0,1+0j,'one',None,True]
x3=[[1,2,3,4],[5,6,7],[8,9]] # nested list
# 리스트 안에는 무엇이든 올 수 있음
print(x1,x2,x3,sep='\n')
```

```
[1, 2, 3, 4]
```

```
[1, 1.0, (1+0j), 'one', None, True]
```

```
[[1, 2, 3, 4], [5, 6, 7], [8, 9]]
```

```
e1=list()  
type(e1)
```

list

```
len(e1)
```

NameError: name 'e1' is not defined

```
list('cat')
```

['c', 'a', 't']

```
a_tuple=('abc')  
list(a_tuple)
```

['a', 'b', 'c']

### 2.1.1 - Accessing Items in lists

```
# indexing -> original object, slicing -> make sub-list  
myList=[5,2.3,'hello']  
print(  
    myList[0],  
    myList[-1],  
    myList[-3]  
)
```

5 hello 5

```
many_types=[1,55.5,"Am I in a list?",True,"the end"]
print(
    many_types[2:4],
    many_types[2:],
    many_types[:3],
    many_types[-4:-2],
    many_types[-2:-5:-1],
    many_types[:-3],
    many_types[:3:2],
    many_types[:-2:2],
    sep='\n'
)
```

```
['Am I in a list?', True]
['Am I in a list?', True, 'the end']
[1, 55.5, 'Am I in a list?']
[55.5, 'Am I in a list?']
[True, 'Am I in a list?', 55.5]
[1, 55.5]
[1, 'Am I in a list?']
[1, 'Am I in a list?']
```

```
# Multidimensional lists can also be indexed and sliced.
x=[[1,2,3],[4,5,6],[7,8,9,0]]
x[0]
```

```
[1, 2, 3]
```

```
x[0][1]
```

2

```
x[0][0:2]
```

```
[1, 2]
```

### 2.1.2 - Updating Lists

```
# List object is an immutable!!
colorList=['red','blue','green','black','white']
print(id(colorList),colorList)
```

```
2665431628288 ['red', 'blue', 'green', 'black', 'white']
```

```
colorList[2]='yellow'
print(id(colorList),colorList)
```

```
2665431628288 ['red', 'blue', 'yellow', 'black', 'white']
```

```
colorList[2:4]=['gray','purple']
colorList
```

```
['red', 'blue', 'gray', 'purple', 'white']
```

```
colorList[1:4]='r e d'
colorList
```

```
['red', 'r', ' ', 'e', ' ', 'd', 'white']
```

```
colorList[1:]=['orange']
colorList
```

```
['red', 'orange']
```

### 2.1.3 - Basic List Operations

```
L1=[1,2,3]; L2=[4,5,6]; L3=[7,8,9,0]
print(
    L1+L2+L3,
    [L1],
    [L1]+[L2]+[L3],
    sep='\n')
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 0]
[[1, 2, 3]]
[[1, 2, 3], [4, 5, 6], [7, 8, 9, 0]]
```

```
print(
    L1*3,
    3*L1,
    [L1]*3,
    [L1+L2]*3,
    [1,2,3] in [L1]+[L2],
    sep='\n'
)
```

```
[1, 2, 3, 1, 2, 3, 1, 2, 3]
[1, 2, 3, 1, 2, 3, 1, 2, 3]
[[1, 2, 3], [1, 2, 3], [1, 2, 3]]
[[1, 2, 3, 4, 5, 6], [1, 2, 3, 4, 5, 6], [1, 2, 3, 4, 5, 6]]
True
```

## 2.1.4 - Built-in List Functions and Methods

```
# len, min, max, del x[slice]  
# x.append(value), x.extend(list), x.remove(value), x.count(value),  
# x.insert(index, value), x.index(value), x.sort(), x.reverse()  
x=[0,1,2,3,4,5,6,7,8,9]  
del x[0] # 많이 사용  
x
```

[1, 2, 3, 4, 5, 6, 7, 8, 9]

```
del x[:3]  
x
```

[4, 5, 6, 7, 8, 9]

```
del x[:]  
x
```

[]

```
x=[0,1,2,3,4,5,6,7,8,9]  
len(x)
```

10

```
# in place  
x.reverse()  
x
```

[9, 8, 7, 6, 5, 4, 3, 2, 1, 0]

```
# not in place  
sorted(x)
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
x
```

```
[9, 8, 7, 6, 5, 4, 3, 2, 1, 0]
```

```
# in place  
x.sort()  
x
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

### 2.1.5 - Coping List

```
pockets=[5,3,6,7]  
pockets_copy=pockets  
pockets_copy.append(1)  
pockets_copy
```

```
[5, 3, 6, 7, 1]
```

```
# pockets_copy is just labeling of pockets  
pockets
```

```
[5, 3, 6, 7, 1]
```

```
# [:] makes copy of original list object
# At list [:] is copy, but at Array(numpy) [:] is view
pockets_realloc=pockets[:]
pockets_realloc.append(1)
pockets_realloc
```

```
[5, 3, 6, 7, 1, 1]
```

```
pockets
```

```
[5, 3, 6, 7, 1]
```

```
pockets_method=pockets.copy()
pockets_method.append(1)
pockets_method
```

```
[5, 3, 6, 7, 1, 1]
```

```
pockets
```

```
[5, 3, 6, 7, 1]
```

## 2.2 Tuples

2.2.0.1 : tuples are basically immutable lists. To create a tuple, () or tuple()

```
t=(12345,54321,'hello')
t
```

```
(12345, 54321, 'hello')
```

```
# if it contains a single variables, should include comma!
x=(2)
type(x)
```

int

```
x=(2,)
type(x)
```

tuple

```
a=1,2,3,'hello'
a
```

(1, 2, 3, 'hello')

```
w,x,y,z=a
print(w,x,y,z)
```

1 2 3 hello

```
x,_,y,_=a
print(x,y)
```

1 3

```
x,*y=a
print(x,y)
```

1 [2, 3, 'hello']

### 2.2.1 - Accesing values in tuples

```
# can be indexed or sliced.  
x=(1,2,3,4,5,6,7,8,9)  
print(  
    x[5],  
    x[-3],  
    x[3:7],  
    x[3:7:2],  
    sep='\n'  
)
```

```
6  
7  
(4, 5, 6, 7)  
(4, 6)
```

### 2.2.2 - Updating Tuples

```
# Tuples are immutable : impossible to add/remove/replace elements in a tuple  
tup1=(12,34.56)  
tup1[0]=100
```

```
TypeError: 'tuple' object does not support item assignment
```

```
tup2=([1,2],[3,4])  
tup2[0][0]=100  
tup2
```

```
([100, 2], [3, 4])
```

### 2.2.3 - Basic tuples operations

```
tup1=(1,2,3,4)
tup2='a','b','c'
tup1+tup2
```

```
(1, 2, 3, 4, 'a', 'b', 'c')
```

```
tup1*3
```

```
(1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4)
```

```
'a' in tup2
```

```
True
```

```
tup1[0:1]+tup2[1:]
```

```
(1, 'b', 'c')
```

```
tup1[0:1]+tup2[1]
```

```
TypeError: can only concatenate tuple (not "str") to tuple
```

```
# concatenation
tup_even=tup1[1::2]+(6,)
tup_even
```

```
(2, 4, 6)
```

## 2.2.4 - Built-in tuple functions and methods

```
# len, max, min, x.index, x.count
animals=('lama','sheep','lama',48)
len(animals)
```

4

```
print(
    animals.index('lama'),
    animals.count('sheep'),
    sep='\n')
```

0

1

```
# to fix tuple, tuple -> list -> tuple
x_list=list(x)
x_list.reverse()
x=tuple(x_list)
x
```

(9, 8, 7, 6, 5, 4, 3, 2, 1)

```
# tuple is faster than list!
%timeit x=[1,2,3,4,5]
%timeit x=(1,2,3,4,5)
```

```
47.3 ns ± 0.743 ns per loop (mean ± std. dev. of 7 runs, 10,000,000 loops each)
10.2 ns ± 0.0818 ns per loop (mean ± std. dev. of 7 runs, 100,000,000 loops each)
```

## 2.3 Dictionaries

2.3.0.1 : a mutable, unordered collection of key-value pairs

2.3.0.1.1 : unordered collection 최근 업데이트로 입력순으로 order가 생김. 그러나 sequence는 여전히 아님

2.3.0.2 : To create a dict, {key1:value1, key2:value2} initiate with {}, dict()

2.3.0.3 : Keys must be unique and immutable data! (If tuples, it can only contain immutable data.)

```
type({})
```

```
dict
```

```
# two-value sequenced can convert to dict easy!
lot=[(1,2),(3,4),(5,6)]
dict(lot)
```

```
{1: 2, 3: 4, 5: 6}
```

```
los=['a1','b2','c3']
dict(los)
```

```
{'a': '1', 'b': '2', 'c': '3'}
```

```
dict(a=1,b=2,c=3)
```

```
{'a': 1, 'b': 2, 'c': 3}
```

```
a="abc"  
dict(a=1,b=2,c=3)
```

```
{'a': 1, 'b': 2, 'c': 3}
```

### 2.3.1 - Accesing values in dictionary

```
dict1=dict(Name='Zara', Age=7, Class='First')  
dict1
```

```
{'Name': 'Zara', 'Age': 7, 'Class': 'First'}
```

```
dict1['Name']
```

```
'Zara'
```

### 2.3.2 - Updating dict

```
dict1['Age']=8  
dict1['School']='ABC School'  
dict1
```

```
{'Name': 'Zara', 'Age': 8, 'Class': 'First', 'School': 'ABC School'}
```

```
a='Town'  
dict1[a]='Downtown'  
dict1
```

```
{'Name': 'Zara',  
 'Age': 8,  
 'Class': 'First',  
 'School': 'ABC School',  
 'Town': 'Downtown'}
```

### 2.3.3 - Basic dict operations

```
# Check only key value!!  
print(  
    'Name' in dict1,  
    'Age' in dict1,  
    'Country' in dict1,  
    'Country' not in dict1,  
    'Zara' in dict1,  
    sep='\n'  
)
```

```
True  
True  
False  
True  
False
```

### 2.3.4 - Built-in dict functions and methods

```
# len, del x[key]  
# x.clear(), x.items(), x.keys(), x.values(), x.update(dict2)  
print(dict1, len(dict1), sep='\n')
```

```
{'Name': 'Zara', 'Age': 8, 'Class': 'First', 'School': 'ABC School', 'Town': 'Downtown'}  
5
```

```
del dict1['Town']  
dict1
```

```
{'Name': 'Zara', 'Age': 8, 'Class': 'First', 'School': 'ABC School'}
```

```
dict1.clear()  
dict1
```

```
{}
```

```
del dict1  
dict1
```

```
NameError: name 'dict1' is not defined
```

```
dict1=dict(a=1,b=2,c=3)  
dict1.keys()
```

```
dict_keys(['a', 'b', 'c'])
```

```
list(dict1.keys())
```

```
['a', 'b', 'c']
```

```
dict1.values()
```

```
dict_values([1, 2, 3])
```

```
list(dict1.values())
[1, 2, 3]

dict1.items()
dict_items([('a', 1), ('b', 2), ('c', 3)])

d2=dict(apples=1,oranges=2,pears=2)
ud=dict(pears=4,grapes=5,lemons=6)
d2.update(ud)
d2
```

```
{'apples': 1, 'oranges': 2, 'pears': 4, 'grapes': 5, 'lemons': 6}
```

### 2.3.5 - Coping Dictionary

```
dict1
{'a': 1, 'b': 2, 'c': 3}

dict2=dict1.copy()
dict2.clear()
dict1

{'a': 1, 'b': 2, 'c': 3}

dict2=dict1
dict2.clear()
dict1

{}
```

## 2.4 Sets

2.4.0.1 : unordered collection with immutable and unique elements -> Just dict.key !

2.4.0.2 : initiate should be set(), {} is dict!!

```
# elements of set should be immutable!
list1=[[1,2],[3,4]]
set(list1)
```

TypeError: unhashable type: 'list'

```
list2=[(1,2),(3,4)]
set(list2)
```

{(1, 2), (3, 4)}

```
# if elements are not unique, automatically remove duplicates
list3=[1,2,2,3,4]
set(list3)
```

{1, 2, 3, 4}

```
dict1=dict(a=1,b=2,c=3)
set(dict1)
```

{'a', 'b', 'c'}

```
set(dict1.keys())
```

{'a', 'b', 'c'}

```
set(dict1.values())
```

```
{1, 2, 3}
```

```
set(dict1.items())
```

```
{('a', 1), ('b', 2), ('c', 3)}
```

## 2.4.1 - Accesing or Updating items in a Set

### 2.4.1.1 indexing and slicing not supported!!

### 2.4.1.2 set is mutable but set elements are immutable

## 2.4.2 - Basic sets operation

### 2.4.3 Warning!

#### 2.4.3.1 Shallow copy vs. Deep copy

```
import copy as cp
```

```
A=dict(a=1,b=2,c=[1,2,3])
```

```
B=A.copy()
```

```
C=cp.deepcopy(A)
```

```
B['a']=9
```

```
B['c'][0]=9
```

```
print(A,B,C,sep='\n')
```

```
{'a': 1, 'b': 2, 'c': [9, 2, 3]}\n{'a': 9, 'b': 2, 'c': [9, 2, 3]}\n{'a': 1, 'b': 2, 'c': [1, 2, 3]}
```

# 3 Control Flows

## 3.0.0.1 - Control statements in Python

- Conditional Constructs : if
- Loops : for, while

### 3.1 Statements & Comments

### 3.2 if Conditionals

```
# if condition:  
#     statement  
# ---optional!---  
# elif condition:  
#     statement  
# elif condition:  
#     statement  
# else:  
#     statement  
exam1=90; exam2=85  
if(exam1>=90) and (exam2>=90):  
    print('Excellent!')
```

```
if(exam1>=90) | (exam2>=90):  
    print('Good')
```

Good

```
a=10  
if a>10:  
    print("a>10")  
else:  
    print("a=10")
```

a=10

```
# Exercise 1  
a=[1,2,3,4,6,7,9]  
b=a.copy()  
b.sort()  
  
if a==b:  
    print("a is sorted")  
else:  
    print("a is not sorted")
```

a is sorted

```
# Exercise2  
x=[1,2,3,4,5,6,7,8]  
  
x_sort=x.copy()  
x.sort()
```

```

len_x=len(x_sort)
mid_even=(x_sort[int(len(x_sort)/2-1)]+x_sort[int(len(x_sort)/2)])/2
mid_odd=x_sort[int(len(x_sort)//2)]

if len(x_sort)%2==0:
    print("n={}이고, 중간값은 {}입니다.".format(len_x,mid_even))
elif len(x_sort)%2==1:
    print("n={}이고, 중간값은 {}입니다.".format(len_x,mid_odd))
else :
    print("중간값을 산출할 수 없습니다.")

```

n=80이고, 중간값은 4.5입니다.

### 3.3 Loops - for loop

```

# for interating_var in iterable:
#     statements

# exercise 3

a=input("When you enter...")
a_int=int(a)
print("Input a number: "+a,
      "The output will be...",
      sep="\n")
for i in range(1,10):
    print(a+" x "+str(i)+" = "+str(a_int*i))

```

Input a number: 99

The output will be...

```
99 x 1 = 99
99 x 2 = 198
99 x 3 = 297
99 x 4 = 396
99 x 5 = 495
99 x 6 = 594
99 x 7 = 693
99 x 8 = 792
99 x 9 = 891
```

```
# exercise 4
sum_even=0
sum_odd=0
for i in range(10001):
    if i%2==0:
        sum_even+=i
    else:
        sum_odd+=i

print(sum_even,
      sum_odd,
      sum_even+sum_odd,
      sep='\n')
```

```
25005000
25000000
50005000
```

```
# exercise 5
fibonacci=[1,1]
for i in range(10):
```

```

    fibonacci.append(fibonacci[i]+fibonacci[i+1])
print(
    fibonacci,
    sum(fibonacci),
    sep='\n'
)

```

[1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144]

376

```

# exercise 6

a=input("When you enter alpha-num string...")
letter_index=list()
digits_index=list()
non_index=list()
print(
    "Sample Data : "+a,
    "The output will be...",
    sep='\n'
)
for i in range(0,len(a)):
    if a[i].isalpha():
        letter_index.append(i)
    elif a[i].isnumeric():
        digits_index.append(i)
    else: non_index.append(i)

print(
    "Letters {} at {}".format(len(letter_index),letter_index),
    "Digits {} at {}".format(len(digits_index),digits_index),
    "And there are {} at {} which are NOT alphanumeric".format(len(non_index),non_index),
)

```

```
    sep='\n'
```

```
)
```

Sample Data : sample asdf11231,2,1,3.2 21

The output will be...

Letters 11 at [0, 1, 2, 3, 4, 5, 7, 8, 9, 10, 11]

Digits 11 at [12, 13, 14, 15, 16, 18, 20, 22, 24, 27, 28]

And there are 7 at [6, 17, 19, 21, 23, 25, 26] which are NOT alphanumeric

## 4 Functions

### 4.0.0.1 Sqrt 사용자 정의 함수 실습

```
def mySqrt (n):  
    old=0  
    new=n/2  
    while abs(old-new)>1e-10 :  
        old=new  
        new=1/2*(old+n/old)  
    return new
```

```
mySqrt(3)
```

1.7320508075688772

```
list_int=[1,2,3,4,5,6,7,8]  
list_str=["1","2","3","4","5","6","7","8"]
```

```
list(filter(lambda x:x,list_str))
```

['1', '2', '3', '4', '5', '6', '7', '8']

```
list(filter(lambda x:x+"1",list_str))
```

```
['1', '2', '3', '4', '5', '6', '7', '8']
```

```
bool("2")
```

```
True
```

```
list(filter(lambda x:x,list_int))
```

```
[]
```

```
list(filter(lambda x:bool(x-1),list_x))
```

```
[2, 3, 4]
```

```
filter(lambda x:x-1,list_int)
```

```
<filter at 0x1f5d235af10>
```

## 5 Numpy

```
import numpy as np
```

```
a=np.array([np.arange(6)*6]+np.arange(0,51,10).reshape(6,1)
```

```
a
```

```
array([[ 0,  1,  2,  3,  4,  5],
       [10, 11, 12, 13, 14, 15],
       [20, 21, 22, 23, 24, 25],
       [30, 31, 32, 33, 34, 35],
       [40, 41, 42, 43, 44, 45],
       [50, 51, 52, 53, 54, 55]])
```

```
a[0,3:5]
```

```
array([3, 4])
```

```
a[4:,4:]
```

```
array([[44, 45],
       [54, 55]])
```

```
a[0:,2]
```

```
array([ 2, 12, 22, 32, 42, 52])
```

```
a[0:,2:3]
```

```
array([[ 2],  
       [12],  
       [22],  
       [32],  
       [42],  
       [52]])
```

```
a[0:,2]=[1,2,3,4,5,6]
```

```
a
```

```
array([[ 0,  1,  1,  3,  4,  5],  
       [10, 11,  2, 13, 14, 15],  
       [20, 21,  3, 23, 24, 25],  
       [30, 31,  4, 33, 34, 35],  
       [40, 41,  5, 43, 44, 45],  
       [50, 51,  6, 53, 54, 55]])
```

```
a[2::2,0:6:2]
```

```
array([[20,  3, 24],  
       [40,  5, 44]])
```

```
# Operations on Arrays  
# Numpy aggregates
```

```
# Exercise 4  
# F=9/5C+32
```

```
c=np.arange(20,31)  
c*9/5+32
```

```
array([68. , 69.8, 71.6, 73.4, 75.2, 77. , 78.8, 80.6, 82.4, 84.2, 86. ])
```

```
# Exercise 5  
np.random.seed(1)  
Z=np.random.rand(5)  
print(Z,(Z-Z.min())/(Z.max()-Z.min()),sep="\n")
```

```
[4.17022005e-01 7.20324493e-01 1.14374817e-04 3.02332573e-01  
1.46755891e-01]  
[0.57886944 1. 0. 0.41962504 0.20360935]
```

```
# Exercise 6  
Z=np.random.uniform(0,10,10)  
np.floor(Z)
```

```
array([6., 4., 5., 1., 1., 8., 9., 3., 6., 8.])
```

```
np.ceil(Z)-1
```

```
array([6., 4., 5., 1., 1., 8., 9., 3., 6., 8.])
```

```
Z-Z%1
```

```
array([6., 4., 5., 1., 1., 8., 9., 3., 6., 8.])
```

```
# Exercise 7  
x=np.zeros((6,6))  
y=np.arange(0,6)
```

```
x+y
```

```
array([[0., 1., 2., 3., 4., 5.],  
       [0., 1., 2., 3., 4., 5.],  
       [0., 1., 2., 3., 4., 5.],  
       [0., 1., 2., 3., 4., 5.],  
       [0., 1., 2., 3., 4., 5.],  
       [0., 1., 2., 3., 4., 5.]])
```

```
x+y[:,np.newaxis]
```

```
array([[0., 0., 0., 0., 0., 0.],  
       [1., 1., 1., 1., 1., 1.],  
       [2., 2., 2., 2., 2., 2.],  
       [3., 3., 3., 3., 3., 3.],  
       [4., 4., 4., 4., 4., 4.],  
       [5., 5., 5., 5., 5., 5.]])
```

```
x+y.reshape(6,1)
```

```
array([[0., 0., 0., 0., 0., 0.],  
       [1., 1., 1., 1., 1., 1.],
```

```
[2., 2., 2., 2., 2., 2.],  
[3., 3., 3., 3., 3., 3.],  
[4., 4., 4., 4., 4., 4.],  
[5., 5., 5., 5., 5., 5.]])
```

```
# Exercise 9  
np.random.seed(123)  
mat = np.random.randn(6, 4); mat
```

```
array([[-1.0856306 ,  0.99734545,  0.2829785 , -1.50629471],  
      [-0.57860025,  1.65143654, -2.42667924, -0.42891263],  
      [ 1.26593626, -0.8667404 , -0.67888615, -0.09470897],  
      [ 1.49138963, -0.638902 , -0.44398196, -0.43435128],  
      [ 2.20593008,  2.18678609,  1.0040539 ,  0.3861864 ],  
      [ 0.73736858,  1.49073203, -0.93583387,  1.17582904]])
```

```
mat=np.concatenate([mat,np.exp(mat[:,0])[:,np.newaxis]],axis=1)  
mat
```

```
array([[-1.0856306 ,  0.99734545,  0.2829785 , -1.50629471,  0.33768877],  
      [-0.57860025,  1.65143654, -2.42667924, -0.42891263,  0.56068263],  
      [ 1.26593626, -0.8667404 , -0.67888615, -0.09470897,  3.54641154],  
      [ 1.49138963, -0.638902 , -0.44398196, -0.43435128,  4.44326571],  
      [ 2.20593008,  2.18678609,  1.0040539 ,  0.3861864 ,  9.07869158],  
      [ 0.73736858,  1.49073203, -0.93583387,  1.17582904,  2.09042747]])
```

```
mat=mat+[1,-1,1,-1,1]; mat
```

```
array([[-8.56306033e-02, -2.65455342e-03,  1.28297850e+00,  
      -2.50629471e+00,  1.33768877e+00],  
      [ 4.21399748e-01,  6.51436537e-01, -1.42667924e+00,
```

```
-1.42891263e+00,  1.56068263e+00] ,
[ 2.26593626e+00, -1.86674040e+00,  3.21113848e-01,
-1.09470897e+00,  4.54641154e+00] ,
[ 2.49138963e+00, -1.63890200e+00,  5.56018040e-01,
-1.43435128e+00,  5.44326571e+00] ,
[ 3.20593008e+00,  1.18678609e+00,  2.00405390e+00,
-6.13813601e-01,  1.00786916e+01] ,
[ 1.73736858e+00,  4.90732028e-01,  6.41661316e-02,
1.75829045e-01,  3.09042747e+00]])
```

```
mat=mat*np.array([1/10,1/10,1/10,10,10,10]).reshape(6,1)
mat
```

```
array([[ -8.56306033e-03, -2.65455342e-04,  1.28297850e-01,
-2.50629471e-01,  1.33768877e-01] ,
[ 4.21399748e-02,  6.51436537e-02, -1.42667924e-01,
-1.42891263e-01,  1.56068263e-01] ,
[ 2.26593626e-01, -1.86674040e-01,  3.21113848e-02,
-1.09470897e-01,  4.54641154e-01] ,
[ 2.49138963e+01, -1.63890200e+01,  5.56018040e+00,
-1.43435128e+01,  5.44326571e+01] ,
[ 3.20593008e+01,  1.18678609e+01,  2.00405390e+01,
-6.13813601e+00,  1.00786916e+02] ,
[ 1.73736858e+01,  4.90732028e+00,  6.41661316e-01,
1.75829045e+00,  3.09042747e+01]])
```

```
np.median(mat, axis=0)
```

```
array([ 8.80013969,  0.0324391 ,  0.38497958, -0.19676037, 15.67945792])
```

```
np.percentile(mat,70, axis=0)
```

```
array([21.14379101,  2.48623197,  3.10092086, -0.12618108, 42.66846589])
```

```
np.sort(mat[2,:])[::-1]
```

```
array([ 0.45464115,  0.22659363,  0.03211138, -0.1094709 , -0.18667404])
```

```
np.argsort(mat[2,:])[::-1]
```

```
array([4, 0, 2, 3, 1])
```

```
np.sum(np.where(np.exp(mat)>3,1,0),axis=1)
```

```
array([0, 0, 0, 3, 4, 4])
```

```
# or  
np.sum(np.exp(mat)>3, axis=1)
```

```
array([0, 0, 0, 3, 4, 4])
```

```
np.nonzero(mat>np.mean(mat, axis=0))
```

```
(array([0, 1, 1, 2, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5]),  
 array([3, 1, 3, 3, 0, 2, 4, 0, 1, 2, 4, 0, 1, 3]))
```

# 금융공학프로그래밍3, Quiz1

## 1. Calculate the following

```
import math
```

### (1) 근의 공식 코드 작성

```
a=2.0;b=-1.0;c=-15.0;  
print(  
    (-b+math.sqrt(b**2-4*a*c))/(2*a),  
    (-b-math.sqrt(b**2-4*a*c))/(2*a)  
)
```

3.0 -2.5

### (2) 정규분포 확률밀도함수 코드 작성

```
mean=2; std=math.sqrt(3); x=1;  
print(  
    (1/(math.sqrt(2*math.pi)*std))*(math.e**(-(x-mean)**2)/(2*(std)**2)))  
)
```

0.19496965572274114

## 2. Why the error occurs?

### (1) Input error

```
a=input("enter a number:")  
  
a+3
```

input 값은 string타입으로 받게되므로, integer타입인 3과 더하기 연산을 하면 오류 발생, input 값을 받아 integer로 변환하여 integer간 연산을하거나, 3을 string으로 변환하여 문자열간 연산을하거나 해야함.

### (2) Range error

```
tmp='My String'  
tmp[10]
```

tmp는 공백포함 9자리 문자열이므로 index는 0~8까지만 할 수 있고 9부터는 범위 밖이므로 오류 발생.

### (3) method error

```
ex1='sample string'  
ex2=ex1.upper  
ex2[:4]
```

method를 사용할 때, 뒤에 ()를 붙여야 작동함. 붙이지 않을 경우 단순히 함수를 호출하는 것.

### 3. String

```
grade='ABCDF'
```

(1) '+'를 이용해서 'ABC~~D~~FFFDCBA' 만들기

```
grade_str=grade+grade[-1]+grade[::-1]  
grade_str
```

'ABC~~D~~FFFDCBA'

(2) A의 개수

```
grade_str.count('A')
```

2

(3) 'FFF'를 반환하는 4가지 다른 표현식

```
print(  
    grade_str[4:7],  
    grade_str[6:3:-1],  
    grade_str[-7:-4],  
    grade_str[-5:-8:-1]  
)
```

FFF FFF FFF FFF

#### (4) 'ABCDAAAADCBA'로 수정하기

```
grade_str=grade_str.replace('F','A')  
grade_str
```

'ABCDAAAADCBA'

#### (5) 소문자로 모두 바꾸기

```
grade_str=grade_str.lower()  
grade_str
```

'abcdaaaadcba'

## 4. Why the error occurs?

### (1) slicing 반환 형식

```
L=[[1,3,5,7,9],[2,4,6,8,10]]  
L[0][1:2]=30
```

list를 원소로 가지는 list에서 slicing하는 경우, output은 list로 나타나게 되는데, list에 integer 30을 할당하려고 하므로 오류가 발생함.

30을 list로 만들어 할당하거나, slicing 대신 indexing을 통해 output을 integer로 만드는 방법을 사용해야 함.

## (2) 단일 원소 tuple 표현

```
T=(10,20,30)
```

```
T[:2]+(40)
```

tuple에 더하기를 사용하는 경우인데, (40)은 tuple이 아니라 integer이므로 오류가 발생함. 단일 원소를 가지는 tuple을 표현하려면 (40,)으로 써야함.

## (3) 변수 할당 labeling vs. copy

```
D=dict(A=10,B=20,C=30)
# Copy method를 써야 복사본이 할당됨
# D2=D.copy()
D2=D
del D2['A']
D['A']
```

D2=D는 D2에 D.copy가 할당되는게 아니라 단순히 D에 D2라는 label만 새로 달아준것임. 따라서, D2에 'A' key를 제거하면 D의 'A' key도 사라지게 되므로 호출시 오류 발생.

## (4) Key값은 Immutable 해야함

```
D3={[ 'Park' , 'Male' ]:30,}
```

Dictionary 타입의 key값은 immutable한 값만 허용되므로 mutable한 list를 사용하면 오류 발생. tuple을 쓰던지 해야함.

## (5) slicing&indexing for Dict

```
dict_y={(1,):10,(2,):20,(3,):30,(4,):40}  
dict_y[(1,)]
```

Dictionary은 순서가 없음. slicing이 불가하며 indexing도 key값으로만 접근 가능.

## 5. List

```
days=['Mon','Tues','Wed','Thur','Fri',['Sat','Sun']]
```

### (1) indexing, slicing

```
print(  
    [days[5]] ,  
    days[-1::-2] ,  
    days[5][0] ,  
    sep='\n'  
)
```

```
[['Sat', 'Sun']]  
[['Sat', 'Sun'], 'Thur', 'Tues']  
Sat
```

### (2) ‘+’ 활용

```
days2=[days[0:5]]+days[5][0:1]+days[5][1:]  
days2
```

```
[['Mon', 'Tues', 'Wed', 'Thur', 'Fri'], 'Sat', 'Sun']
```

### (3) remove method

```
days2[0].remove('Wed')  
days2[0].remove('Fri')  
days2
```

```
[['Mon', 'Tues', 'Thur'], 'Sat', 'Sun']
```

### (4) insert method

```
days2[0].insert(2,'W')  
days2
```

```
[['Mon', 'Tues', 'W', 'Thur'], 'Sat', 'Sun']
```

## 6. list2

```
Nums=[1,5,2,7,3,6,4]
```

### (1) append method

```
Nums.append(7)
```

```
Nums
```

```
[1, 5, 2, 7, 3, 6, 4, 7]
```

## (2) sorting list

```
Nums.sort()
```

```
Nums.reverse()
```

```
Nums
```

```
[7, 7, 6, 5, 4, 3, 2, 1]
```

## (3) slicing and replacing

```
Nums[::2]=['a','a','a','a']
```

```
Nums
```

```
['a', 7, 'a', 5, 'a', 3, 'a', 1]
```

## 7. tuple

```
price=(180,130,110,160,140,170)
```

### (1) sorting tuple via translate to list

```
price_list=list(price)
price_list.sort()
price=tuple(price_list)
price
```

(110, 130, 140, 160, 170, 180)

## (2) basic operation

```
170 in price
```

True

## (3) concatenating tuple

```
price=price[0:4]+(0,)*3+price[len(price)-1:]
price
```

(110, 130, 140, 160, 0, 0, 0, 180)

# 금융공학프로그래밍3, Quiz2

## 1. Dictionary

```
icecream=dict(cheery=[12,100],cookies=[18,200],greentea=[15,140],mango=[10,200])
```

(1)

Change the price of 'cookies' as 13

```
icecream['cookies'][0]=13  
icecream['cookies']
```

[13, 200]

(2)

Add the following information to the dictionary 'icecream'. - 'Strawberry' is priced at 12 and inventory is 170. - 'pistachio' is priced at 15 and inventory is 100.

```
update_dict=dict(Strawberry=[12,170],pistachio=[15,100])  
icecream.update(update_dict)  
icecream
```

```
{'cheery': [12, 100],  
 'cookies': [13, 200],  
 'greentea': [15, 140],  
 'mango': [10, 200],  
 'Strawberry': [12, 170],  
 'pistachio': [15, 100]}
```

## 2. Dictionary2

```
d1=dict(key1=1,key2=3,key3=2,key10=7)  
d2=dict(key3=1,key2=2,key7=2,key5=4,key1=7,key9=8,key0=7)
```

(1)

Select the largest two values of the dictionary ‘d2’.

```
d2_list=list(d2.values())  
d2_list.sort()  
d2_list.reverse()  
d2_list[0:2]
```

[8, 7]

(2)

Create a set of values which are the unique values of ‘d2’.

```
set(d2.values())
```

{1, 2, 4, 7, 8}

### (3)

Write code that returns True if keys of ‘d1’ are all contained in the key of ‘d2’ and False otherwise.

```
set(d1.keys()) <= set(d2.keys())
```

False

### (4)

Create a set of values which are common in both ‘d1’ and ‘d2’

```
set(d1.values())&set(d2.values())
```

{1, 2, 7}

## 3. if statement

For any arbitrary string object A, if the first character of A is ‘#’, we want it to be replaced with ‘\*’. Otherwise, we want to add ‘\*’ as the first character of A. Write appropriate code using an if statement for this

```
A="#abcde"

if A[0]=="#":
    A="*"+A[1:]
else:
    A="*"+A

A
'*abcde'
```

#### 4. for loop

Calculate the following by using ‘for’ loop :  $\sum_{i=1}^{10} i^4$

```
result=int()

for i in range(1,11):
    result+=i**4

result
```

25333

# 금융공학프로그래밍3, Quiz3

## 1. Calculate the following by using ‘while’ loop.

$$\sum_{i \geq 1, i^4 < 1000}^{10} i^4$$

```
i=1
sum_i4=0
while i>=1 and i**4<1000 and i<=10:
    sum_i4+=i**4
    i+=1
sum_i4
```

979

## 2. List comprehension

```
x=[1,2,3,4]
y=[5,6,7,8]
```

### (1) Cartesian product

```
z=list([i,j] for i in x for j in y)
z
```

```
[[1, 5],
 [1, 6],
 [1, 7],
 [1, 8],
 [2, 5],
 [2, 6],
 [2, 7],
 [2, 8],
 [3, 5],
 [3, 6],
 [3, 7],
 [3, 8],
 [4, 5],
 [4, 6],
 [4, 7],
 [4, 8]]
```

## (2) Cartesian product with condition

```
z=list([i,j] for i in x for j in y if i+j>=8)
z
```

```
[[1, 7],
 [1, 8],
 [2, 6],
 [2, 7],
 [2, 8],
```

```
[3, 5],  
[3, 6],  
[3, 7],  
[3, 8],  
[4, 5],  
[4, 6],  
[4, 7],  
[4, 8]]
```

### 3. Function

```
def grading (score):  
    if type(score)!=int and type(score)!=float:  
        print("Score must be integer or float type.")  
    elif score>100 or score<0:  
        print("Score must be a number between 0 and 100!!")  
    elif score>90:  
        print("Grade is A !")  
    elif score>80:  
        print("Grade is B !")  
    elif score>70:  
        print("Grade is C !")  
    elif score>60:  
        print("Grade is D !")  
    else:  
        print("Grade is F !")
```

```
grading(score=75)
```

Grade is C !

```
grading(score=-5)
```

Score must be a number between 0 and 100!!

#### 4. Explain why the error occurs

- (1) positional argument와 keyword argument를 혼용하는 경우, positional argument가 먼저 와야합니다. keyword argument를 먼저 사용하는 경우 오류가 발생합니다.

```
def infoprint( name, age, gender):
    print(name, 'is', age, 'years old', gender, '.')

infoprint (name='Kim', 30, 'male')
```

- (2) 'Kim'은 positional argument로 name에 할당, 'male'은 keyword로 gender에 할당되고 age는 할당값이 없어서 오류가 발생하였습니다. 함수 정의시 age는 default 값을 정의하지 않았으므로 반드시 값을 할당해야 합니다.

```
def infoprint( name, age, gender):
    print(name, 'is', age, 'years old', gender, '.')

infoprint ( 'Kim', gender='male' )
```

- (3) 함수 myfactorial 정의시 변수 fac을 사용하는데, 해당 변수는 함수 안에서 정의된바 없으므로 함수 밖에서 정의된 global variable인 fac을 가져와서 사용하게 됩니다. 그러나, 이런 경우 global variable의 읽기만 가능하고 쓰기는 불가능하므로 for문 내에 fac을 수정하는 것은 허용되지 않습니다.

```
fac = 1

def myfactorial(n):
    for i in range(n):
        fac *= i+1
    return fac

myfactorial(n=5)
```

## 금융공학프로그래밍3, Quiz4

### 1. Find out which one of following is wrong?

```
game/
__init__.py
sound/
__init__.py
echo.py
wav.py
graphic/
__init__.py
screen.py
render.py
play/
__init__.py
run.py
test.py
```

① >>> from game.sound import echo

```
>>> echo.echo_test()
```

**Right**

② >>> from game.sound.echo import echo\_test

```
>>> echo_test()
```

### Right

③ >>> import game

```
>>> game.sound.echo.echo_test()
```

**FALSE!!** : `__init__.py`가 empty이므로 game package import 시 하단의 하부구조가 자동으로 import 되지 않습니다. 따라서, eco.py 내의 함수 `eco_test()`를 호출하기 위해서는 `eco.py` 또는 `eco_test()`를 직접 호출하여야 합니다.

④ >>> import game.sound.echo

```
>>> game.sound.echo.echo_test()
```

### Right

## 2.

`np.arange[5]`를 이용하여 array구조를 만들면, `dtype='integer'`인 array가 생성됩니다.

이중 하나의 object를 float타입으로 수정하더라도, `dtype`은 `integer`로 고정되므로 입력한 float타입의 데이터가 `integer`로 수정되어 할당됩니다.

## 3.

indexing방식으로 호출하였으므로 1-dimensional이 됩니다. 2-dimensional을 유지하기 위해서는 slicing 방식으로 호출하여야 합니다.

(ex: `rmat[0:1]`)

## 4.

np.hstack 함수는 array를 병합하는 함수로, np.concatenate에 axis=1 옵션과 동일한 함수입니다.

병합하는 두 array간 dimension이 동일해야하는데, a=2, b=1이므로 오류가 발생합니다.

다른 dimension간 병합을 수행하려면 np.column\_stack 함수를 사용하여야 합니다.

## 5.

a는 2-dim. 3X5 array, b는 0-dim. 3 array입니다.

b[np.newaxis,:]는 0-dim인 b에 x축을 추가하여 1X3 array로 만들어주게 됩니다.

3X5와 1X3간에 덧셈이 성립하지 않으므로 오류가 발생합니다.

한편, b[:,np.newaxis]를 사용하면 3X1 array가 되므로, 덧셈이 가능합니다.

# Python Midterm

20249132 김형환

## Problem 1.

### 1. 채권의 가격과 듀레이션

- 각 시점  $t = 1, \dots, n$ 에서  $C_t$ 의 현금흐름이 발생함.
- 연간 쿠폰 지급 횟수가  $f$  번이며 ( $f = 1, 2, 4$ 인 경우만 고려), 쿠폰 이자율은 연  $r\%$ 임.
- 쿠폰은 매 시점마다 지급되며, 액면가격은  $PV$  원은 만기 시점에 쿠폰과 함께 지급됨.
- 채권의 만기는  $M$  (년)임.
- 시장의 만기수익률이 연  $y\%$ 임.
- 이 경우 해당 채권 가격과 듀레이션은 다음 식으로 계산될 수 있음

$$\text{채권 가격} : P = \sum_{t=1}^n \frac{C_t}{(1+(y/100)/f)^t}, \text{듀레이션} : D = \sum_{t=1}^n t \cdot \frac{\frac{C_t}{(1+(y/100)/f)^t}}{P}$$

(1)

5개의 인자 `facevalue=`, `couprate=`, `yield=`, `maturity=`으로 입력하고, `frequency=`에 맞게 “annual”, “semi-annual”, “quarterly” 중 하나의 문자 값으로 입력하면, 위 식에 따라 계산된 채권가격과 듀레이션을 아래와 같은 형태의 튜플로 반환하는 함수 `bondftn`을 작성하여라.

```

def bondftn(pv,r,y,m,f="annual",rnd=3):
    if type(pv)!=float and type(pv)!=int : return print("error: pv should be number!!!!")
    if type(r)!=float and type(r)!=int: return print("error: r should be number!!!!")
    if type(y)!=float and type(y)!=int: return print("error: y should be number!!!!")
    if type(m)!=int: return print("error: M should be integer!!!!")
    if f=="annual": fre=1
    elif f=="semi-annual": fre=2
    elif f=="quarterly": fre=4
    else: return print("error: f can be 'annual', 'semi-annual', 'quarterly'")
    n=fre*m
    coupon=pv*r/100/fre
    price=0.0
    value_duration=0.0
    for i in range(n):
        if i==n-1: cf_i=pv+coupon
        else: cf_i=coupon
        pv_cf_i=cf_i/((1+(y/100/fre))**((i+1)))
        tw_pv_cf_i=((i+1)/fre)*cf_i/(1+(y/100/fre))**((i+1))
        price+=pv_cf_i
        value_duration+=tw_pv_cf_i
    duration=value_duration/price
    result=(round(price,rnd),round(duration,rnd))
    return result

```

```
bondftn(pv=100,r=5,y=4.5,m=2,f="quarterly")
```

(100.951, 1.916)

## (2)

만기 M이 5, 4, 3, 2, 1인 각 경우에 대해, 쿠폰이자율이 5%, 4%, 3%, 2%, 1% 인 각각에 대하여 만기 수익률이 10%에 11%로 상승함에 따라 채권 가격의 변화율이 몇 %인지를 모두 계산한 뒤, 아래와 같은 중첩된 딕셔너리 result\_dict으로 저장하여라.

단, 액면가는 100, 쿠폰 지급 횟수는 연간 1회라고 하자.

```
result_dict=dict()
for i in range(5):
    m_i=5-i
    result_dict['M='+str(m_i)]=dict()
    for j in range(5):
        r_j=5-j
        price_before=bondftn(pv=100,r=r_j,y=10,m=m_i,f="annual")[0]
        price_after=bondftn(pv=100,r=r_j,y=11,m=m_i,f="annual")[0]
        price_return=(price_after-price_before)/price_before
        result_dict['M='+str(m_i)][str(r_j)+'%']=round(price_return,5)
result_dict
```

```
{'M=5': {'5%': -0.03974,
          '4%': -0.04046,
          '3%': -0.04126,
          '2%': -0.04215,
          '1%': -0.04314},
 'M=4': {'5%': -0.03287,
          '4%': -0.03332,
          '3%': -0.03381,
          '2%': -0.03434,
          '1%': -0.03491},
 'M=3': {'5%': -0.02544,
          '4%': -0.02568,
```

```

'3%': -0.02593,
'2%': -0.02619,
'1%': -0.02648},
'M=2': {'5%': -0.01749,
'4%': -0.01758,
'3%': -0.01765,
'2%': -0.01776,
'1%': -0.01784},
'M=1': {'5%': -0.00901,
'4%': -0.009,
'3%': -0.009,
'2%': -0.009,
'1%': -0.00901}}

```

```
result_dict['M=5']['5%']
```

-0.03974

### (3)

만기 M이 5, 4, 3, 2, 1인 각 경우에 대해, 쿠폰이자율이 5%, 4%, 3%, 2%, 1% 인 각각에 대하여 만기수익률이 10%로 주어졌을 때의 듀레이션을 (1)에서의 bondftn을 활용하여 모두 계산하여라.

또한 이를 (2)에서와 동일한 구조를 가지는 (중첩된) 딕셔너리 result\_dict\_dur에 저장하여라.

즉 아래와 같은 방식으로 값을 추출할 수 있어야 한다. 단, 여기서도 액면가는 100, 쿠폰 지급 횟수는 연간 1회라고 하자.

```

result_dict_dur=dict()
for i in range(5):
    m_i=5-i
    result_dict_dur['M='+str(m_i)]=dict()

```

```

for j in range(5):
    r_j=5-j
    result_dict_dur['M=' + str(m_i)][str(r_j) + '%']=bondftn(pv=100,r=r_j,
                                                          y=10,m=m_i,
                                                          f="annual",rnd=5)[1]
result_dict_dur

```

```

{'M=5': {'5%': 4.48786,
          '4%': 4.57019,
          '3%': 4.66101,
          '2%': 4.76171,
          '1%': 4.874},
 'M=4': {'5%': 3.6951,
          '4%': 3.74653,
          '3%': 3.80216,
          '2%': 3.8625,
          '1%': 3.9282},
 'M=3': {'5%': 2.84899,
          '4%': 2.87566,
          '3%': 2.90394,
          '2%': 2.93397,
          '1%': 2.96593},
 'M=2': {'5%': 1.95023,
          '4%': 1.95941,
          '3%': 1.96896,
          '2%': 1.97889,
          '1%': 1.98923},
 'M=1': {'5%': 1.0, '4%': 1.0, '3%': 1.0, '2%': 1.0, '1%': 1.0}}

```

```
result_dict_dur['M=5']['4%']
```

4.57019

## Problem 2.

### 2. 자동차 보험회사에 관한 몬테카를로 시뮬레이션

어느 자동차 보험회사의 보험료 수입과 보험금 지급은 다음의 프로세스로 설명된다고 하자.

- 보험금의 연간 청구 건수는 평균이 100인 포아송 분포를 가짐.
- 각 청구 건수 별 청구금액은 모양이 2, 척도가  $1/2$  (즉, 평균은 1이고 분산은  $1/2$ )인 감마분포로부터 나오며, 청구가 이루어지자 마자 보험금은 지급된다고 가정.
- 각 청구 건수 별 청구 발생시점은 0에서 1(년) 사이의 균일분포를 따른다고 가정.
- 보험회사의 보험료 수입은 연간 150의 비율로 1년 균등히 보험료를 받게 됨. 즉 어떤 해에  $t$ 시점 ( $0 \leq t \leq 1$ )까지 받게 되는 전체 보험료는  $150t$ 임.

### (1)

어떤 1년 동안 발생하는 모든 청구들의 시간과 금액을 모의실험해 보자.

0에서 시작하여 1년 동안 보험회사의 잔고를 계산한 뒤 balance라는 리스트로 저장하여라.

단, balance의 첫번째 값은 0이며, 보험 청구가 발생하는 시점( )에만, 해당 시점의 balance (그 시점까지 받은 보험료 수익(150 ))을 더하고, 그 시점에서 청구로 지급되는 보험금을 빼 주는 방식)를 계산하여 순서대로 저장할 것.

### Answer

먼저, 보험금의 연간 청구건수가 평균이 100인 포아송분포를 따르므로 pmf,  $P(X = x) = \frac{\lambda^x e^{-\lambda}}{x!}$ ,  $\lambda = 100$ 입니다.

다음으로, 보험금의 청구금액은 평균은 1, 분산은  $1/2$ 인 감마분포를 따르므로  $pdf, f(x) = x^{k-1} \frac{e^{-x/\theta}}{\theta^k \Gamma(k)}$ ,  $k\theta = 1$ ,  $k\theta^2 = 1/2$ 입니다. 즉,  $k = 2$ ,  $\theta = 1/2$ 입니다.

각 청구금액은 즉시 지급되며, 발생시점은  $0 \sim 1$  사이의 uniform distribution을 따르므로  $pdx. f(x) = \frac{1}{1-0} = 1$ 입니다.

이제, 1년간 발생하는 보험손익의 모의실험에 대한 청구건수(포아송분포)를  $N$ 번이라고 하고, 청구금액(감마분포) 및 청구시점(균등분포)를 각각  $g_i$ 와  $t_i$  for  $1, \dots, N$ 라고 하면,  $t = t_i$  시점의 balance는 아래와 같이 계산됩니다.

$$balance_{t_i} = 150t_i - \sum_{k=1}^i g_k$$

또한, 기말시점( $t=1$ )의 최종 balance는  $150 - \sum_{all k} g_k$ 가 됩니다.

```
# 포아송, 감마, 균등분포에 대한 모의실험은 numpy의 random module 활용
import numpy as np
from numpy import random as rd
```

```
rd.seed(0) # seed를 설정하여 통제
N=rd.poisson(lam=100,size=1)
N=N[0]
t=rd.uniform(low=0.0,high=1.0,size=N)
t=np.sort(t)
g=rd.gamma(shape=2.,scale=0.5,size=N)
balance=[0] # balance의 초기값은 0
for i in range(N):
    balance.append(round(t[i]*150-sum(g[:i+1]),4))
balance.append(round(150-sum(g),4)) # balance의 기말값
```

```
print(balance[:5],balance[N-5:],sep="\n")
```

```
[0, -1.9755, -2.1182, -2.7092, -3.9799]  
[44.5214, 47.2399, 48.4395, 48.2655, 47.6871, 48.74, 50.4839]
```

## (2)

위 모의실험을 10000회 반복하여 다음에 대한 확률을 추정해 보자.

- 보험회사가 최종적으로 가지게 되는 balance의 기대값(모의실험을 10000회 반복하였을 때 각 모의실험마다 계산된 balance의 최종값들의 평균으로 추정)은 얼마인가?
- 보험회사의 balance가 1년 중 한번 이상 -5 이하로 떨어질 확률(10000회의 모의실험 중에서 balance 가 -5이하로 떨어진 적이 있었던 경우의 비율로 추정)은 얼마인가?

### Answer

평균은 약 49.86, -5이하로 떨어질 확률은 약 7.05%입니다.

```
last_balance=list()  
minus5_balance=0  
for k in range(10000):  
    rd.seed(k)  
    N=rd.poisson(lam=100,size=1)  
    N=N[0]  
    t=rd.uniform(low=0.0,high=1.0,size=N)  
    t=np.sort(t)  
    g=rd.gamma(shape=2.,scale=0.5,size=N)  
    balance=[0]  
    for i in range(N):  
        balance.append(round(t[i]*150-sum(g[:i+1]),4))  
    last_balance.append(round(150-sum(g),4))  
    balance.sort()  
    if balance[0]<=-5:minus5_balance+=1
```

```
print(sum(last_balance)/10000,  
      minus5_balance/10000,sep="\n")
```

49.864134890000095

0.0705

## **Part II**

### **채권분석('24봄)**

# 채권분석 과제1

## 1. CB

(*Question*)

전환사채를 액면 F 만큼 매입할 때 전환가가 P라면, 전환주식수는  $F/P$ 로 주어진다. 특정 전환사채를 액면 1억원 만큼 시장에서 매입하였다. 매입가는 1억 1천만원이다. 전환가격은 5만원으로 주어져 있다.

해당 주식의 시장가가 6만원일 때 전환하여 매도하였다면 투자자의 수익률은 무엇인가? %로 쓰되, 소수점 둘째 자리까지만 쓰시오.

(*Answer*) 9.09%

$$\begin{aligned} &= \frac{1}{5} = 2,000 \\ &= 2000 \times 60,000 = 120,000 \\ &= 11\%, \quad = \frac{1.2}{1.1} - 1 \approx 9.09\% \end{aligned}$$

## 2. FRN

(*Question*)

정부가 2년 만기 변동금리채권을 발행한다고 가정하자. 기준금리는 만기 1년 현물 이자율 (spot rate 또는 1년 만기 무이표채 국채의 이자율)이고, 해당 변동금리채권의 이자는 1년마다 계산하여 지급한다, 즉 1년 말에 지급하고 2년 만기시 원금과 함께 지급한다.

현재 1년 spot rate 을  $r_{1,0}$ , 1년 후 spot rate 을  $r_{1,1}$  이라 하면 받게 되는 총 금액은 액면 100 당  $100r_{1,0}$  (1년 시점) 과  $100(1 + r_{1,1})$  (2년 시점)이다. 물론  $r_{1,1}$  은 현재 시점에서는 알 수 없는 확률변수이다.

액면이 100 인 위의 2년 만기 변동금리채권의 현재 시점의 공정한 가격을 구하시오. 투자자는 시장에서 1년 만기 무이표채 국채를 현재 시점이던 1년 후 시점이던 자유롭게 거래할 수 있다.

(*Answer*) 100

상기 FRN은 1년 만기 무이표 국채의 이자율만큼의 쿠폰을 지급하고 있다. 이러한 액면 100인 FRN 의 현금흐름은 현재 1년 만기 무이표국채에  $100r_{1,0}$  만큼 투자하고, 1년 뒤 만기시점에 다시 1년 만기 무이표국채에  $100 + r_{1,1}$  투자함으로써 복제할 수 있다.

즉, 해당 FRN은 무위험이자율 만큼의 연수익률을 보장하는 채권으로서 쿠폰=할인율이므로 공정가격은 **100(par-value)**에 형성되어야 한다.

#### **i** Foward rate를 이용한 풀이

$r_{1,1}$ 은 확률변수이므로, FRN의 2년뒤 현금흐름  $100(1 + r_{1,1})$ 을 현재가치로 환산한다면 그 공정 가치는  $\frac{100(1+E[r_{1,1}])}{(1+r_{2,0})^2}$  으로 표현할 수 있다.

한편, 1년뒤 1년 spot-rate의 기댓값은 foward rate으로 볼 수 있고, 현재시점의 2년 spot-rate는  $(1 + r_{1,0})(1 + f_{1,1})$ 로 표현할 수 있다. 따라서, FRN의 공정가격은 다음과 같다.

$$FRN\ price = \frac{100r_{1,0}}{1 + r_{1,0}} + \frac{100(1 + E[r_{1,1}])}{1 + r_{2,0}} = \frac{100r_{1,0}}{1 + r_{1,0}} + \frac{100(1 + f_{1,1})}{(1 + r_{1,0})(1 + f_{1,1})} = \frac{100(1 + r_{1,0})}{1 + r_{1,0}} = 100$$

## 3. YTM

(*Question*)

잔존만기가 3년이고 연 8% 이자를 6개월마다 제공하는 채권을 가정하자. A 투자자는 이 채권에 관심이 있고 거래하는 딜러는 100달러 액면당 92.5067 달러를 요구한다.

이때 이 채권의 수익률은 얼마인가? % 로 표시하되 소수점은 반올림하여 정수로 나타내시오.

(*Answer*) 11%

채권의 수익률  $y$ 와 가격에 관한 산식은 다음과 같다.

$$92.5067 = \sum_{k=1}^6 \frac{8/2}{(1+y/2)^k} + \frac{100}{(1+y/2)^6}$$

이를 엑셀 해찾기 기능으로 근사한  $y$ 값은 약 11%이다.

## 4. Continuous compounding

(Question) 10.71%

위의 문제는 compounding frequency 를 연 1회로 가정하고 있다. 이것은 이산복리법이다. 이 frequency 가 무한이 되는 연속복리법에 따르면 1달러 투자가 1년 후에  $e^y$  가 된다. 여기서  $y$  는 연속복리이자율이다.

위 현금흐름 수익률을 연속복리이자율로 표현하시오. %로 나타내되 소수점 둘째 자리까지 쓰시오.

(Answer)

(3)의 현금흐름을 연속복리로 표현하면 다음과 같다.

$$92.5067 = \sum_{k=1}^6 \frac{8}{2} e^{-0.5ky} + 100e^{-3y}$$

이를 엑셀 해찾기 기능으로 근사한  $y$ 값은 10.71%이다.

**i** 수익률 관계를 이용한 풀이

(3)의 수식과 비교하여 단순화  $e^{0.5y_{continuous}} = (1+y_{annual}/2) = 1.055$ 로 표현할 수 있고 값은 10.71%로 오차범위 내에서 동일하다.

## 5. Arbitrage

(Question)

아래와 같은 세 개의 채권이 시장에서 거래되고 있다. 트레이더 B 는 이를 보고 곧 차익거래의 기회가 있음을 파악하고 전략적으로 거래하여 이익을 얻었다. 어떤 전략이 가능할지 본인의 생각을 기술하시오.

- 채권 a: 만기 1년 무이표채, 액면 100 당 가격 90
- 채권 b: 만기 2년 무이표채, 액면 100 당 가격 80
- 채권 c: 만기 2년 이표채, coupon rate 10%, 매년 말 이자지급, 액면 100 당 가격 100

(Answer) 채권a:채권b=1:11로 투자하여 채권c를 만들고 시장에 매도

채권a와 채권b의 수익률은 zero-rates이며, 이를  $y_a, y_b$ 라고 할 때, 다음과 같이 쓸 수 있다.

$$\frac{100}{1+y_a} = 90, \frac{100}{(1+y_b)^2} = 80 \Rightarrow \frac{1}{1+y_a} = 0.9, \frac{1}{(1+y_b)^2} = 0.8$$

이를 이용하여 채권c의 현금흐름을 평가하면,  $\frac{10}{1+y_a} + \frac{110}{(1+y_b)^2} = 9 + 88 = 97$

즉, 액면 100인 채권c의 공정가격은 97이나, 현재 100에 거래되고 있으므로 3만큼 고평가되어있는 상태이다. 따라서 트레이더B가 채권a:채권b=1:11 비중으로 매수한다면 액면이 10인 채권c의 현금흐름을 정확하게 복제할 수 있는데, 이를 이용하면 무위험 차익거래를 할 수 있다.

예를 들어,

- (1) 액면 100억원 채권c를 시장에 매도(공매도 or 발행)하여 100억원의 자금을 조달
- (2) 액면 10억원 채권a를 9억원에, 액면 110억원 채권b를 88억원에 매수
- (3) 1년뒤 채권a의 원금 10억을 받아 채권c의 이표 10억원을 갚고, 2년뒤 채권b의 원금 110억을 받아 채권c의 원금+이표 110억원을 갚을 수 있으므로 현재 시점에서 미래현금흐름은 0임(리스크=0)
- (4) 한편, 현재 100억원을 빌려 97억원을 사용하였으므로 3억원의 무위험차익 실현 (또는, 남은 3억원을 채권b에 투자하여 현재시점의 투입자본없이 2년뒤 3.75억의 수익 실현 가능)

# 채권분석 과제2

## 1. Callable bond

(*Question*)

만기 2년, 액면이자율 10% (연 1회 지급), 액면가 100 인 수의상환채권 callable bond 를 생각하자. 이 채권의 상환액 call price 는 100, 상환기간은 1년부터 (이자지급 이후) 만기까지 이다. 즉, 채권발행자 (기채자, 채무자) 는 상환기간 동안 원하는 때 100으로 채권을 상환할 수 있다. 1년이 지난 시점에서 만기 수익률을 10.05% 라 가정하자.

채권의 가격  $P_0$  를 구하시오.

시중금리변화에 따라 이 만기수익률이 즉각적으로 10bp 상승할 때의 가격  $P_+$  를 구하시오.

그리고 만기수익률이 즉각적으로 10bp 하락할 때의 가격  $P_-$  를 구하시오.

채권발행자는 채권의 가치가 100을 넘어서면 상환할 인센티브가 있음을 고려하라. 이를 세 값을 바탕으로 우리는 수의상환채권에 대하여 유효한 드레일레이션을  $-\frac{1}{P_0} \times \frac{P_+ - P_-}{20bp}$  로 구할 수 있다. 소수점 네째자리까지 리포트하시오.

(*Answer*)

1년이 지난 시점에서 수의상환채권의 잔여현금흐름은 만기 일시납 110이며, YTM은 10.05%이므로 가격  $P_0$ 는 약 99.9546입니다.

$$P_0 = 110 / (1 + 0.1005)$$

$P_0$

[1] 99.95457

만기수익률이 10bp씩 즉각적으로 움직일 때, 수의상환부를 고려하지 않은 채권가격은 아래와 같습니다.

```
P0_up=110/(1+0.1005+0.001)  
P0_down=110/(1+0.1005-0.001)  
c(P0_up,P0_down)
```

```
[1] 99.86382 100.04548
```

그러나, 수의상환채권의 가격은 call price를 초과할 수 없으므로  $P_+ = 99.8638$ ,  $P_- = 100$ 입니다.

위 가격으로부터 산출한 잔존만기 1년의 수의상환채권의 듀레이션은 약 0.6812입니다.

$$D_{callable} - \frac{1}{P_0} \times \frac{P_+ - P_-}{20bp} = 0.6812$$

```
duration=(-1/P0)*(P0_up-100)/0.002  
round(duration,4)
```

```
[1] 0.6812
```

## 2. Spot rate

(*Question*)

hw2.xlsx에는 Treasury par yield (%) 값이 일부 주어져 있다. 이것은 2024년 3월 21일 기준 US Treasury Par Yield Curve이다.

선형보간법을 이용하여 빈 곳을 채우고, 1년에 2회 이자를 지급하는 기준으로 하여 spot rate을 구하시오. 단, 0.5년과 1년 채권은 무이표 채권이다. 제출할 때에는 %로 소수점 둘째 자리까지 보이시오.

(*Answer*)

먼저, 선형보간법을 통해 기간별 Par-yield를 구하도록 하겠습니다.

```

library(tidyverse)
par_yield <- tibble("year"=seq(0.5,10,0.5),
                     "period"=c(1:20),
                     "paryield"=c(5.36,5.01,NA,4.62,NA,
                                4.42,NA,NA,NA,4.26,
                                NA,NA,NA,4.28,NA,
                                NA,NA,NA,NA,4.27))

```

```

par_yield <- par_yield %>%
  mutate(interpolation=if_else(paryield %>% is.na(),
                               (lag(paryield)+lead(paryield))/2,
                               paryield)) %>%
  mutate(interpolation=if_else(interpolation %>% is.na(),
                               (lag(lag(interpolation))+lead(lead(interpolation)))/2,
                               interpolation)) %>%
  mutate(interpolation=if_else(interpolation %>% is.na(),
                               (lag(interpolation)+lead(interpolation))/2,
                               interpolation))

par_yield <- par_yield %>%
  mutate(interpolation=if_else(interpolation %>% is.na(),
                               ((20-period)*par_yield$interpolation[14]
                                +(period-14)*par_yield$interpolation[20])/6,
                               interpolation))

par_yield %>% as.data.frame()

```

	year	period	paryield	interpolation
1	0.5	1	5.36	5.360000
2	1.0	2	5.01	5.010000
3	1.5	3	NA	4.815000
4	2.0	4	4.62	4.620000

5	2.5	5	NA	4.520000
6	3.0	6	4.42	4.420000
7	3.5	7	NA	4.380000
8	4.0	8	NA	4.340000
9	4.5	9	NA	4.300000
10	5.0	10	4.26	4.260000
11	5.5	11	NA	4.265000
12	6.0	12	NA	4.270000
13	6.5	13	NA	4.275000
14	7.0	14	4.28	4.280000
15	7.5	15	NA	4.278333
16	8.0	16	NA	4.276667
17	8.5	17	NA	4.275000
18	9.0	18	NA	4.273333
19	9.5	19	NA	4.271667
20	10.0	20	4.27	4.270000

다음으로, Par-yield와 6개월/1년 spot rate를 통해 기간별 spot rate를 boot-strapping 방식으로 산출하도록 하겠습니다.

Par-yield는 채권의 현재가격을 액면가격으로 만들어주는 coupon-rate입니다. 즉, 액면가 100의 채권이 연간 par-yield만큼 쿠폰을 semi-annually 지급한다면 다음과 같이 쓸 수 있습니다.

$$\begin{aligned}
 100 &= \sum_{k=1}^{n-1} \frac{\text{Par}/2}{(1 + \text{Spot}_k/2)^k} + \frac{100 + \text{Par}/2}{(1 + \text{Spot}_n/2)^n} \\
 &\Rightarrow \frac{1}{(1 + \text{Spot}_n/2)^n} = \frac{1}{100 + \text{Par}/2} (100 - \sum_{k=1}^{n-1} \frac{\text{Par}/2}{(1 + \text{Spot}_k/2)^k}) \\
 &\Rightarrow \text{Spot}_n = ((\frac{100 + \text{Par}/2}{100 - \text{Sum of Coupon PV}})^{\frac{1}{n}} - 1) \times 2
 \end{aligned}$$

위 수식을 R을 이용해서 산출하고, 소수점 둘째자리까지 반올림한 결과는 아래와 같습니다.

```

spot <- par_yield %>%
  select(year,period,interpolation) %>%
  mutate(coupon=interpolation/2,
    spot=if_else(period<=2,interpolation,NA)) %>%
  mutate(dc=1/(1+spot/100/2)^period) %>%
  mutate(cum_dc=cumsum(dc)) %>%
  mutate(sum_pvc=coupon*(cum_dc-dc))

for (i in 3:20){
  spot$sum_pvc[i] <- spot$coupon[i]*spot$cum_dc[i-1]
  spot$dc[i] <- (100-spot$sum_pvc[i])/(100+spot$coupon[i])
  spot$spot[i] <- {{(1/spot$dc[i])^(1/spot$period[i])-1}*200}
  spot$cum_dc[i] <- spot$cum_dc[i-1]+spot$dc[i]
}
spot <- spot %>% mutate(spot2=spot %>% round(2))
spot %>% select(year,period,interpolation,spot2)

```

```

# A tibble: 20 x 4
  year period interpolation spot2
  <dbl> <int>        <dbl> <dbl>
1 0.5     1          5.36  5.36
2 1       2          5.01  5.01
3 1.5     3          4.81  4.81
4 2       4          4.62  4.61
5 2.5     5          4.52  4.51
6 3       6          4.42  4.4 
7 3.5     7          4.38  4.36
8 4       8          4.34  4.32
9 4.5     9          4.3   4.28
10 5      10         4.26  4.24
11 5.5    11         4.26  4.25

```

12	6	12	4.27	4.25
13	6.5	13	4.28	4.26
14	7	14	4.28	4.27
15	7.5	15	4.28	4.27
16	8	16	4.28	4.26
17	8.5	17	4.27	4.26
18	9	18	4.27	4.26
19	9.5	19	4.27	4.26
20	10	20	4.27	4.26

### 3. Forward rate

(Question)

위의 문제에서 구한 현물이자율을 이용하여 선도이자율을 구하여라. 이 때 선도이자율은 6개월 구간마다 주어진다. 물론 0 부터 6개월까지의 선도이자율은 6개월 현물이자율과 같다. 선도이자율도 %로 소수점 둘째 자리까지 구하시오.

(Answer)

semi-annually 현물이자율에서 6개월 선도이자율에 대한 수식은 다음과 같습니다.

$$(1 + Spot_{n-1})^{n-1} (1 + Forward_{n-1,n}) = (1 + Spot_n)^n$$

$$\Rightarrow Forward_{n-1,n} = \frac{(1 + Spot_n)^n}{(1 + Spot_{n-1})^{n-1}} - 1$$

위 수식과 (2)의 소수점 둘째자리까지 반올림한 **Spot rate**를 이용하여 산출한 선도이자율은 아래와 같습니다.

```

forward <- spot %>%
  select(year, period, spot, spot2) %>%
  mutate(spot_yield=(1+spot/100/2)^period,
         spot_yield2=(1+spot2/100/2)^period) %>%
  mutate(forward={{spot_yield/lag(spot_yield)-1}*200} %>% round(2),
         forward2={{spot_yield2/lag(spot_yield2)-1}*200} %>% round(2))
forward %>% select(year, period, spot, forward2)

```

```

# A tibble: 20 x 4
  year period   spot forward2
  <dbl> <int> <dbl>     <dbl>
1    0.5      1  5.36     NA
2     1       2  5.01     4.66
3    1.5       3  4.81     4.41
4     2       4  4.61     4.01
5    2.5       5  4.51     4.11
6     3       6  4.40     3.85
7    3.5       7  4.36     4.12
8     4       8  4.32     4.04
9    4.5       9  4.28     3.96
10    5      10  4.24     3.88
11   5.5      11  4.25     4.35
12    6      12  4.25     4.25
13   6.5      13  4.26     4.38
14    7      14  4.27     4.4
15   7.5      15  4.27     4.27
16    8      16  4.26     4.11
17   8.5      17  4.26     4.26
18    9      18  4.26     4.26
19   9.5      19  4.26     4.26
20   10      20  4.26     4.26

```

### 🔥 Spot rate 소수점 자리 문제

Forward rate를 산출할 때, Spot rate를 소수점 두번째 자리까지 반올림하지 않는다면, Spot rate의 오차는 미미하더라도 Forward rate 산출에 영향을 주게 되며, 오차는 기간이 길어질수록 커집니다.

```
forward %>% select(year,period,forward2, forward) %>% filter(year>=8)
```

```
# A tibble: 5 x 4
  year period forward2 forward
  <dbl>   <int>    <dbl>    <dbl>
1     8      16     4.11     4.25
2    8.5      17     4.26     4.24
3     9      18     4.26     4.24
4    9.5      19     4.26     4.23
5    10      20     4.26     4.23
```

## 4. Continuous compounding - Forward rate

(*Question*)

챕터2에서 우리는 선도이자율이 구간별로 주어질 수 있음을 배웠다.

Semi-annual 베이스로 현물이자율을 정의했을 때,  $r_1$ 은 1년 현물이자율,  $r_{1.5}$ 는 1년 6개월 현물이자율을 나타낸다. 그리고 6개월부터 1년까지 기간에 대한 선도이자율은  $(1+r_1/2)^2 = (1+r_{0.5}/2)(1+f_{[0.5,1]}/2)$ 를 만족한다.

현물이자율을 연속복리법으로 정의했을 때, 1년 현물이자율을  $r_1$ 이라 놓으면 시작점에서 1의 투자는  $e^r_1$ 으로 성장한다. 마찬가지로 시간  $t$ 에 대한 현물이자율을  $r_t$ 라 놓으면 시작점에서의 1의 투자는  $e^{tr_t}$ 로 성장한다. 두 시점  $a < b$ 에 대하여 연속복리법에 의한 현물이자율  $r_a, r_b$ 가 주어져 있다. 투자는  $(a, b)$  구간에 적용되는 선도거래를 해결하려고 한다.

이 때 중요한 것은 선도이자율인데, 무차익원리에 의한 선도이자율  $f(a, b)$  를  $a, b, r_a, r_b$  를 이용하여 나타내시오. 물론 연속복리법을 적용한다.

(*Answer*)

연속복리법 하에서,  $f(a, b), a, b, r_a, r_b$  간에는 아래와 같은 식이 성립해야 합니다.

$$e^{a \times r_a} \times e^{(b-a) \times f(a, b)} = e^{b \times r_b}$$

이를 통해  $f(a, b)$ 를 표현하면,

$$\Rightarrow e^{ar_a + (b-a)f(a, b)} = e^{br_b}$$

$$ar_a + (b-a)f(a, b) = br_b \Rightarrow f(a, b) = \frac{br_b - ar_a}{b-a}$$

## 5. Pricing convention - full price

(*Question*)

2024년 3월 15일 호가가 액면 100 당 103-22+로 주어진 미 국채가 있다. 이 채권의 만기는 2029년 10월 15일이고 표면금리는 6.125%이다. 이자는 매 4월 15일, 10월 15일 지급될 때, full price를 구하시오. 소수점 여섯째자리까지 반올림하여 나타내시오.

(*Answer*)

미국 국채의 가격표시방법에 따라, 103-22+의 가격표시는 아래와 같습니다.

$$103 - 22+ = 103 + \frac{22}{32} + \frac{1}{64} = 103.703125$$

```
options(digits=10)
clean_price <- 103+22/32+1/64
clean_price
```

```
[1] 103.703125
```

해당 가격은 호가가격이므로, clean price=103.703125입니다.

경과이자(accrued interest)를 가산해야 Full price를 산출할 수 있습니다.

경과이자는  $\frac{x}{2} \times \frac{1}{—}$ 을 통해 구할 수 있습니다.

경과일은 152일, 이자지급기간은 183일이고, 산출된 경과이자는 약 2.543716입니다.

따라서, full price = clean(quoted) price + accrued interest = **106.246841**입니다.

```
library(lubridate)

accrued_date <- ymd("20240315")-ymd("20231015")
interest_period <- ymd("20240415")-ymd("20231015")
accrued_interest={(100*0.06125)/2}*(as.integer(accrued_date)/as.integer(interest_period))
full_price=round(clean_price+accrued_interest,6)

paste(accrued_date,interest_period,accrued_interest,full_price,sep=" / ")
```

```
[1] "152 / 183 / 2.54371584699454 / 106.246841"
```

# 채권분석 과제3

## Question 1

투자자 A 는 표면금리 7% 미 국채를 액면 1,000 usd 구입하였다. 이 국채의 호가는 101-25+ 이다. 결제일은 2024년 2월 5일이고, 쿠폰 지급일은 매년 4월 15일, 10월 15일이다. 지불한 invoice price(full price)를 센트까지만 구하시오.

## Answer

먼저, 액면 100당 Quote price(clean price)= $101-25+=101+25/32+1/64=101.7969$ 입니다.

다음으로 경과이자(accrued interest)는 “(지난이자지급일로부터 경과일)/이자지급주기\*이자”로 계산할 수 있습니다. (real/real 기반)

경과일은 113일(2023.10.15~2024.02.05), 이자지급주기는 183일(2023-10-15~2024.04.15)이므로, 경과이자는 약 21.6120입니다.

따라서, **Full price**는 약 1039.58입니다.

```
library(tidyverse)

clean <- (101+25/32+1/64)*10
day_interest <- as.integer(ymd(20240415)-ymd(20231015))
day_accrued <- as.integer(ymd(20240205)-ymd(20231015))
accrued <- 35*day_accrued/day_interest
full <- clean+accrued
paste(clean,accrued %>% round(6),full %>% round(6),sep=" / ")
```

```
[1] "1017.96875 / 21.612022 / 1039.580772"
```

## Question 2

시장에서 현물이자율이 1년 4%, 2년 5% 라 가정하자. 그리고 1년 현물이자율의 연 변동(log rate)의 표준편자는 0.005 라 하자. 1년 후 1년 현물이자율의 시나리오를  $r_u, r_d$  라 놓으면 시장 정보를 반영하는 이 두 값을 구하시오.

## Answer

1년 후 현물이자율이 상승하였을 때를  $r_u$ , 하락하였을 때를  $r_d$ 로 나눌 수 있으며 현물이자율의 로그수익률의 연표준편차가 0.005이므로, 아래와 같이 표현할 수 있습니다.

$$r_u = r_d e^{2\sigma} = r_d e^{0.01}$$

한편, 2년 현물이자율에 투자한 zero-coupon bond와 1년 zero-coupon 및 1년 후에 다시 zero-coupon(forward)에 투자한 포트폴리오의 가치는 같아야 하므로,

$$\frac{100}{1.05^2} = \frac{100}{(1.04)(1 + E(r_1))} = 0.5 \times \frac{100}{1.04(1 + r_u)} + 0.5 \times \frac{100}{1.04(1 + r_d)}$$

이분법을 통해 값을 추정하면,  $r_u \approx 6.04\%$ .  $r_d \approx 5.98\%$

```
initial_d <- 0.01; initial_u <- 0.10
r_d <- (initial_u+initial_d)/2; r_u <- r_d*exp(0.01)

while(abs(0.5*100/(1.04*(1+r_u))+0.5*100/(1.04*(1+r_d))-100/1.05^2)>0.001){
  if(0.5*100/(1.04*(1+r_u))+0.5*100/(1.04*(1+r_d))-100/1.05^2>0){initial_d <- r_d}
  if(0.5*100/(1.04*(1+r_u))+0.5*100/(1.04*(1+r_d))-100/1.05^2<=0){initial_u <- r_d}
  r_d <- (initial_u+initial_d)/2
```

```

r_u <- r_d*exp(0.01)
paste(r_u %>% round(4), r_d %>% round(4))

```

[1] "0.0604 0.0598"

### Question 3

2년 만기이고 표면금리가 8% 인 회사채를 가정하자. 표면금리의 지급은 연 1회이다. 그리고 이 회사채는 1년 시점에서 액면가 100 으로 call 할 수 있는 수의상환채이다. 신용위험 때문에 시장 에서는 이 회사채를 위에서 구한 금리모형에 스프레드를 더하여 그 가치를 계산한다.

시장가가 101.9 라 할 때, static spread 와 option-adjusted spread 를 구하여라

### Answer

먼저, 수의상환부회사채의 시장가격 101.9와 문제 2의 zero-rate를 통해 **Static spread**를 산출하면 아래와 같습니다.

$$101.9 = \frac{8}{(1.04 + ss)} + \frac{108}{(1.05 + ss)^2} \Rightarrow ss \approx 1.988\%$$

```

market_price <- 101.9; initial_d <- 0.001; initial_u <- 0.1
ss <- (initial_u+initial_d)/2

while(abs(8/(1.04+ss)+108/(1.05+ss)^2-market_price)>0.00001){
  if(8/(1.04+ss)+108/(1.05+ss)^2-market_price>0){initial_d <- ss}
  if(8/(1.04+ss)+108/(1.05+ss)^2-market_price<=0){initial_u <- ss}
  ss <- (initial_u+initial_d)/2}
ss

```

[1] 0.01988313

다음으로, 문제 2의 금리모형을 이용하여 OAS를 구해보겠습니다.

먼저, OAS를 적용한 할인율은 zero rate + OAS로 표현할 수 있습니다.

문제 2에 따라 1년 zero-rate는 4%, 상승시  $r_u = 0.0604$ , 하락시  $r_d = 0.0598$ 이므로,

$$r_1^* = 0.04 + OAS, r_u^* = 0.0604 + OAS, r_d^* = 0.0598 + OAS$$

수의상환회사채는 1년 후 가격이 액면가보다 높으면 100에 상환되므로,  $r_u^*$ ,  $r_d^*$ 가 8%보다 높은 경우 100에 상환됩니다.

따라서, 수의상환회사채의 금리 상승/하락에 따른 가치평가는 아래와 같습니다.

$$\begin{aligned} 101.9 &= \frac{1}{1+r_1^*} [8 + \frac{1}{2} (\min\{\frac{108}{1+r_u^*}, 100\} + \min\{\frac{108}{1+r_d^*}, 100\})] \\ &= \frac{1}{1.04+OAS} [8 + \frac{1}{2} (\min\{\frac{108}{1.0604+OAS}, 100\} + \min\{\frac{108}{1.0598+OAS}, 100\})] \\ &\Rightarrow OAS \approx 1.978\% \end{aligned}$$

```
initial_d <- 0.001; initial_u <- 0.1
OAS <- (initial_u+initial_d)/2
while(abs((1/(1.04+OAS))*(8+0.5*(min(108/(1+r_u+OAS),100)+min(108/(1+r_d+OAS),100)))-market_price) > 0.001)
  if((1/(1.04+OAS))*(8+0.5*(min(108/(1+r_u+OAS),100)+min(108/(1+r_d+OAS),100)))-market_price) < market_price
    if((1/(1.04+OAS))*(8+0.5*(min(108/(1+r_u+OAS),100)+min(108/(1+r_d+OAS),100)))-market_price) > market_price
      OAS <- (initial_u+initial_d)/2
  }
OAS
```

[1] 0.01978343

### ! Difference in each spreads

Nominal spread, Static spread, Options adjusted spread

1. Nominal spread : 단순히 국채 vs. 회사채의 YTM을 비교
2. Static spread : zero rate +  $\alpha$ 를 계산, 기간구조를 반영하고 있으나 향후 금리변동은 반영하지 않고 단순 현금흐름만 비교하므로, Zero-volatility spread라고도 함
3. Options adjusted spread : 수익상환채의 옵션에 따른 향후 현금흐름 변동가능성을 고려하여 산출. 미래 금리변동으로 인해 옵션이 권리행사되는 경우를 감안하여 산출

## Question 4

PSA 방법에 따르면 PSA 스피드에 따라 월별 CPR이 결정되고, 이는 다시 SMM을 결정한다. 강의노트의 PSA 방법론을 참조하여 다음의 SMM 표를 완성하시오.

월	100 PSA	60 PSA	230 PSA
1			
20			
200			

## Answer

PSA란 연조기상환율(CPR:Conditional Prepayment Rate)이 매달 0.2% 증가하며, 6%까지 증가한 후 만기까지 지속되는 형태의 조기상환모형입니다.

100 PSA가 기준이며, 50 PSA인 경우 매달 0.1%씩 증가 등 비율이 달라지게 됩니다.

다음으로, 월별조기상환율(SMM:Single Monthly Mortality rate)는 CPR을 월환산한 것으로,  $1 - SMM = (1 - CPR)^{\frac{1}{12}}$  입니다.

이에 따라 산출한 (1-SMM) 표는 아래와 같습니다.

월	100 PSA	60 PSA	230 PSA
1	$(1 - 0.2\%)^{\frac{1}{12}}$	$(1 - 0.2\% \times 0.6)^{\frac{1}{12}}$	$(1 - 0.2\% \times 2.3)^{\frac{1}{12}}$
20	$(1 - 4\%)^{\frac{1}{12}}$	$(1 - 4\% \times 0.6)^{\frac{1}{12}}$	$(1 - 4\% \times 2.3)^{\frac{1}{12}}$
200	$(1 - 6\%)^{\frac{1}{12}}$	$(1 - 6\% \times 0.6)^{\frac{1}{12}}$	$(1 - 6\% \times 2.3)^{\frac{1}{12}}$

이를 정리한 SMM 표는 아래와 같습니다.

월	100 PSA	60 PSA	230 PSA
1	0.000167	0.0001	0.000384
20	0.003396	0.002022	0.008010
200	0.005143	0.003051	0.012299

## Question 5

다음의 CMO 는 9% 금리를 갖는 모기지 풀을 담보로 가지고 있다.

tranche	par(m USD)	coupon rate(%)
A	300	7
B	200	6.75
C	200	7.25
D	250	7.75

투자자가 금리 9% 의 notional IO 투자를 원한다고 하자. 해당 상품의 액면가를 계산하시오.

## Answer

먼저, CMO의 담보에서는 9%의 현금흐름이 발생하는 반면 각 tranche의 쿠폰은 9% 미만이므로 초과이자가 \$17.125m만큼 발생합니다.

이 초과이자분으로 금리 9%의 원금 없이 이자만 지급하는 구조화채권(Nominal IO)을 설계한다면, 액면가(명목금액)은 9%의 이자로 CMO의 초과수익인 \$17.125m를 지급해야합니다.

따라서, 액면가(명목금액)는  $\$17.125m / 9\% = \$190.2778m$ 입니다.

```
total_inflow=(300+200+200+250)*0.09  
total_outflow=300*0.07+200*0.0675+200*0.0725+250*0.0775  
excess_interest=total_inflow-total_outflow  
notional=excess_interest/0.09  
paste(excess_interest, notional, sep=" / ")
```

```
[1] "17.125 / 190.277777777778"
```

## **Part III**

### **투자분석('24봄)**

# 투자분석 과제1

## Question

You are a stock analyst and want to evaluate stocks. You may select any 3 stocks and get information from data source you prefer. You need a risk-free rate and market risk premium for this assignment. Choose appropriate risk-free rate such as 3-month T-bill rate or 1-month LIBOR and risk premium for the market portfolio. Use the constant growth dividend discount model. You may use any computer tool but do not hand in the data or the program code.

- a) What are the names of stocks you choose?
- b) Specify risk-free rate, risk premium and the data source.
- c) What are the required rates of return? Use the CAPM. You need to specify firm's beta and the expected rate of return of the market index portfolio.
- d) Calculate present value of growth opportunity (PVGO) for each stock.
- e) Find the intrinsic value (0) and 1-year value (1) for each stock.
- f) Find the expected rate of return for each stock, i.e.,  $(1 - 0)/0$  where 0 is the current market price.
- g) Based on the calculation above, which stock do you buy or sell? Explain briefly

## 문제풀이

### (a) 주식 선정

저는 *CME group(CME)*, *ICE(ICE)*, *Nasdaq(NDAQ)* 세가지 종목을 선정하였습니다.

선정 배경으로는,

- (1) 제가 거래소 산업에 관심이 많고,
- (2) 세 주식 모두 미국에 상장되어있는 대표적인 글로벌 거래소이며,
- (3) 동일한 거래소 산업이고 S&P500지수의 구성종목이라 동일 선상에서 비교하기 적합할 것으로 보이기 때문입니다.

관련 자료는 [Yahoo Finance](#)를 참조하였습니다.

### (b) 무위험이자율 및 리스크프리미엄

risk-free rate는 최근 미국에서 대체지표금리로 선정한 SOFR 기반의 금리를 누적하여 3개월 금리로 환산한 3-month SOFR Average를 risk-free rate로 선정하였습니다.

[New York Fed : SOFR Average](#)

각 주식과 시장(S&P500)의 리스크프리미엄은 과거 3년간('21~'23)의 연평균수익률 및 무위험이자율을 이용하여 산출할 예정입니다.

각 데이터를 취합하여 정리한 결과 및 무위험이자율, 리스크프리미엄의 산출은 아래와 같습니다.

```
# import data&library
rm(list = ls())
library(tidyverse)
nasdaq <- read_csv("investment_hw/투자분석_NDAQ.csv")
cme <- read_csv("investment_hw/투자분석_CME.csv")
ice <- read_csv("investment_hw/투자분석_ICE.csv")
rf <- read_csv("investment_hw/투자분석_sofrai.csv")
market <- read_csv("investment_hw/투자분석_snp500.csv")

# tidy data
rf2 <- rf %>%
  filter(`90-Day Average SOFR` %>% is.na()==FALSE) %>%
  mutate(date=paste0(substr(`Effective Date`, 7, 10),
```

```

        substr(`Effective Date`,1,2),
        substr(`Effective Date`,4,5)) %>% as.integer(),
year=substr(`Effective Date`,7,10),
month=substr(`Effective Date`,1,2),
group="rfr",
price=`90-Day Average SOFR`) %>%
select(year,month,date,group,price)

# Data Pre-processing
tidydata <- tibble()
tidydata <- nasdaq %>%
  mutate(group="nasdaq") %>%
  union_all(cme %>% mutate(group="cme")) %>%
  union_all(ice %>% mutate(group="ice")) %>%
  union_all(market %>% mutate(group="market")) %>%
  mutate(date=paste0(substr(Date,1,4),
                     substr(Date,6,7),
                     substr(Date,9,10)) %>% as.integer(),
         year=substr(Date,1,4),
         month=substr(Date,6,7),
         price=`Adj Close`) %>%
  select(year,month,date,group,price) %>%
  union_all(rf2) %>%
  arrange(date,group)

# Use daily adj. closing price, if rfr is NA then use yesterday rfr.
tidydata_wide <- tidydata %>%
  pivot_wider(names_from = "group", values_from = "price") %>%
  filter(nasdaq %>% is.na()==FALSE,
         cme %>% is.na()==FALSE,

```

```

ice %>% is.na()==FALSE,
market %>% is.na()==FALSE) %>%
mutate(rfr=if_else(rfr %>% is.na(),lag(rfr)/100,rfr/100)) %>%
mutate(cme_yield=log(cme)-log(lag(cme)),
ice_yield=log(ice)-log(lag(ice)),
nasdaq_yield=log(nasdaq)-log(lag(nasdaq)),
market_yield=log(market)-log(lag(market))) %>%
filter(cme_yield %>% is.na()==FALSE)

```

```

rf3 <- tidydata_wide %>%
filter(year!=2020&year!=2024,as.integer(month)%%3==0) %>%
arrange(date %>% desc()) %>%
group_by(year,month) %>%
slice(1) %>%
ungroup() %>%
summarise(rfr_3year=mean(rfr))

tidydata_wide %>% filter(date==20231229) %>% select(rfr) %>% bind_cols(rf3)

```

```

# A tibble: 1 x 2
  rfr rfr_3year
  <dbl>     <dbl>
1 0.0536    0.0223

```

```

tidydata_wide %>%
filter(year!=2020&year!=2024) %>%
summarise(cme_premium=sum(cme_yield)/3-rf3$rfr_3year,
ice_premium=sum(ice_yield)/3-rf3$rfr_3year,
nasdaq_premium=sum(nasdaq_yield)/3-rf3$rfr_3year,
market_premium=sum(market_yield)/3-rf3$rfr_3year)

```

```
# A tibble: 1 x 4
  cme_premium ice_premium nasdaq_premium market_premium
  <dbl>        <dbl>        <dbl>        <dbl>
1     0.0679      0.0272      0.0826      0.0573
```

무위험이자율은 기간말 기준 5.36%, 과거 3개년 동안 무위험이자율에 투자한 연평균수익률은 약 2.23%, 과거 3개년 동안 실현된 각 주식과 시장의 연환산 리스크프리미엄은 아래 표와 같습니다.

구 분	S&P500	CME	ICE	NASDAQ
연수익률	7.96%	9.03%%	4.95%	10.49%
프리미엄	5.74%	6.81%%	2.73%	8.27%

### ⚠ Excess return vs. Risk premium

초과수익률은 과거의 값을 바탕으로 산출되며, 리스크프리미엄은 기대수익률을 기반으로 산출. 위의 풀이는 초과수익률을 산출하였으므로 오답이며, 리스크프리미엄은 CAPM 등의 기대수익률을 가지고와서 산출해야 함.

$$\text{Excess return} = r - r_f$$

$$\text{Risk premium} = E[r] - r_f$$

## (c) CAPM

각 종목의 베타를 산출하기 위해 필요한 파라미터는 아래와 같습니다.

- 무위험이자율,  $R_f$  : 기간말('23.12월말)의 3-month SOFR average
- 시장수익률,  $R_m$  : S&P500지수의 수익률
- 개별수익률,  $R_{stock}$  : 개별 주식의 수익률

수익률은 일간/월간 두 가지를 사용하여 비교할 예정이며, 회귀분석에 사용한 값은 아래와 같습니다.

- 기간 : (일별) 2023년 250거래일, (월간) 2021년~2023년 36개월

- 산출방법 : (일별) 전일 종가 대비 당일 종가의 로그수익률, (월별) 이전달 말 종가 대비 이번달 말 종가의 로그수익률

마지막으로, 베타를 이용하여 기대수익률을 산출할 때 이용한  $R_m$ 은 다음과 같습니다.

- (일별) 250일간 S&P500지수 일평균수익률의 연환산(=연수익률)
- (월별) 36개월간 S&P500지수 월평균수익률의 연환산(=3개년수익률/3)

### ⚠ Warning

로그수익률(연속복리수익률)을 사용하였으므로 보유기간수익률로 표현할 때 단순 덧셈만으로 환산할 수 있습니다.

또한, 기간 중 배당 및 CA로 인한 차이는 수정주가(Adj. close price)를 사용해서 보정하였습니다.

이에 따라, R 코드로 산출한 개별종목별 베타 / 기대수익률 / 산출기간동안 실현수익률은 아래와 같습니다.

```
# calcuate beta, required return

# daily beta in '23
dailybeta <- tidydata_wide %>%
  filter(year=="2023") %>%
  summarise(beta_cme=cov(market_yield,cme_yield)/var(market_yield),
            beta_ice=cov(market_yield,ice_yield)/var(market_yield),
            beta_nasdaq=cov(market_yield,nasdaq_yield)/var(market_yield),
            market_avg_yield=sum(market_yield),
            cme_avg_yield=sum(cme_yield),
            ice_avg_yield=sum(ice_yield),
            nasdaq_avg_yield=sum(nasdaq_yield)) %>%
  mutate(group="daily")

# monthly beta
monthlybeta <- tidydata_wide %>%
```

```

arrange(year,month,date %>% desc) %>%
group_by(year,month) %>%
slice(1) %>%
ungroup() %>%
mutate(cme_yield=log(cme)-log(lag(cme)),
       ice_yield=log(ice)-log(lag(ice)),
       nasdaq_yield=log(nasdaq)-log(lag(nasdaq)),
       market_yield=log(market)-log(lag(market))) %>%
filter(cme_yield %>% is.na()==FALSE) %>%
filter(year!="2024",year!="2020") %>%
summarise(beta_cme=cov(market_yield,cme_yield)/var(market_yield),
           beta_ice=cov(market_yield,ice_yield)/var(market_yield),
           beta_nasdaq=cov(market_yield,nasdaq_yield)/var(market_yield),
           market_avg_yield=sum(market_yield)/3,
           cme_avg_yield=sum(cme_yield)/3,
           ice_avg_yield=sum(ice_yield)/3,
           nasdaq_avg_yield=sum(nasdaq_yield)/3) %>%
mutate(group="monthly")

beta <- tibble()
beta <- dailybeta %>%
union_all(monthlybeta) %>%
mutate(tidydata_wide %>% filter(date==20231229) %>% select(rfr)) %>%
select(group,market_avg_yield,cme_avg_yield,ice_avg_yield,nasdaq_avg_yield,
       rfr,beta_cme,beta_ice,beta_nasdaq)

capm <- tibble()
capm <- beta %>%
pivot_longer(cols = c("beta_cme","beta_ice","beta_nasdaq"),
             names_to = "stock", values_to = "beta") %>%

```

```

  mutate(expected_return=beta*(market_avg_yield-rfr)+rfr) %>%
  mutate(stock=substr(stock,6,nchar(stock))) %>%
  mutate(realized_return=if_else(stock=="cme",cme_avg_yield,
                                 if_else(stock=="ice",ice_avg_yield,nasdaq_avg_yield))) %>%
  select(group,stock,beta,expected_return,realized_return,market_avg_yield,rfr)

```

capm

```

# A tibble: 6 x 7
  group   stock   beta expected_return realized_return market_avg_yield     rfr
  <chr>  <chr>   <dbl>        <dbl>          <dbl>        <dbl>    <dbl>
1 daily    cme    0.315      0.105       0.272      0.217  0.0536
2 daily    ice     0.828      0.189       0.240      0.217  0.0536
3 daily  nasdaq  0.866      0.195      -0.0373     0.217  0.0536
4 monthly  cme    0.353      0.0628      0.0903     0.0796 0.0536
5 monthly  ice     1.12       0.0829      0.0495     0.0796 0.0536
6 monthly nasdaq 0.993      0.0795      0.105      0.0796 0.0536

```

일별/월별로 구분한 각 주식의 베타와 기대수익률입니다.

보다 일반적이고 평탄화된 결과를 위해 월별 데이터를 중심으로 설명하겠습니다.

먼저 각 주식의 베타는 **CME:0.35, ICE:1.12, Nasdaq 0.99**입니다. 이는 거래소산업이 특성상 변동성이 크지 않고 경제상황에 민감하지 않아 베타가 시장포트폴리오와 유사한 1 수준에서 형성되는 것으로 보입니다. 특히, CME는 세계 최대의 파생상품거래소로서 규모가 매우 큰 고배당 우량주인 특징이 있어 변동성이 낮아 베타가 0.35로 낮게 형성된 것으로 추정됩니다.

### 베타 산출방법

베타는  $R_{stock,t} = \alpha + \beta R_{m,t} + \epsilon$ 의 선형회귀분석 결과로 산출되는 계수입니다.

즉,  $\beta = \rho_{stock,m} \frac{\sigma_{stock}}{\sigma_m} = \frac{Cov_{stock,m}}{\sigma_m^2}$ 로 산출할 수 있습니다.

월별수익률을 산출한 과거 3년(2021~2023)간 평균월별수익률을 연율화하면 로그수익률 특성상 지난 3년간 연평균수익률과 같습니다. 즉, 시장포트폴리오의 기대수익률은 약 7.96%라고 할 수 있으며, 무위험 이자율은 2023.12월말 값인 5.36%를 사용합니다.

이제,  $R_{stock} = \beta(R_m - R_f) + R_f$ 를 통해 각 주식의 요구수익률을 산출할 수 있습니다.

구 분	CME	ICE	NASDAQ
Beta	0.35	1.12	0.99
Required Return	6.28%	8.29%	7.95%

#### (d) PVGO

먼저, 거래소산업은 성장주보다는 우량주 성향이 강하고, 배당성향이 다소 높고 정기적인 배당을 하는 경향이 있습니다.. 따라서 배당할인모형(Constant-growth DDM)을 사용할 예정입니다.

먼저, 기업의 내재가치에 배당할인모형을 적용한 PVGO의 산식은 아래와 같습니다.

$$PVGO = V_0 - \frac{E_1}{k} = \frac{D_1}{k-g} - \frac{E_1}{k}$$

여기서, 요구수익률  $k$ 는 (c)에서 CAPM으로 구한 요구수익률을 사용할 예정이며,  $D_1, E_1, g$ 를 구하기 위해 직전주당배당( $D_0$ ), 직전 주당수익( $E_0$ ), 직전재투자성향( $b_0 = \frac{E_0 - D_0}{E_0}$ ), 직전 자기자본수익률  $ROE_0$ 를 이용하여 다음과 같이 산출하도록 하겠습니다.

$$g = b_0 \times ROE_0 = \frac{E_0 - D_0}{E_0} \times ROE_0, \quad D_1 = D_0(1+g), \quad E_1 = E_0(1+g)$$

$$\Rightarrow PVGO = \frac{D_0(1+g)}{k-g} - \frac{E_0(1+g)}{k}$$

#### i 주당순이익 산출 : DDM

다음 년도의 주당순이익  $E_1$ 은 올해의 순이익  $E_0$ 에 배당할인모형의 배당성장률  $g$ 를 반영하여 산출하였습니다.

이제, 각 주식의 재무상태표를 참조하여  $D_0$ ,  $E_0$ ,  $b_0$ ,  $ROE_0$ 를 산출하겠습니다. 다만, 직전사업연도(2023)는 시장벤치마크(S&P500)의 수익률이 약 21%로, 과거 평균치의 약 3배를 상회하고 있어 각 기업의 EPS, DPS 등이 평소 수치보다 고평가되었을 가능성성이 있습니다. 이를 감안하여  $D_0$ ,  $E_0$  산출시에는 과거 3개년 당기순이익, 현금배당금의 산술평균을 사용할 것이며  $ROE_0$ 는 과거 3개년의 기하평균을 사용하겠습니다.

각 값들은 Yahoo Finance 및 Stockrow를 참조하였습니다.

```
financial <- read_csv("investment_hw/투자분석_financialstatement.csv")
```

```
# DDM 0|용
financial <- financial %>%
  group_by(stock) %>%
  summarise(eps=sum(earning)/sum(shares),
            dps=sum(dividend)/sum(shares),
            bps=sum(capital)/sum(shares),
            roe=prod(1+roe)^(1/3)-1) %>%
  mutate(reinvestment=(eps-dps)/eps) %>%
  mutate(growth=roe*reinvestment) %>%
  mutate(d1=dps*(1+growth),
        e1=eps*(1+growth)) %>%
  left_join(., capm %>% filter(group=="monthly")) %>% select(stock, expected_return, rfr),
  by="stock") %>%
  mutate(v0_ddm=d1/(expected_return-growth)) %>%
  mutate(pvgo_ddm=v0_ddm-e1/expected_return)
financial
```

```
# A tibble: 3 x 13
  stock    eps    dps    bps    roe reinvestment    growth     d1      e1 expected_return
  <chr>   <dbl>  <dbl>  <dbl>  <dbl>           <dbl>    <dbl>  <dbl>  <dbl>           <dbl>
1 cme     7.86  7.49   75.3  0.105       0.0475  0.00497 7.53   7.90          0.0628
2 ice     4.67  1.52   42.3  0.116       0.675   0.0786  1.64   5.04          0.0829
```

```

3 nasd~ 2.25 0.785 15.6 0.163      0.652 0.106 0.868 2.49      0.0795
# i 3 more variables: rfr <dbl>, v0_ddm <dbl>, pvgo_ddm <dbl>

```

산출 결과는 아래와 같습니다.

구 분	CME	ICE	NASDAQ
$E_0$	7.86	4.67	2.25
$D_0$	7.49	1.52	0.79
$ROE_0$	10.5%	11.6%	16.3%
$b_0$	4.7%	67.5%	65.2%
$g$	0.5%	7.9%	10.6%
$D_1$	7.53	1.64	0.87
$E_1$	7.90	5.04	2.49
$k$	6.28%	8.29%	7.95%
$V_0$	130.15	380.07	-32.82(NaN)
$PVGO$	4.32	319.28	-64.19(NaN)

배당할인모형을 사용하여 평가해본 결과, PVGO는 CME가 **4.32\$**, ICE가 **319.28\$**, NASDAQ은 **-64.19\$**로 산출되었습니다. 여기서 NASDAQ 거래소의 경우 기업의 요구수익률보다 성장률이 높아 ( $k < g$ ) 기업의 내재가치( $V_0$ ) 평가에 배당할인모형을 사용할 수 없는 케이스입니다. 즉, PVGO가 정상적으로 산출되지 않았습니다.

NASDAQ 거래소의 요구수익률은 7.95% 및 성장률은 10.6%로 산출되었는데, 현재시점에서 배당할인모형을 통해 이 주식을 평가한다는 것은 기업에 요구되는 수익률보다 성장하는 속도가 높아서 기업의 미래 배당가치가 발산( $\rightarrow \infty$ )한다는 것을 의미합니다. 이러한 이유가 발생하는 이유를 추정해보면, 아마도 NASDAQ의 현재주가가 저평가되어있거나, 최근 성과가 좋아서 단기적으로  $ROE_0$ 가 높게 산출되었거나, CAPM의 요구수익률이 실제 주식의 요구수익률보다 낮기 때문일 것 입니다.

NASDAQ 거래소의 과거 추이를 볼 때, 최근 3년간 실현수익이 연평균 약 10.5%로 요구수익률보다 높고 성장률과 유사하므로 CAPM의 요구수익률이 실제 주식의 요구수익률을 정확하게 평가하지 못한 것이 주된 이유일 것으로 보입니다.

이러한 문제점은,

- (1) CAPM 대신 APT 등의 다른 모형으로 요구수익률( $k$ )을 평가해보거나,
- (2) 배당할인모형(DDM) 대신 다른 모형을 이용함으로써 해결할 수 있습니다.

### **i CAPM의 한계**

CAPM으로 산출한 요구수익률  $k$ 는 개별주식의 여러 특성을 정확히 반영하지 못하였을 가능성이 있습니다. CAPM은 시장에 대한 민감도  $\beta$ 와 시장포트폴리오의 리스크 대비 수익률( $R_m - R_f$ )을 이용해 주가의 기대수익률(요구수익률)을 평가하는데, 이 한가지 요인(factor)으로 기대수익률이 결정된다고 가정하기 때문입니다.

Multi-factor 모형은 배우기 이전이므로 수업시간에 배운 초과이익모형(**Residual Income Model**)을 이용하여 세 주식의  $V_0$ ,  $PVGO$ 를 다시 평가해보도록 하겠습니다.

먼저, RIM의 산식은 다음과 같습니다.

$$V_0 = BE_0 + \sum_{i=1}^{\infty} \frac{E[RE_i]}{(1+k)^i}$$

여기서  $BE_0$ 은 주당순자산으로  $D_0$ ,  $E_0$ 과 같이 과거 3개년 평균치를 사용하고, 초과이익의 기대값  $E[RE_i]$ 는  $i$ 년도의 주당순이익( $E_i$ )-주당배당금( $D_i$ )으로 결정되며, 주당순이익은  $E_i = ROE_0 \times BE_{i-1}$ , 주당배당금은  $k \times BE_{i-1}$ 로 산출,  $BE_i = BE_{i-1} \times (1 + R_f)$ (즉, 초과이익이 무위험이자율만큼 성장)하도록 하겠습니다.

$$V_0 = BE_0 + \sum_{i=1}^{\infty} \frac{(ROE_0 - k)BE_0(1 + r_f)^i}{(1 + k)^i} = BE_0 + \frac{(ROE_0 - k)BE_0}{k - r_f}$$

위와 같이 RIM으로 기업의 내재가치를 산출할 수 있으며, 산출결과는 아래와 같습니다.

```
financial_rim <- financial %>%
  mutate(v0_rim=bps+(roe-expected_return)*bps/(expected_return-rfr)) %>%
  mutate(pvgo_rim=v0_rim-(eps*(1+rfr))/expected_return) %>%
  select(stock,bps,roe,rfr,expected_return,v0_rim,pvgo_rim)
```

```
financial_rim
```

```
# A tibble: 3 x 7
  stock     bps     roe     rfr expected_return v0_rim pvgo_rim
  <chr>   <dbl>   <dbl>   <dbl>        <dbl>    <dbl>    <dbl>
1 cme      75.3  0.105  0.0536     0.0628   417.    285.
2 ice      42.3  0.116  0.0536     0.0829   90.5    31.2
3 nasdaq   15.6  0.163  0.0536     0.0795   65.8    35.9
```

**i** 주당순이익 산출 : RIM

다음 연도의 주당순이익  $E_1$ 은 올해의 순이익  $E_0$ 에 초과이익성장률인 무위험이자율만큼 가산하여 산출하였습니다.

구분	CME	ICE	NASDAQ
$V_0, DDM$	130.15	380.07	-
$PVGO, DDM$	4.32	319.28	-
$V_0, RIM$	417.10	90.55	65.81
$PVGO, RIM$	291.86	34.20	37.43

RIM 방식으로 내재가치를 평가한 결과, PVGO의 값은 CME가 291.86\$, ICE가 34.20\$, NASDAQ에서 37.43\$로 산출되었습니다.

**(e) Find  $V_0$ ,  $V_1$**

제가 사용한 RIM 산출식에서  $V_1 = V_0(1 + r_f)$ 입니다. (DDM은  $V_1 = V_0(1 + g)$ )

NASDAQ에서 DDM은 사용할 수 없으므로 RIM을 기준으로 산출하겠습니다.

```
financial_rim %>%
  mutate(v1_rim=v0_rim*(1+rfr)) %>%
  select(stock,v0_rim,v1_rim)
```

```
# A tibble: 3 x 3
  stock   v0_rim v1_rim
  <chr>    <dbl>  <dbl>
1 cme      417.   439.
2 ice      90.5   95.4
3 nasdaq   65.8   69.3
```

기존에 산출하였던  $V_0$ 와 산출한  $V_1$ 은 아래와 같습니다.

구 분	CME	ICE	NASDAQ
$V_0, RIM$	417.10	90.55	65.81
$V_1, RIM$	439.44	95.40	69.33

#### (f) Find the expected rate of return for each stock

```
stock_return <- financial_rim %>%
  mutate(v1_rim=v0_rim*(1+rfr)) %>%
  select(stock,v1_rim) %>%
  left_join(.,tidydata %>% filter(date==20240228) %>% select(group,price),
            by=c('stock'='group')) %>%
  mutate(return=(v1_rim-price)/price)
stock_return
```

```
# A tibble: 3 x 4
  stock   v1_rim price return
  <chr>    <dbl> <dbl>  <dbl>
1 cme      439.   220.   1.00
2 ice      95.4   138.  -0.311
3 nasdaq   69.3   56.1   0.236
```

2월말 기준 각 주식의 종가를 기준으로 계산한 연환산 기대수익률은 다음과 같습니다.

구 분	CME	ICE	NASDAQ
$V_1, RIM$	439.44	95.40	69.33
$ClosePrice$	219.68	138.39	56.11
$ExpectedReturn, annum$	100.03%	-31.06%	23.55%

### (g) Which stock is the best?

#### NASDAQ exchange 주식

단순히 (f)의 기대수익률로 볼 때는 CME가 가장 높지만, 이는 BPS가 높고 배당성향이 1에 가까운 CME 거래소의 특성상 RIM 방식에서 높은 내재가치를 평가받은 것으로 보입니다. 이는 DDM 방식으로 평가하였을 때, 성장률  $g$ 가 매우 낮아 내재가치가 다소 낮게 산출된 점에서 잘 드러납니다.

또한, CME는 시장 베타도 0.35수준으로 낮아 절대적인 요구수익률이 높지 않고 변동성도 낮아 흔히 말하는 재미없는 주식입니다. 장기적인 투자를 선호하고 자본금이 많은 기관투자자 등이 주로 투자하는 주식일 것으로 예상됩니다.

반면에 NASDAQ은 RIM방식으로 평가한 기대수익률이 연 23.55% 수준으로 양호하고, 시장 베타도 거의 1에 가까워 적당한 변동성을 가지고 있습니다. DDM방식에서는 평가가 불가능하였으나, 이 의미는 ROE와 배당성향이 높아 기대되는 배당성장률  $g$ 가 CAPM의 요구수익률  $k$ 보다 높다는 뜻입니다. 즉, 초과이익을 발생시켜 시장을 outperform할 확률이 높다고 생각하였습니다.

즉, 배당성향이 매우 높고 변동성이 낮아 안정적인 우량주인 CME보다는 최근 성과가 훌륭하여 성장성이 있으며, 안정적으로 요구수익률 대비 초과이익을 달성할 것으로 예상되고, 적정 수준의 변동성을 지닌 **NASDAQ** 거래소이 투자대상으로 가장 적합하다고 판단됩니다.

한편, ICE는 DDM에서는 내재가치가 좋게 평가되었으나 RIM에서는 낮게 평가되어 결국 기대수익률이 (-)되는 분석결과를 고려하여 선택하지 않았습니다.



## 투자분석 과제2

### Problems

You are a portfolio manager and want to create a portfolio. Choose any three stocks which are traded in the same currency. market. Collect monthly returns of stock prices for 5 years at least. You need to have a risk-free rate for this assignment. Assume that a short-sale is not allowed. You may use any computer tool but do not hand in the data or the program code.

- a) What are the names of the stocks you have chosen? Mention the database to provide data.
- b) Find annualized arithmetic average and variance of each stock's excess return and variance-covariance matrix of 3 stocks' excess returns.
- c) Assume the expected rate of return and future standard deviation (SD) are same as historical arithmetic average and SD for each stock. Plot investment opportunity set for the three stocks. Explain briefly how you do it.
- d) Find the Tangent portfolio and Capital Allocation Line.
- e) By some reason, you should hold more than 30% of one stock among three stocks. (Select one stock and assign weight more of 30% to the stock. Then the weights for other stocks do not exceed 70%). How does the efficient frontier change? What is the optimal portfolio that maximizes the Sharpe ratio? What are the expected return, the standard deviation, and the Sharpe ratio of the optimal portfolio? Explain why you would have this portfolio.
- f) You are worried about a worst scenario. What is the (monthly) 5% VaR of the optimal portfolio in e) under the normality assumption? What is the 5% expected shortfall without the normality and explain how you compute it.
- g) Provide suggestions on the proportion of risk-free assets to invest in when you choose the optimal risky portfolio in e). You may consider three values for the coefficient of risk aversion: 2, 3.5 and 5.

## Answer

(a)

저는 *CME group(CME), ICE(ICE), Nasdaq(NDAQ)K* 세가지 종목을 선정하였습니다.

선정 배경으로는,

- (1) 제가 거래소 산업에 관심이 많고,
- (2) 세 주식 모두 미국에 상장되어있는 대표적인 글로벌 거래소이며,
- (3) 동일한 거래소 산업이고 S&P500지수의 구성종목이라 동일 선상에서 비교하기 적합할 것으로 보이기 때문입니다.

세 주식의 일별수정주가(Adj. close), 벤치마크지수인 S&P500지수의 일별수정가격, 무위험이자율로 채택한 미연준의 일별 Effective-FFR(Federal Funds Rate)를 활용하였습니다.

기간 : 직전 10년(2014.4 ~ 2024.3)

출처 : [Yahoo Finance] (<https://finance.yahoo.com/>)

산출방법 :

- (월수익률) 지난달 말 대비 월말 수익률
- (월초과수익률) 월수익률 - 월말FFR/12
- (월분산) 월/월초과수익률의 표본표준편차
- (평균수익률) 월/월초과수익률을 산술평균하여 연환산 (x12)
- (평균분산) 월분산을 산술평균하여 연환산 (x12)

```
library(tidyverse)
# stocks and market index S&P500 from yahoo finance
cme <- read_csv("investment_hw/cme.csv") %>% tibble()
ndaq <- read_csv("investment_hw/ndaq.csv") %>% tibble()
ice <- read_csv("investment_hw/ice.csv") %>% tibble()
spx <- read_csv("investment_hw/spx.csv") %>% tibble()
# risk-free rate is effective-FFR(federal funds rate)
ffr <- read_csv("investment_hw/fedfunds.csv") %>% tibble()
```

```
# set period 10years
strt_dd='20140101'; end_dd='20240331'
```

(b)

```
# tidy data
raw_data <- tibble()

raw_data <- cme %>% mutate(cme=`Adj Close`) %>% select(Date,cme) %>%
  left_join(ndaq %>% mutate(ndaq=`Adj Close`) %>% select(Date,ndaq)) %>%
  left_join(ice %>% mutate(ice=`Adj Close`) %>% select(Date,ice)) %>%
  left_join(spx %>% mutate(spx=`Adj Close`) %>% select(Date,spx)) %>%
  mutate(day=gsub("-", "", Date)) %>%
  mutate(year=substr(day, 1, 4)) %>%
  mutate(month=substr(day, 1, 6)) %>%
  filter(day>=strt_dd, day<=end_dd) %>%
  left_join(ffr %>%
    mutate(ffr=FEDFUNDS/100,
          month=substr(gsub("-", "", as.character(DATE)), 1, 6)) %>%
    select(month,ffr)) %>%
  select(year,month,day,cme,ndaq,ice,spx,ffr,Date)

# using monthly return
monthly_raw <- tibble()

monthly_raw <- raw_data %>%
  group_by(year,month) %>%
  arrange(day %>% desc()) %>%
  slice(1) %>%
  pivot_longer(.,c("cme","ndaq","ice","spx"),
              names_to = "name",values_to = "price") %>%
  ungroup() %>%
  arrange(name,year,month) %>%
  mutate(return=price/lag(price)-1) %>% # monthly return
```

```

    mutate(excess_return=return-ffr*(1/12)) %>%
    filter(as.integer(month)>=201404)

# calculate arithmetic mean&var and var-cov matrix of excess return of stocks
monthly_stat <- tibble()
monthly_stat <- monthly_raw %>%
  ungroup() %>%
  group_by(name) %>%
  summarise(avg_excess=mean(excess_return)*12,
            var_excess=sd(excess_return)^2*12,
            avg_return=mean(return)*12,
            var_return=sd(return)^2*12,
            avg_rf=mean(ffr)) # Annualized

# Cov matrix
variance_matrix <- monthly_raw %>%
  ungroup() %>%
  select(month,name,excess_return) %>%
  pivot_wider(names_from = "name", values_from = "excess_return") %>%
  select(cme,ice,ndaq,spx) %>%
  var()

# Corr matrix
cor_matrix <- monthly_raw %>%
  ungroup() %>%
  select(month,name,excess_return) %>%
  pivot_wider(names_from = "name", values_from = "excess_return") %>%
  select(cme,ice,ndaq,spx) %>%
  cor()

```

위 방법을 기준으로 산출한 각 주식의 평균초과수익률(연환산) 및 공분산행렬은 아래와 같습니다.

Excess return : CME 15.0%, ICE 14.4%, Nasdaq 18.8%

```
monthly_stat
```

```
# A tibble: 4 x 6
  name avg_excess var_excess avg_return var_return avg_rf
  <chr>    <dbl>      <dbl>     <dbl>      <dbl>    <dbl>
1 cme      0.150     0.0340    0.165     0.0340  0.0141
2 ice      0.144     0.0414    0.158     0.0415  0.0141
3 ndaq     0.188     0.0415    0.202     0.0413  0.0141
4 spx      0.101     0.0229    0.115     0.0229  0.0141
```

### Variance-Covariance Matrix of three stocks&Benchmark index

```
variance_matrix
```

```
          cme         ice        ndaq        spx
cme  0.0028310476 0.001767299 0.001138796 0.0008194417
ice  0.0017672988 0.003452184 0.002149990 0.0016652315
ndaq 0.0011387956 0.002149990 0.003456322 0.0016308630
spx  0.0008194417 0.001665231 0.001630863 0.0019071007
```

### Correlation Matrix of three stocks&Benchmark index

```
cor_matrix
```

```
          cme         ice        ndaq        spx
cme  1.0000000 0.5653136 0.3640534 0.3526612
ice  0.5653136 1.0000000 0.6224185 0.6489942
ndaq 0.3640534 0.6224185 1.0000000 0.6352191
spx  0.3526612 0.6489942 0.6352191 1.0000000
```

(c)

각 주식의 기대수익률과 미래변동성이 과거실현값과 동일하다고 가정하겠습니다.

공매도는 없다고 가정하였으므로 각 주식의 비중이 양수가 되도록 설정하고, 먼저 두개의 주식으로 구성된 포트폴리오의 efficient frontier를 각각 구하도록 하겠습니다.

그 다음, 하나의 주식과 다른 두 주식의 포트폴리오를 재결합하면 새로운 포트폴리오를 구성할 수 있으며, 이를 반복하여 efficient frontier를 도식화할 수 있습니다.

아래 코드에서는 ICE의 구성비중을 5%, 10%, … , 95%로 고정하고 CME-Nasdaq 포트폴리오와 재결합을 반복하여 efficient frontier를 구현하였습니다.

```
# Portfolio return/vol of variance combinations three stocks
portfolio <- tibble()
portfolio <- tibble(cme=seq(0,1,0.002),ice=seq(1,0,-0.002)) %>%
  union_all(.,tibble(cme=seq(0,1,0.002),ice=0)) %>%
  union_all(.,tibble(cme=0,ice=seq(1,0,-0.002)))

for(i in seq(0.05,0.95,0.05)){portfolio <- portfolio %>%
  union_all(.,tibble(cme=seq(0,1-i,0.005),ice=i))}
for(i in seq(0.25,0.30,0.01)){portfolio <- portfolio %>%
  union_all(.,tibble(cme=seq(0,1-i,0.005),ice=i)})

portfolio <- portfolio %>%
  mutate(ndaq=1-cme-ice) %>%
  mutate(return=cme*monthly_stat$avg_return[1]
    +ice*monthly_stat$avg_return[2]
    +ndaq*monthly_stat$avg_return[3],
    vol=sqrt(cme^2*monthly_stat$var_return[1]
      +ice^2*monthly_stat$var_return[2]
      +ndaq^2*monthly_stat$var_return[3]
      +2*cme*ice*variance_matrix[2]
      +2*cme*ndaq*variance_matrix[3]
```

```

+2*ndaq*ice*variance_matrix[7])) %>%
mutate(sharpe=(return-monthly_stat$avg_rf[1])/vol)

plot_portfolio <- ggplot(data=portfolio,mapping=aes(x=vol,y=return))+  

  geom_point(data=portfolio %>% filter(ndaq==0),size=0.5,color="red")+
  geom_point(data=portfolio %>% filter(ice==0),size=0.5,color="blue")+
  geom_point(data=portfolio %>% filter(cme==0),size=0.5,color="green")+
  scale_x_continuous(limits=c(0.08,0.22))+
  scale_y_continuous(limits=c(0.15,0.21))+  

  labs(title = "Invest Opportunity of three stocks",
       x="standard deviation",y="expected return") +
  annotate(geom="text",x=0.182,y=0.17,label="CME")+
  annotate(geom="text",x=0.202,y=0.205,label="Nasdaq")+
  annotate(geom="text",x=0.202,y=0.162,label="ICE")+
  theme_bw()

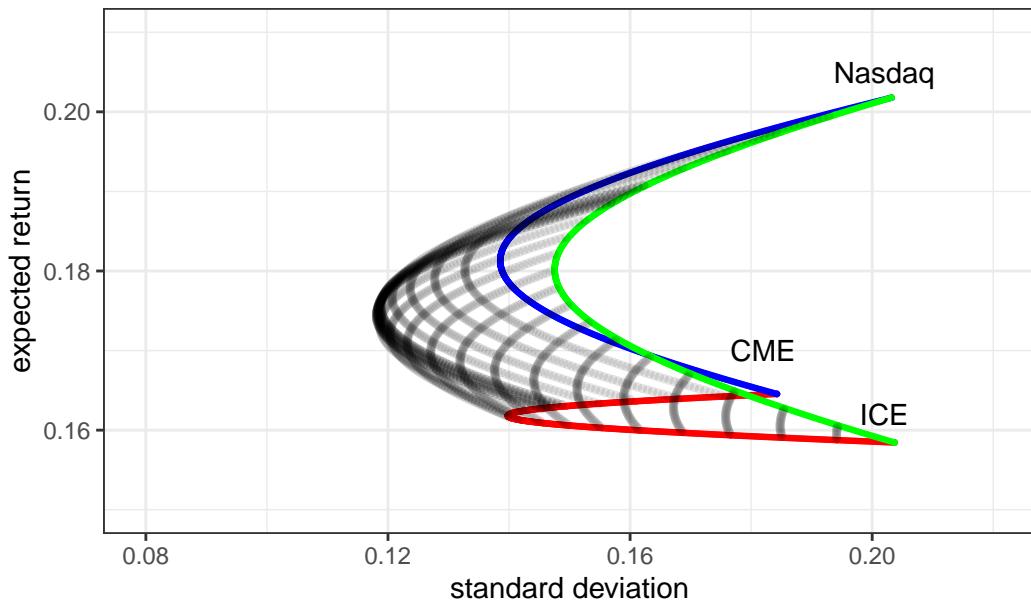
plot_combination <- plot_portfolio +  

  geom_point(data=portfolio %>% filter(cme!=0&ice!=0&ndaq!=0),size=1,
             color="black",alpha=0.1)

plot_combination

```

### Invest Opportunity of three stocks



(d)

Tangent p/f는 Sharpe r/o를 최대화시키는 세 주식의 조합으로, 위에서 도식화한 투자기회에 대하여 무위험이자율을 y절편으로 가지는 접선을 그려서 시각화할 수 있습니다.

접선의 기울기는 포트폴리오의 초과수익률을 변동성으로 나눈 값으로 Sharpe r/o가 되므로, 이 접선이 CAL에 해당합니다.

```

sharpe=max(portfolio$sharpe)
tangent <- tibble(vol=seq(0.08,0.17,0.01)) %>%
  mutate(return=monthly_stat$avg_rf[1]+sharpe*vol)

plot_tangent <- plot_combination+
  geom_line(data=tangent)+
  scale_x_continuous(limits=c(0.04,0.26))+ 
  scale_y_continuous(limits=c(0.11,0.25))+ 
  annotate(geom="text",x=0.125,y=0.23,label="CAL, y=1.3667x+0.0141")+
  annotate(geom="text",x=0.125,y=0.22,label="sharpe r/o=1.3667")+
  annotate(geom="text",x=0.1,y=0.18,label="Tangent p/f")

```

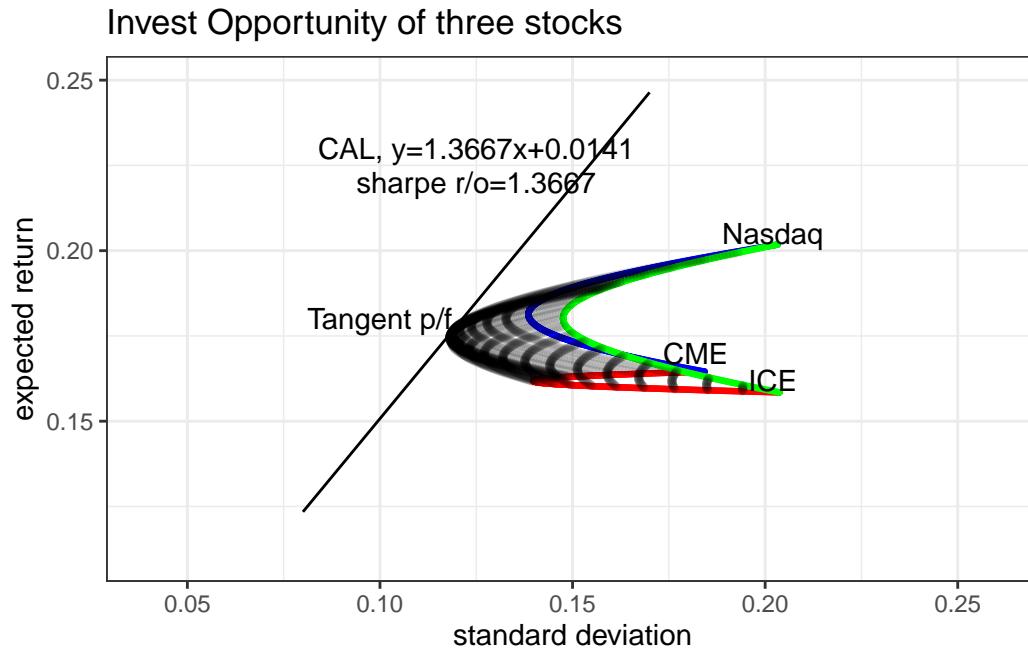
Scale for x is already present.

Adding another scale for x, which will replace the existing scale.

Scale for y is already present.

Adding another scale for y, which will replace the existing scale.

```
plot_tangent
```



○ 때, Tangent p/f의 기대수익률/변동성/구성비율은 아래와 같습니다.

```
portfolio %>% arrange(sharpe %>% desc()) %>% slice(1)
```

```
# A tibble: 1 x 6
  cme    ice   ndaq return    vol sharpe
  <dbl> <dbl> <dbl>   <dbl> <dbl>   <dbl>
1 0.36  0.27  0.37   0.177 0.119   1.37
```

(e)

Tangent p/f에서 CME, Nasdaq의 비중은 각각 36%, 37%로 이미 30%를 초과하였습니다. 따라서 30%를 초과보유할 주식을 선정할 때 두 주식을 선정한다면 최적포트폴리오는 (d)의 Tangent p/f와 동일할 것입니다.

한편, ICE를 선정한다면 (d)의 Tangent p/f 구성이 불가능합니다. 이 경우 새로운 efficient frontier는 아래와 같이 도식화할 수 있으며, 새로운 최적포트폴리오는 ICE의 비중이 정확히 30%일 때 결정됩니다.

```
# Set ice>=0.3
portfolio2 <- portfolio %>% filter(ice>=0.3)
sharpe2=max(portfolio2$sharpe)
tangent2 <- tibble(vol=seq(0.1,0.14,0.01)) %>%
  mutate(return=monthly_stat$avg_rf[1]+sharpe2*vol)

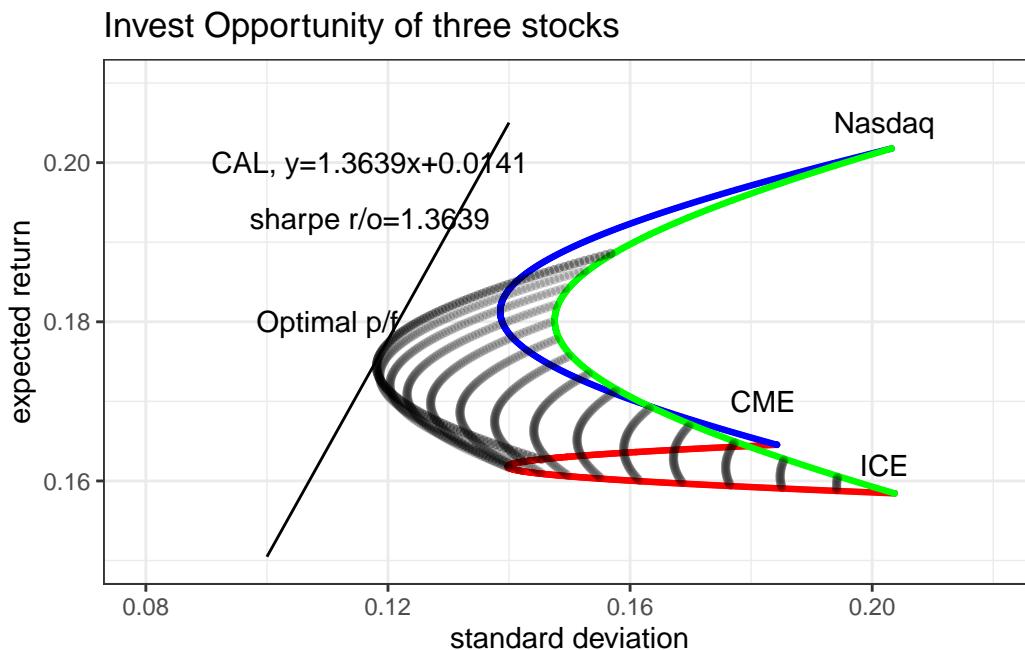
portfolio2 %>% arrange(desc(sharpe))
```

```
# A tibble: 1,905 x 6
  cme    ice   ndaq return    vol sharpe
  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
1 0.34    0.3  0.36  0.176 0.119  1.36
2 0.34    0.3  0.36  0.176 0.119  1.36
3 0.345   0.3  0.355 0.176 0.119  1.36
4 0.345   0.3  0.355 0.176 0.119  1.36
5 0.335   0.3  0.365 0.176 0.119  1.36
6 0.335   0.3  0.365 0.176 0.119  1.36
7 0.35    0.3  0.35  0.176 0.119  1.36
8 0.35    0.3  0.35  0.176 0.119  1.36
9 0.33    0.3  0.37  0.176 0.119  1.36
10 0.33   0.3  0.37  0.176 0.119  1.36
# i 1,895 more rows
```

```

plot_tangent2 <- plot_portfolio +
  geom_point(data=portfolio2 %>% filter(cme!=0&ice!=0&nDAQ!=0),
             size=1,color="black",alpha=0.2) +
  geom_line(data=tangent2) +
  annotate(geom="text",x=0.117,y=0.2,label="CAL, y=1.3639x+0.0141") +
  annotate(geom="text",x=0.117,y=0.193,label="sharpe r/o=1.3639") +
  annotate(geom="text",x=0.11,y=0.18,label="Optimal p/f")
plot_tangent2

```



Optimal p/f의 기대수익률/변동성/구성비율은 아래와 같습니다.

```

optimal <- portfolio2 %>% arrange(sharpe %>% desc()) %>% slice(1)
optimal

```

```

# A tibble: 1 x 6
  cme    ice   nDAQ return    vol sharpe
  <dbl> <dbl> <dbl>   <dbl> <dbl>   <dbl>
1  0.34    0.3   0.36   0.176 0.119    1.36

```

(f)

먼저, 포트폴리오의 연환산수익률이 정규분포를 따른다면 표준정규분포표를 참조하여 **Value at Risk**를 다음과 같이 산출할 수 있습니다.

$$5\% \text{ VaR} = E(r_p) - 1.65\sigma_p = 0.176 - 1.65 \times 0.119 = -1.99\%$$

```
VaR=optimal$return-1.65*optimal$vol; VaR
```

```
[1] -0.01990571
```

만약 수익률의 분포가 정규분포가 아니라면, Historical VaR 및 ES(Expected Shortfall)을 산출할 수 있습니다. 이를 데이터의 참조기간인 과거 10년간 최적포트폴리오의 월수익률이 필요합니다.

위의 최적포트폴리오는 **CME 34%, ICE 30%, Nasdaq 36%**로 구성된 포트폴리오이므로, 거래비용 등을 무시하고 매월말 포트폴리오의 구성비율을 조정한다고 가정하고 포트폴리오의 명목금액을  $P_t$ , 각 주식의  $(t, t+1)$  월수익률을  $r_{t,k}$ 라고 한다면  $(t, t+1)$ 간 포트폴리오의 월수익률  $r_t$ 는 다음과 같습니다.

$$P_{t+1} = 0.34 \times P_t \times (1 + r_{t,cme}) + 0.3 \times P_t \times (1 + r_{t,ice}) + 0.36 \times P_t \times (1 + r_{t,ndaq})$$

$$\Rightarrow 1 + r_t = \frac{P_{t+1}}{P_t} = 0.34 \times (1 + r_{t,cme}) + 0.3 \times (1 + r_{t,ice}) + 0.36 \times (1 + r_{t,ndaq})$$

이를 적용하면 과거 10년간 최적포트폴리오의 월수익률 120개를 얻을 수 있습니다.

이를 오름차순으로 정렬하면 **Historical 5% VaR**은 6번째 관측값이며, **5% ES**는 6개의 관측값의 평균으로 산출할 수 있습니다. 이는 월수익률 기반 VaR 및 ES이므로  $\sqrt{12}$ 를 곱하여 연환산하도록 하겠습니다.

```

optimal_monthly <- tibble()
optimal_monthly <- monthly_raw %>%
  select(month, name, return) %>%
  pivot_wider(names_from = "name", values_from = "return") %>%
  mutate(pf_return=optimal$cme*(1+cme)+optimal$ice*(1+ice)+optimal$ndaq*(1+ndaq)-1) %>%
  select(month, cme, ice, ndaq, pf_return) %>%
  arrange(pf_return) %>%
  slice(1:6)
optimal_monthly

```

```

# A tibble: 6 x 5
  month      cme      ice     ndaq pf_return
  <chr>    <dbl>    <dbl>    <dbl>    <dbl>
1 202204 -0.0779 -0.123   -0.117   -0.106
2 202002 -0.0842 -0.105   -0.119   -0.103
3 202003 -0.127  -0.0917  -0.0697  -0.0957
4 202209 -0.0899 -0.101   -0.0448  -0.0769
5 202201  0.00455 -0.0739  -0.147   -0.0734
6 202205 -0.0935 -0.116   -0.0134  -0.0714

```

Annualized 5% VaR은 -24.78%, ES는 -30.38%입니다.

정규분포 가정 하에 산출한 VaR과 큰 차이가 나는 이유는 지난 22년 Covid-19 팬데믹으로 인해 아래적으로 주가가 월 7% 이상 하락하는 급락장이 지속(Left fat tail)되었는데, 이때의 outlier 표본이 Historical VaR 산출을 지배한 반면 정규분포 근사시에는 반영되지 않아 차이가 발생하는 것으로 추정됩니다.

```

hist_VaR <- optimal_monthly$pf_return[6]*sqrt(12)
hist_ES <- mean(optimal_monthly$pf_return[1:6])*sqrt(12)
paste(round(hist_VaR,6), round(hist_ES,6) , round(VaR,6), sep=" / ")

```

```
[1] "-0.247279 / -0.303873 / -0.019906"
```

(g)

위의 최적포트폴리오와 무위험자산을 이용하여 투자를 결정한다면, 최종적인 포트폴리오의 기대수익률 및 변동성은 자본배분선( $E(r_p) = 1.3639\sigma_p + 0.0141$ ) 위에서 결정될 것입니다.

이때, 무위험자산의 비중은 투자자의 위험회피정도에 따른 효용함수  $U = E(r_p) - \frac{1}{2}A\sigma_p^2$ 를 최대화시키는 수준에서 결정됩니다.

이에 따라 산출한 무위험자산의 비중은 모든 경우(A=2, 3.5, 5)에서 0이 됩니다. 이는 과거기간 중 제로금리 기간이 다소 길어 무위험자산의 수익률이 낮은 반면, 선정한 주식의 기대수익률은 상대적으로 높아 일어난 것으로 추정됩니다.

부가적으로, 무위험자산의 비중 산식  $r^* = 1 - \frac{E(r_p) - r_f}{A\sigma_p^2}$ 를 역산하여 위험회피정도  $A$ 를 역산해보면, 약 11.48까지는 최적포트폴리오를 100% 보유하는 것이 효용을 극대화시키는 투자결정입니다.

```
# Utility function to verify risk-free asset ratio
utility <- tibble(vol=seq(optimal$vol,0,-0.0001)) %>%
  mutate(return=sharpe2*vol+monthly_stat$avg_rf[1]) %>%
  mutate(rf_ratio=(optimal$return-return)/(optimal$return-monthly_stat$avg_rf[1])) %>%
  mutate(a2{return-0.5*2*vol^2,
           a3.5{return-0.5*3.5*vol^2,
           a5{return-0.5*5*vol^2)
utility %>% arrange(desc(a5)) %>% slice(1)
```

```
# A tibble: 1 x 6
  vol return rf_ratio     a2    a3.5    a5
  <dbl>   <dbl>     <dbl> <dbl> <dbl> <dbl>
1 0.119   0.176      0  0.162  0.151  0.141
```

```
(optimal$return-monthly_stat$avg_rf[1])/optimal$vol^2
```

```
[1] 11.48027
```

## **Part IV**

### **재무회계('24봄)**

## 재무회계 Ch1-3 과제

### (1) Question 1-31

31. **Accrual versus cash basis of accounting.** The following information is based on the financial statements of Hewston, a large manufacturing firm. Annual revenues are \$66,387 million and net expenses (including income taxes) are \$62,313 million. During the year, the firm collected \$65,995 million in cash from customers and had cash outflows associated with payments to suppliers and vendors of \$56,411 million.
- a. Calculate net income and net cash flow for the year.
  - b. How can cash collected from customers be less than revenues?
  - c. How can cash payments to suppliers and vendors be less than expenses?

Figure 5.1: Chapter1-31

### Answer

$$(a) \text{net income} = 66387 - 62313 = 4,074\text{million\$}$$

$$\text{net cash flow} = 65995 - 56411 = 9,584\text{million\$}$$

(b) If customer buy a product of Hewston and want to pay later, then Hewston get a certain amount of revenue but get no cash. Instead, Hewston get a asset, namely accrued revenue, that will be paid for cash.

In that case, a company can get revenues more than collected cash.

(c) If Hewston buy material from suppliers and vendors to produce its own products but want to pay after selling product, then there exists a certain amount of expense and occurs a liability, namely accrued expense, that will pay for cash.

In that case, a company's expenses is bigger than its cash payments.

## (2) Question 1-34

- 34. Balance sheet relations.** Selected balance sheet amounts for SinoTwelve, a Chinese manufacturer, appear next. All amounts are in thousands of U.S. dollars (\$). Compute the missing amounts.

	2013	2012
Total Assets . . . . .	?	\$5,450,838
Current Liabilities . . . . .	\$4,488,461	3,527,504
Current Assets . . . . .	?	3,062,449
Total Liabilities and Shareholders' Equity . . . . .	7,199,847	?
Noncurrent Liabilities . . . . .	1,098,123	?
Shareholders' Equity . . . . .	?	1,134,276
Noncurrent Assets . . . . .	2,494,481	?
Total Liabilities. . . . .	?	?

Figure 5.2: Chapter1-34

### Answer

$$\text{Total Assets} = \text{Current Assets} + \text{Noncurrent Assets}$$

$$\text{Total Liabilities} = \text{Current Liabilities} + \text{Noncurrent Liabilities}$$

$$\text{Total Liabilities and Shareholders' Equity} = \text{Total Liabilities} + \text{Shareholders' Equity}$$

$$\text{And, Total Assets} = \text{Total Liabilities and Shareholders' Equity}$$

Apply above,

Category	2013	2012
Total Assets	<b>7,199,847</b>	5,450,838
Current Liabilities	4,488,461	3,527,504
Current Assets	<b>4,705,366</b>	3,062,449

Category	2013	2012
Total Liabilities and Shareholders' Equity	7,199,847	<b>5,450,838</b>
Noncurrent Liabilities	1,098,123	789,058
Shareholders' Equity	<b>1,613,263</b>	1,134,276
Noncurrent Assets	2,494,481	2,388,389
Total Liabilities	<b>5,586,594</b>	<b>4,316,562</b>

### (3) Question 2-10

#### PROBLEMS

10. **Dual effects of transactions on balance sheet equation and journal entries.** Assume that during Year 15, Bullseye Corporation, a U.S. retailer, engages in the following six transactions. Bullseye Corporation applies U.S. GAAP and reports its results in millions of U.S. dollars (\$). Do not be concerned that after these transactions, the balance in the Cash account is negative. The firm will deal with that issue in transactions not shown here.

- (1) The firm issues 20 million shares of \$0.0833 par value common stock for a total of \$960 million cash.
- (2) It purchases merchandise costing \$1,500 million on account.
- (3) The firm acquires a new store location, consisting of a building costing \$3,200 million and land costing \$930 million. It pays cash to the owner of the property.
- (4) The firm purchases fixtures for the new store costing \$860 million, on account.
- (5) The firm pays the merchandise supplier in transaction (2) the amount due.
- (6) The firm pays the supplier of the fixtures in transaction (4) half of the amount due in cash. The firm pays the other half by issuing 8.6 million common shares to the

supplier. At the time of this transaction, Bullseye Corporation's shares traded at \$50 per share in the market.

- a. Indicate the effects of these six transactions on the balance sheet equation using this format:

Transaction Number	Assets	=	Liabilities	+	Shareholders' Equity
(1)	<u>+\$960</u>		\$0		<u>+\$960</u>
Subtotal	\$960	=	\$0	+	\$960

- b. Give the journal entries for each of the six transactions.

Figure 5.3: Chapter2-10

#### Answer

(a)

Transaction Number	Assets	=	Liabilities	+	Shareholder's Equity
(1)	+\$960m		\$0		+\$960m

Transaction Number	Assets =	Liabilities +	Shareholder's Equity
(2)	+\$1,500m	+\$1,500m	\$0
(3)	+\$3,200m +\$930m -\$4,130m	\$0	\$0
(4)	+\$860m	+\$860m	\$0
(5)	-\$1,500m	-\$1,500m	\$0
(6)	-\$430m +\$430m -\$430m	-\$430m -\$430m	+\$430m
<b>Total</b>	<b>\$1,390m</b>	<b>\$0 +</b>	<b>\$1,390m</b>

(b)

Transaction Num. (1)	Debit	Credit
Cash	\$960m	
Common stock		\$1.7m
Additional capital		\$958.3m

Transaction Num. (2)	Debit	Credit
Inventory	\$1,500m	
Payable		\$1,500m

Transaction Num. (3)	Debit	Credit
Building	\$3,200m	
Landscape		\$930m
Cash paid		\$4,130m

Transaction Num. (4)	Debit	Credit
Fixture	\$860m	
Payable		\$860m

Transaction Num. (5)	Debit	Credit
Payable	\$1,500m	
Cash		\$1,500m

Transaction Num. (6)	Debit	Credit
Payable	\$430m	
Cash paid		\$430m
Cash	\$430m	
Common stock		\$0.7m
Additional capital		\$429.3m
Payable	\$430m	
Cash paid		\$430m

#### (4) Question 3-12

**12. Relations between financial statements.** The following selected information is based on the Year 7 financial statements adapted from those of Beyond Petroleum (BP). BP applies IFRS and reports its results in millions of U.S. dollars. Compute the missing information in each of the following four independent cases. The letters in parentheses refer to the following:

BS—Balance sheet

IS—Income statement

SCF—Statement of cash flows

	\$	?
a. Accounts Receivable, January 1, Year 7 (BS) . . . . .	\$	?
Sales on Account for Year 7 (IS) . . . . .	288,951	
Collections from Customers on Account during Year 7 (SCF) . . . . .	289,623	
Accounts Receivable, December 31, Year 7 (BS) . . . . .	38,020	
b. Income Taxes Payable, January 1, Year 7 (BS) . . . . .	\$	2,635
Income Tax Expense for Year 7 (IS) . . . . .	10,442	
Payments to Governments during Year 7 (SCF) . . . . .	?	
Income Taxes Payable, December 31, Year 7 (BS) . . . . .	3,282	
c. Trade Payables, January 1, Year 7 (BS) . . . . .	\$	42,236
Purchases of Supplies during Year 7 (SCF) . . . . .	15,162	
Payments to Suppliers during Year 7 (SCF) . . . . .	?	
Trade Payables, December 31, Year 7 (BS) . . . . .	43,152	
d. Retained Earnings, January 1, Year 7 (BS) . . . . .	\$	88,453
Net Income for Year 7 (IS) . . . . .	21,169	
Dividends Declared and Paid during Year 7 (SCF) . . . . .	8,106	
Retained Earnings, December 31, Year 7 (BS) . . . . .	?	

Figure 5.4: Chapter3-10

#### Answer

(a)

$$\text{Accounts Receivable in Jan.} + 288,951 - 289,623 = 38,020$$

*Accounts Receivable in Jan.* = \$38,692

(b)

$$2,635 + 10,442 - \text{Payments to Gov.} = 3,282$$

$$\text{Payments to Gov.} = \$9,795$$

(c)

$$42,236 + 15,162 - \text{Payments to suppliers} = 43,152$$

$$\text{Payments to suppliers} = \$14,246$$

(d)

$$88,453 + 21,169 - 8,106 = \text{Retained Earnings in Dec.}$$

$$\text{Retained Earnings in Dec.} = \$101,516$$

## 재무회계 Ch4 과제

### (1) Question 4-9

9. For each of the following items, indicate whether the item meets all of the criteria in the definition of a liability. If so, how does the firm value it?
  - a. Interest accrued but not paid on a note.
  - b. Advances from customers for goods and services to be delivered later.
  - c. Confirmed orders from customers for goods and services to be delivered later.
  - d. Product warranties.
  - e. Damages the company must pay if it loses a pending lawsuit.
  - f. Contractual promises to purchase specific quantities of natural gas for each of the next 10 years.
  - g. Promises by an airline to provide flights in the future in exchange for miles flown, if customers accumulate a certain number of miles at regular fares.

Figure 5.5: Chapter4-9

First, the criteria in the definition of a liability is following :

- (1) Probably, future sacrifices of economic benefits.
- (2) Arising from present obligation to transfer assets or provide services.
- (3) As a result of a past transaction.

In this perspective,

- (a) **is a liability.** accrued interest payable.  
(b) **is a liability.** (Advances from customers) It is equal to amount of received cash.

**(c) is not a liability.** criteria (3) is not satisfied.

**(d) is a liability.** (Provisions) It can be valued by some ratio of original product's price.

**(e) can be liability or not.** It depends on a probability of lawsuit.

If a probability of loss lawsuit is more than certain level,(In GAAP 80%, IFRS 50%) then it will be a liability. In that case, it can be valued by expected loss that should pay.

**(f) is not a liability.** criteria (3) is not satisfied.

**(g) is a liability.** It can be valued by some ratio of flight ticket's regular price.

## (2) Question 4-15

- 15. Balance sheet relations.** Jennings Group, a Malaysian investment management company, reported the following data for four recent years. Jennings applies Malaysian accounting standards and reports its results in millions of ringgit (RM). Compute the missing balance sheet amounts for each of the four years. In answering this question, assume that Jennings uses U.S. GAAP. (Adapted from the financial statements of Genting Group.)

	Year 7	Year 6	Year 5	Year 4
Noncurrent Assets.....	?	RM18,717.4	RM11,289.1	RM9,713.9
Shareholders' Equity .....	RM21,537.3	16,666.9	9,002.0	?
Total Assets .....	?	28,224.7	?	?
Current Liabilities .....	?	4,351.3	1,494.2	1,755.2
Current Assets .....	10,999.2	?	?	6,882.6
Noncurrent Liabilities .....	5,721.7	?	?	3,540.7
Total Liabilities and Shareholders' Equity.....	30,178.9	?	18,491.3	?

Figure 5.6: Chapter4-15

$$\text{Total Assets} = \text{Current Assets} + \text{Noncurrent Assets}$$

$$\text{Total Liabilities} = \text{Current Liabilities} + \text{Noncurrent Liabilities}$$

$$\text{Total Liabilities and Shareholders' Equity} = \text{Total Liabilities} + \text{Shareholders' Equity}$$

And, *Total Assets = Total Liabilities and Shareholders' Equity*

Apply above,

Category	Year7	Year6	Year5	Year4
Noncurrent Assets	<b>19,179.7</b>	18,717.4	11,289.1	9,713.9
Shareholders' Equity	21,537.3	16,666.9	9,002.0	<b>11,300.6</b>
Total Assets	<b>30,178.9</b>	28,224.7	<b>18,491.3</b>	<b>16,596.5</b>
Current Liabilities	<b>2,919.9</b>	4,351.3	1,494.2	1,755.2
Current Assets	10,999.2	<b>9,507.3</b>	<b>7,202.2</b>	6,882.6
Noncurrent Liabilities	5,721.7	<b>7,206.5</b>	<b>7,995.1</b>	3,540.7
Total Liabilities and Equity	30,178.9	<b>28,224.7</b>	18,491.3	<b>16,596.5</b>

## 재무회계 Ch5 과제

### (1) Question 5-12

12. **Revenue recognition.** Fonterra Cooperative Group Limited (Fonterra), a New Zealand dairy cooperative, uses the accrual basis of accounting and recognizes revenue at the time it sells products or renders services. Fonterra applies New Zealand accounting standards and reports its results in millions of New Zealand dollars (NZ\$). In answering this problem, assume Fonterra uses IFRS. Indicate which of the following transactions or events immediately give rise to Fonterra's recognition of revenue.
- a. Fonterra has completed the pasteurization of an order of 13,000 liters of milk it will deliver to a grocery store chain next week. Fonterra has not yet delivered the milk or invoiced the grocery store. The selling price of the milk is NZ\$26,000.
  - b. Refer to part a, and assume that the grocery store paid Fonterra a deposit of NZ\$5,000 on the order of the milk.
  - c. Fonterra delivered the milk and billed the grocery store. The grocery store has not yet paid the invoice.
  - d. One day after delivery, the grocery store called Fonterra and reported that it has to destroy 3,000 liters of milk because it had spoiled sometime prior to its delivery. The grocer refuses to pay for these 3,000 liters.
  - e. Fonterra spent NZ\$10 million to develop a technique to transform a by-product of casein (a protein found in milk and cheese) into ethanol. Fonterra expects to use this technique to generate sales of at least NZ\$2 million over the next year.
  - f. Refer to part e, and assume that Fonterra signed contracts worth NZ\$400 million for ethanol sales.

Figure 5.7: Chapter5-12

- (a) NOT recognized. It is before a completion of the earning process.
- (b) NOT recognized. It is before a completion of the earning process.
- (c) **Revenue recognized.** There is a completion of the earning process and receipt.(account receivable)

- (d) NOT recognized. It can be an expense.
- (e) NOT recognized. It is an investment, which is an expense.
- (f) NOT recognized. There is no completion of the earning process.

## (2) Question 5-18

**18. Income statement relations.** Selected income statement information for Years 11, 12, and 13 for SwissTek, a Swiss engineering firm. SwissTek applies U.S. GAAP and reports its results in millions of U.S. dollars.

	Year 13	Year 12	Year 11
Sales of Products .....	\$24,816	?	\$17,622
Income Tax Expense .....	595	?	464
Earnings Before Interest and Taxes .....	4,023	2,557	?
Sales of Services.....	4,367	3,778	3,342
Selling and Administrative Expenses .....	4,975	?	3,780
Cost of Services Sold .....	?	2,570	2,305
Income Before Taxes .....	?	2,076	1,199
Other Operating Income (Expense). ....	?	139	37
Interest and Other Financial Expense .....	286	?	407
Gross Profit .....	8,968	6,744	?
Cost of Products Sold.....	17,292	13,967	13,205
Other Non-operating Income (Expense) .....	?	(321)	(258)
Interest and Dividend Income.....	273	147	?
Net Income .....	3,757	1,390	?

Compute the missing amounts for each of the three years.

Figure 5.8: Chapter5-18

The following table is the re-organized income statement.

Category		Year 13	Year 12	Year 11
(+) Sales of Products		24,816	<b>19,503</b>	17,622
(-) Cost of Products Sold		17,292	13,967	13,205
(+) Sales of Service		4,367	3,778	3,342
(-) Cost of Services Sold		<b>2,923</b>	2,570	2,305
(=) <b>Gross Profit</b>		8,968	6,744	<b>5,454</b>
(-) Selling and Administrative Expenses		4,975	<b>4,326</b>	3,780
(+) Other Operating Income		(30)	139	37
(=) <b>Earning Before Interest and Taxes</b>		4,023	2,557	<b>1,711</b>
(+) Other Non-operating Income		<b>342</b>	(321)	(258)
(+) Interest and Dividend Income		273	147	<b>153</b>
(-) Interest and Other Financial Expense		286	<b>307</b>	407
(=) <b>Income Before Taxes</b>		<b>4,352</b>	2,076	1,199
(-) Income Tax Expense		595	686	464
(=) <b>Net Income</b>		3,757	1,390	<b>735</b>

## 재무회계 Ch6 과제

14. Derive sales revenue from data in the statement of cash flows and balance sheet. Microchem Corporation reported a balance of €5,196 million in accounts receivable at the beginning of the year and €5,334 million at the end of the year. Its statement of cash flows using the direct method reported cash collections from customers of €33,551 million for the year. Assuming that Microchem Corporation makes all sales on account, compute the amount of sales during the year.

### *Answer*

Because a company makes all sales on account, **total sales is sum of cash collections and account receivable**.

There are EUR33,551m cash collections and EUR138m increase in account receivable, so  $33551 + 138 = \text{EUR}33,689\text{m}$  will be a total sales of company during the year.

16. Derive cost of goods sold from data in the statement of cash flows. The section showing cash flow from operations, using the indirect method, for Taylor Stores reported an increase in inventories of \$5.7 million during the year. It reported also that the balance in accounts payable for inventories increased by \$5.9 million. The direct method would show cash payments for merchandise inventory purchased of \$646.9 million. Compute Taylor's cost of goods sold for the year.

### *Answer*

Cost will be **cash payments + increase payable - increase inventories**.

$646.9 + 5.9 - 5.7 = \text{USD}647.1\text{m}$  is cost of goods sold of company for the year.

- 26. Preparing a statement of cash flows from changes in balance sheet accounts.** The comparative balance sheets of Incloud Airlines show the following information for a recent year (amounts in thousands of US\$):

	Change	Amount	Direction
Cash . . . . .	\$ 40,308 <sup>a</sup>	40,308 <sup>a</sup>	Increase
Accounts Receivable . . . . .	15,351	15,351	Decrease
Inventories . . . . .	15,117	15,117	Increase
Prepayments . . . . .	16,776	16,776	Increase
Property, Plant, and Equipment (at Cost) . . . . .	1,134,644 <sup>b</sup>	1,134,644 <sup>b</sup>	Increase
Accumulated Depreciation . . . . .	264,088 <sup>b</sup>	264,088 <sup>b</sup>	Increase
Other Non-operating Assets . . . . .	8,711	8,711	Increase
Accounts Payable . . . . .	660	660	Decrease
Other Current Liabilities . . . . .	114,596	114,596	Increase
Long-Term Debt . . . . .	244,285	244,285	Increase
Other Non-operating Liabilities . . . . .	140,026	140,026	Increase
Common Stock . . . . .	96,991	96,991	Increase
Retained Earnings . . . . .	340,879 <sup>c</sup>	340,879 <sup>c</sup>	Increase

<sup>a</sup>Cash was \$378,511 at the beginning of the year and \$418,819 at the end of the year.

<sup>b</sup>Incloud Airlines did not sell any property, plant, and equipment during the year.

<sup>c</sup>Net income was \$474,378.

- a. Prepare a statement of cash flows for Incloud Airlines for the year. Treat changes in non-operating assets as investing transactions and changes in non-operating liabilities as financing transactions.

**Answer**

*Incloud Airlines*

*Statement of Cash Flow for the recent year(thousands\$)*

Category	Name	Amount
Cashflow from operations		<b>\$835,860</b>
+ Net income		474378
+ Increase in accumulated depreciation		264088
+ Decrease in account receivable		15351

Category	Name	Amount
-	Increase in other current liabilities	114596
-	Increase in inventories	15117
-	Increase in Prepayments	16776
-	Decrease in account payable	660
Cashflow from investing		<b>(\$1,143,355)</b>
-	Increase in property, plant, equipment	1134644
-	Increasing in other non-operating assets	8711
Cashflow from financing		<b>\$347,803</b>
+	Increasing in long-term debt	244285
+	Increasing in common stock	96991
+	Increasing in non-operating liabilities	140026
-	Payment of dividends	133499
<b>Net Change in Cash</b>		<b>\$40,308</b>
Cash, beginning of year		<b>\$378,511</b>
Cash, end of year		<b>\$418,819</b>

**29. Calculating and interpreting cash flows.** Market Star is a marketing services firm that creates advertising copy for clients and places the advertising in television, magazines, and other media. Accounts receivable represent amounts owed by clients, and accounts payable represent amounts payable to various media. Market Star has purchased other marketing services firms in recent years. Selected data for Market Star for three recent years appear next (amounts in millions of US\$):

	2013	2012	2011
Net Income . . . . .	\$ 499	\$ 363	\$279
Depreciation and Amortization Expense . . . . .	226	196	164
Increase (Decrease) in Accounts Receivable . . . . .	514	648	238
Increase (Decrease) in Inventories. . . . .	98	13	35
Increase (Decrease) in Prepayments. . . . .	125	(10)	64
Increase (Decrease) in Accounts Payable . . . . .	277	786	330
Increase (Decrease) in Other Current Liabilities . . . . .	420	278	70
Acquisition of Property, Plant, and Equipment. . . . .	150	130	115
Acquisition of Investments in Securities (noncurrent). . . . .	885	643	469
Dividends Paid . . . . .	122	104	88
Long-Term Debt Issued . . . . .	599	83	208
Common Stock Issued (Reacquired) . . . . .	(187)	(252)	42

- a. Prepare a comparative statement of cash flows for Market Star for the three years. Use the indirect method of computing cash flow from operations.

*Answer*

*Market Star*

*Statement of Cash Flow for 2011 ~ 2013(millions\$)*

Category	Name	2013	2012	2011
Cashflow from operations		<b>\$685m</b>	<b>\$972m</b>	<b>\$506m</b>
+ Net income		499	363	279
+ Increase in depreciation and amortization		226	196	279
+ Increase in account payable		277	786	330
- Increase in account receivable		514	648	238
- Increase in other current liabilities		420	278	70

Category	Name	2013	2012	2011
-	Increase in inventories	98	13	35
-	Increase in Prepayments	125	(10)	64
Cashflow from investing		<b>(\$1,035)</b>	<b>(\$773)</b>	<b>(\$584)</b>
-	Increase in property, plant, equipment	150	130	115
-	Increase in investments in securities	885	643	469
Cashflow from financing		<b>\$290</b>	<b>(\$273)</b>	<b>\$162</b>
+	Increasing in long-term debt	599	83	208
+	Increasing in common stock	(187)	(252)	42
-	Payment of dividends	122	104	88
<b>Net Change in Cash</b>		<b>(\$60)</b>	<b>(\$74)</b>	<b>\$84</b>

## **Part V**

### **경영통계분석('24봄)**

# 경영통계분석 과제1

## 1. Type of variables

### *Question*

Identify whether the following variables are numerical or categorical. If numerical, state whether the variable is discrete or continuous. If categorical, state whether the variable is nominal or ordinal.

a. Number of companies going bankrupt in a year.

: Numerical and Discrete

b. Petal area of rose flowers.

: Numerical and Continuous

c. Key on the musical scale.

: Categorical and Ordinal

d. Heart beats per minute of a Tour de France cyclist, averaged over the duration of the race.

: Numerical and Continuous

e. Stage of fruit ripeness.

: Categorical and Ordinal

f. Angle of flower orientation relative to position of the sun.

: Numerical and Continuous

**g. Tree species**

: Categorical and Nominal

**h. year of birth**

: Numerical and Discrete

**i. Gender**

: Categorical and Nominal

**j. Birth weight**

: Numerical and Continuous

## 2. Discrete data

### *Question*

Birds of the Caribbean islands of the Lesser Antilles are descended from immigrants originating from larger islands and the nearby mainland. The data presented here are the approximate dates of immigration, in millions of years, of each of 37 bird species now present on the Lesser Antilles. The dates were calculated from the difference in mitochondrial DNA sequences between each of the species and its closest living relative on larger islands or the mainland.

```
library(tidyverse)
birds_caribbean <- tibble("dates"=c(0.00,0.00,0.04,0.21,0.29,
0.54,0.63,0.88,0.96,1.25,
1.67,1.75,1.84,1.96,2.01,
2.51,2.72,3.30,3.51,4.05,
4.85,6.94,8.73,10.57,11.11,
12.45,14.00,17.30,17.92,18.05,
```

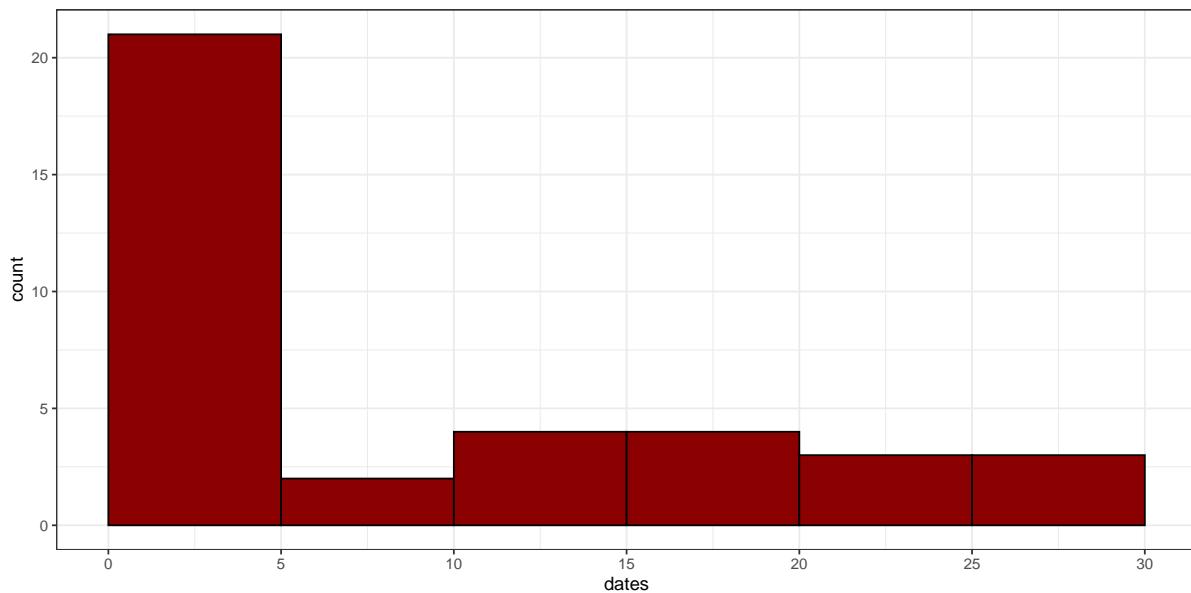
```
18.43, 22.48, 22.48, 23.48, 26.32,  
26.45, 28.87))
```

```
birds_caribbean$dates
```

```
[1] 0.00 0.00 0.04 0.21 0.29 0.54 0.63 0.88 0.96 1.25 1.67 1.75  
[13] 1.84 1.96 2.01 2.51 2.72 3.30 3.51 4.05 4.85 6.94 8.73 10.57  
[25] 11.11 12.45 14.00 17.30 17.92 18.05 18.43 22.48 22.48 23.48 26.32 26.45  
[37] 28.87
```

- a. Plot the data in a histogram and describe the shape of the frequency distribution.

```
ggplot(data=birds_caribbean, aes(x=dates))+  
  geom_histogram(binwidth=5, boundary=0, color="black", fill="darkred") +  
  scale_x_continuous(breaks = seq(0,30,5)) +  
  theme_bw()
```



히스토그램은 위와 같으며, 0~5구간에 전체 37종의 조류 중 20종 이상이 넘는 빈도가 집중되어 있습니다. 이는 과반 이상의 조류가 비교적 최근인 500만년 이내에 Lesser Antilles섬으로 이주해왔다는 것을 의미합니다. 나머지 약 15종의 조류는 각각 2~4종씩 ~1천만년, ~1500만년, …, ~3천만년 구간에 고르게 분포되어 있습니다.

**b. By viewing the graph alone, approximate the mean and median of the distribution. Which should be greater? Explain your reasoning.**

: 히스토그램은 원쪽으로 치우쳐져 있으며, 이는 *Skewness >> 0*이라는 것을 말합니다. 이러한 경우 평균은 중간값보다 크게 형성되고, 오른쪽 꼬리가 다소 긴 점을 감안하여 추정해보면, 중간값은 약 4, 평균은 약 10일 것으로 보입니다.

**c. Calculate the mean and median. Was your intuition in part (b) correct?**

```
birds_caribbean %>%  
  summarise(mean=mean(dates),  
            mid=median(dates))
```

```
# A tibble: 1 x 2  
  mean    mid  
  <dbl> <dbl>  
1 8.66  3.51
```

: 대략적인 경향성은 맞습니다. 실제 중간값은 3.51 및 평균은 8.66입니다.

**d. Calculate the first and third quantiles and the inter quartile range.**

```
birds_caribbean %>%  
  reframe(quantile=quantile(dates))
```

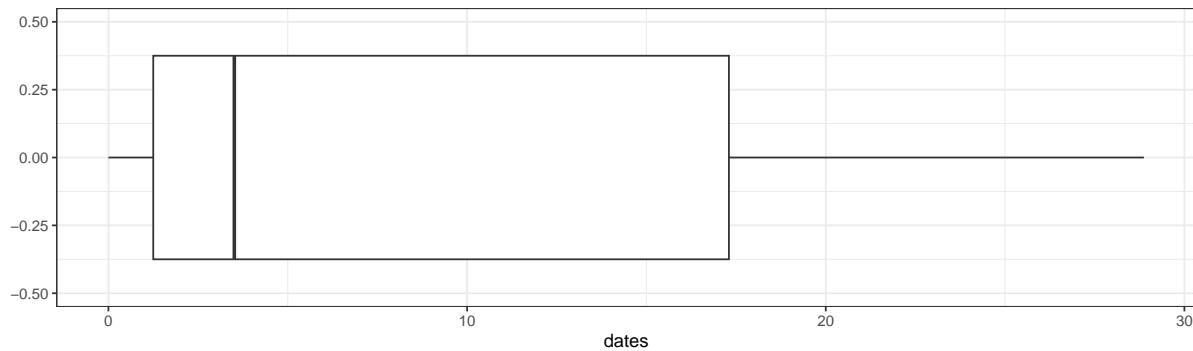
```
# A tibble: 5 x 1  
  quantile  
  <dbl>  
1 0
```

```
2      1.25
3      3.51
4     17.3
5     28.9
```

: First(25%) quantiles은 1.25이며, third(75%) quantiles은 17.30입니다. IQR은 16.05입니다.

e. Draw a box plot for these data

```
ggplot(birds_caribbean, aes(x=dates))+
  geom_boxplot()+
  scale_y_continuous(limits=c(-0.5,0.5))+  
  theme_bw()
```



### 3. Histogram

#### *Question*

Francis Galton presented the following data on the flight speeds of 3207 “old” homing pigeons traveling at least 90 miles.

a. What type of graph is this?

: Histogram

- b. Examine the graph and visually determine the approximate value of the mean (to the nearest 100 yards per minute). Explain how you obtained your estimate.**

: 그래프는 bell-shape의 히스토그램이며, 오른쪽 꼬리가 긴 형태입니다. 아마  $skewness > 0$ 일 것으로 보이며, 평균은 중간값보다 클 것 입니다.

그래프에서 중간값은 오른쪽 꼬리가 긴 분포를 고려할 때 약 1,000일 것으로 추정할 수 있는데, 평균값은 이보다 큰 약 1,100이 될 것으로 추정됩니다.

- c. Examine the graph and visually determine the approximate value of the median (to the nearest 100 yards per minute). Explain how you obtain your estimate.**

: 그래프에서 900~1000이 중심막대인데, bell-shape와 오른쪽 꼬리가 긴 분포를 고려할 때, 중간값은 약 1,000이 될 것으로 추정됩니다.

- d. Examine the graph and visually determine the approximate value of the mode (to the nearest 100 yards per minute). Explain how you obtained your estimate.**

: 최빈값은 가장 많은 빈도를 가지는 값 또는 구간으로, 히스토그램의 가장 높은 막대인 1000~1100에 존재할 것입니다. 막대 왼쪽에 빈도가 더 많은 점을 고려할 때 최빈값은 약 1,000일 것으로 보입니다.

- e. Examine the graph and visually determine the approximate value of the standard deviation (to the nearest 100 yards per minute). Explain how you obtained your estimate.**

: 먼저, 정규분포를 따르는 확률변수에 대해 다음 식이 성립합니다.

$$\text{for } X \sim \text{Normal Dist. then } P[-\sigma < X < \sigma] \approx 68.2\%$$

문제의 히스토그램은 bell-shape로 정규분포를 따른다고 가정하는 것에 큰 문제는 없어보입니다. 히스토그램에서 어림잡아봤을 때, 약  $1,100 \pm 200$  구간이 70~80%에 해당한다고 보여지는데, ( $P[900 < x < 1300] = 70 \sim 80\%$ )

이는 (b)의 추정 평균을 이용하면  $\pm 200$ 입니다. 즉, 히스토그램이 정규분포를 따른다고 가정하면, 히스토그램의 형태는  $N(1050, \sigma^2 = (200 - \alpha)^2)$ 의 확률밀도함수가 유사할 것입니다. 한편, 히스토그램은 오른쪽 꼬리가 긴 형태를 가지고 있어 변동성은 정규분포 대비 다소 클 것입니다.

따라서, 최종적으로 히스토그램 데이터의 표준편차는 약 200으로 추정됩니다.

## 4. Sample Statistics

```
handheight <- tibble("person"=c("A", "B", "C", "D", "E"),
                      "hand"=c(17, 15, 19, 17, 21),
                      "height"=c(150, 154, 169, 172, 175))
```

```
handheight
```

```
# A tibble: 5 x 3
  person   hand height
  <chr>   <dbl>  <dbl>
1 A         17     150
2 B         15     154
3 C         19     169
4 D         17     172
5 E         21     175
```

- Calculate the sample variances for hand and height, respectively.

표분 분산은  $s_x^2 = \frac{\sum_{k=1}^n (X_k - \bar{X})^2}{n-1}$ 입니다.

```
sample_var <- handheight %>%
  mutate(hand_tmp=hand-mean(handheight$hand),
        height_tmp=height-mean(handheight$height)) %>%
  mutate(hand_tmp=hand_tmp^2,
        height_tmp=height_tmp^2) %>%
  summarise(s_var_hand=sum(hand_tmp)/4,
            s_var_height=sum(height_tmp)/4)
sample_var
```

```
# A tibble: 1 x 2
  s_var_hand s_var_height
  <dbl>        <dbl>
1      5.2       126.
```

**b. Calculate the sample covariance.**

표본 공분산은  $s_{xy} = \frac{\sum_{k=1}^n (X_k - \bar{X})(Y_k - \bar{Y})}{n-1}$  입니다.

```
sample_cov <- handheight %>%
  mutate(s_cov_tmp=(hand-mean(handheight$hand))*(height-mean(handheight$height))) %>%
  summarise(s_cov=sum(s_cov_tmp)/4)
sample_cov
```

```
# A tibble: 1 x 1
  s_cov
  <dbl>
1 18.5
```

**c. Calculate the sample correlation and interpret the result.**

표본 상관계수는  $r_{xy} = \frac{s_{x,y}}{s_x s_y}$  입니다.

```
sample_corr=sample_cov$s_cov/sqrt(sample_var$s_var_hand*sample_var$s_var_height)
sample_corr
```

[1] 0.7213147

표본상관계수는 약 0.72이며, 이는 두 변수간에 강한 양의 선형관계가 존재하는 것을 말합니다. 즉, 손의 크기와 키 사이에는 양의 상관관계가 있어 손이 큰 집단은 키도 큰 경향이 있고, 키가 큰 집단은 손도 큰 경향이 있다는 것을 의미합니다.

### R 내장함수

참고로, R 내장함수는 표본연산을 기본으로 하고 있어 내장함수를 이용하여 표현 가능합니다.

```
var(handheight$hand)
```

```
[1] 5.2
```

```
var(handheight$height)
```

```
[1] 126.5
```

```
cov(handheight$hand, handheight$height)
```

```
[1] 18.5
```

```
cor(handheight$hand, handheight$height)
```

```
[1] 0.7213147
```

# 경영통계분석 과제2

## Question 1

At one large midwest university, about 40% of the college seniors have a social science major. Five seniors will be selected at random. Let  $X$  denote the number that **don't** have a social science major.

### (a) List the probability distribution

학생을 선정할 때, 사회과학 전공이 아닌 경우를 1로, 사회과학 전공인 경우를 0으로 하면 이는 확률 0.6의 독립된 베르누이 시행으로 볼 수 있습니다.

따라서, 이를 5번 반복하는 확률변수  $X$ 는  $X \sim B(5, 0.6)$ 를 따른다고 볼 수 있으며,  $P(X = k) = \binom{5}{k} 0.6^k 0.4^{(5-k)}$  이므로 이에 따른 확률분포는 아래와 같습니다.

k X	0	1	2	3	4	5
P(X=k)	0.0102	0.0768	0.230	0.346	0.259	0.0778

```
library(tidyverse)
```

```
binomial <- tibble(k=c(0,1,2,3,4,5), prob=choose(5,k)*0.6^k*0.4^(5-k))  
binomial
```

```
# A tibble: 6 x 2  
  k     prob  
  <dbl>  <dbl>
```

```

1      0 0.0102
2      1 0.0768
3      2 0.230
4      3 0.346
5      4 0.259
6      5 0.0778

```

(b) Calculate mean and variance from the entries in the list from part (a).

평균은 3, 분산은 1.2입니다.

```

mean=sum(binomial$k*binomial$prob)
variance=sum(binomial$k^2*binomial$prob)-mean^2
paste(mean,variance,sep=" / ")

```

```
[1] "3 / 1.2"
```

(c) Calculate  $E(X)=np$  and  $Var(X)=np(1-p)$  and compare your answer with part (b).

$np$ 와  $np(1-p)$ 의 값은 각각 3, 1.2로, 평균 및 분산과 같습니다.

## Question 2

If  $X$  has a normal distribution with  $\mu = 100$  and  $\sigma = 5$ , find  $b$  such that

(a)  $P(X < b) = 0.67$

$b = 102.2$

```

# Set parameter
mean <- 100; std <- 5

# (a) P(X<b)=0.67, Using bisection method
i <- mean-3*std;j <- mean+3*std;mid <- (i+j)/2
while(abs(pnorm(mid,mean,std)-0.67)>0.000001){
  if(pnorm(mid,mean,std)-0.67>=0){j <- mid}
  if(pnorm(mid,mean,std)-0.67<0){i <- mid}
  mid=(i+j)/2
}
paste(j, pnorm(j,mean,std), sep=" / ")

```

[1] "102.200012207031 / 0.670032330457297"

**(b)  $P(X>b)=0.011$**

b=111.451

```

# (b) P(X>b)=0.011, Using bisection method
i <- mean-3*std;j <- mean+3*std;mid <- (i+j)/2
while(abs(pnorm(mid,mean,std)-(1-0.011))>0.000001){
  if(pnorm(mid,mean,std)-(1-0.011)>=0){j <- mid}
  if(pnorm(mid,mean,std)-(1-0.011)<0){i <- mid}
  mid=(i+j)/2
}
paste(i, 1-pnorm(i,mean,std), sep=" / ")

```

[1] "111.451416015625 / 0.0110024524392477"

**(c)  $P(|X-100|<b)=0.966$**

b=10.601

```

# (c) P(|X-100|<b)=0.966, Using bisection method
i <- mean-3*std;j <- mean+3*std;mid <- (i+j)/2
while(abs(1-(1-pnorm(j,mean,std))*2-0.966)>0.000001){
  if(pnorm(mid,mean,std)-(1-(1-0.966)/2)>=0){j <- mid}
  if(pnorm(mid,mean,std)-(1-(1-0.966)/2)<0){i <- mid}
  mid=(i+j)/2
}
paste(j-100,1-(1-pnorm(j,mean,std))*2,sep=" / ")

```

[1] "10.6003761291504 / 0.966000298159686"

(d)  $P(X < 110) = b$

$b = 0.9772$

```

# (d) P(X<110)=b, Using bisection method
b_d <- pnorm(110,mean,std); b_d

```

[1] 0.9772499

(e)  $P(X > 95) = b$

$b = 0.8413$

```

# (e) P(X>95)=b
b_e <- 1-pnorm(95,mean,std); b_e

```

[1] 0.8413447

### Question 3

Suppose the amount of sun block lotion in plastic bottles leaving a filling machine has a normal distribution. The bottles are labeled 300 milliliter(ml) but the actual mean is 302 ml and the standard deviation is 2 ml.

- (a) What is the probability that an individual bottle will contain less than 299 ml?

로션의 양(ml)에 대한 확률변수  $X$ 는  $X \sim N(302, 2^2)$ 를 따르므로,  $P(X < 299) \approx 6.68\%$

```
pnorm(299, 302, 2)
```

[1] 0.0668072

- (b) If you pick up 5 bottles and check the amount of sun block lotion in each bottle, what is the probability that all of 5 bottles contain less than 299 ml? (Assume that they are independent.)

각각의 병이 확률변수  $X_i$ ,  $i = 1, 2, 3, 4, 5$ 라고 하면  $X_i \sim N(302, 2^2)$ 이며 각 확률변수는 독립이므로  $P(X_i \in k_1, X_j \in k_2) = P(X_i \in k_1)P(X_j \in k_2)$ 가 성립합니다.

따라서, 모든 병이 299ml 미만일 확률  $P(X_i < 299 \text{ for all } i) = \prod_{k=1}^5 P(X_i < 299) \approx 0.668^5$

```
pnorm(299, 302, 2)^5
```

[1] 1.330811e-06

### Question 4

The number of complaints per day,  $X$ , received by a cable TV distributor has the probability distribution

x	0	1	2	3
f(x)	0.4	0.3	0.1	0.2

- (a) Find the expected value and the standard deviation of the number of complaints per day.

평균은 1.1, 표준편차는  $\sqrt{1.29} \approx 1.136$ 입니다.

```
complaints <- tibble(x=c(0,1,2,3), prob=c(0.4,0.3,0.1,0.2))
mean <- sum(complaints$x*complaints$prob)
vol <- sum(complaints$x^2*complaints$prob)-mean^2
std <- sqrt(vol)
paste(mean,std,sep=" / ")
```

[1] "1.1 / 1.13578166916005"

- (b) What is the approximate probability that the distributor will receive more than 2 complaints in average during 100 days?

먼저,  $X_i$ 를 i일 뒤에 커플레인이 들어오는 횟수라고 한다면  $X_i$ 는 위 분포를 따르게 될 것입니다. 100일 뒤까지의 커플레인이 들어오는 횟수를  $X_1, X_2, \dots, X_{100}$ 이라고 한다면, 각각의  $X_i$ 는 독립이고 동일한 분포(iid)를 가지는 표본입니다.

한편, 중심극한정리(Central Limit Theorem)에 따라, 표본의 크기가 충분히 크다면 해당 표본들을 정규 분포로 근사시킬 수 있습니다.

$$\text{if } X_i \text{ are iid, } \frac{\sum_{k=1}^n (X_i - \mu)}{\sqrt{n} \times \sigma} \sim N(0, 1) \text{ for large } n$$

100번의 독립시행  $X_1 \sim X_{100}$ 은 정규근사하기 충분한 표본이므로 이를 적용하면 다음과 같습니다.

$$\text{By CLT, } \frac{\sum_{k=1}^{100} (X_i - 1.1)}{\sqrt{100} \times \sqrt{1.29}} = \frac{\sum_{k=1}^{100} X_i - 110}{\sqrt{129}} \sim N(0, 1)$$

이제, 100일 동안의 일평균 컴플레인 횟수가 2보다 클 확률은 다음과 같습니다.

$$P\left[\frac{\sum_{k=1}^{100} X_i}{100} > 2\right] = P\left[\frac{\sum_{k=1}^{100} X_i - 110}{\sqrt{129}} = Z > \frac{90}{\sqrt{129}}\right]$$

해당 확률을 R을 통해 구해보면, 0에 가깝습니다.

```
1-pnorm(90/sqrt(129), 0, 1)
```

[1] 1.110223e-15

(c) If we observe for five days, what is the probability that the distributor will receive exactly 1 complaint only for two days?

컴플레인을 1개만 받을 확률은 0.3입니다.

하루를 컴플레인을 1개만 받는 경우를 1, 1개가 아닌 경우를 0으로만 나눈다면, 해당 시행은 확률 0.3의 베르누이 시행입니다. 이를 5일 동안 관측한다면 각각은 독립이므로 해당 분포는  $B(5, 0.3)$ 인 이항분포입니다.

즉, 5일중 2일만 컴플레인을 정확히 1개 받을 확률은  $\binom{5}{2}0.3^20.7^3 = 30.87\%$

```
choose(5, 2)*0.3^2*0.7^3
```

[1] 0.3087

(d) If we observe for 100 days, what is the approximate distribution of the number of days that the distributor will not receive any complaint?

하루에 컴플레인이 없을 확률은 0.4로, 컴플레인이 없는 경우를 1, 있는 경우를 0으로 나눈다면 해당 시행은 확률 0.4의 베르누이 시행으로 볼 수 있습니다.

이를 100일간의 표본으로 나누어 시행하면 하루단위 시행은 각각 독립이므로, 이는 확률 0.4 및 시행횟수 100의 이항분포( $B(100, 0.4)$ )를 따르게 됩니다.

한편, 해당 이항분포는 동일한 확률을 가진 100개의 베르누이시행( $B_i \sim B(1, 0.4)$ )으로 나누어 볼 수 있고, (b)와 동일하게 CLT를 이용한다면 아래와 같이 표현할 수 있습니다.

$$\frac{\sum_{k=1}^{100} (B_i - 0.4)}{\sqrt{100 \times 0.4 \times 0.6}} \sim N(0, 1), \text{ Since } X = \sum_{k=1}^{100} B_i, \frac{X - 40}{\sqrt{24}} \sim N(0, 1) \Rightarrow X \sim N(40, 24)$$

즉, 평균이 40이고 분산이 24인 정규분포로 근사할 수 있습니다.

(e) Using the approximate distribution in (d), what is the approximate probability that the number of days that the distributor will not receive any complaint is at most 30 days?

해당 확률은  $P(X \leq 30)$  for  $X \sim B(100, 0.4)$ 과 같습니다.

위의 정규분포 근사를 활용하면  $X \sim N(40, 24)$ , then  $P(X \leq 30) \approx 2.06\%$

```
pnorm(30, 40, sqrt(24))
```

```
[1] 0.02061342
```

## Question 5

The probability that a voter will believe a rumor about a politician is 0.3.

(a) Find the probability that the first 3 voters don't believe the rumor but the 4th voter believe it.

$$0.7 \times 0.7 \times 0.7 \times 0.3 = 0.1029 = 10.29\%$$

(b) Find the probability that the exactly one person believe the rumor if 5 voters are told individually

각 유권자는 소문을 개별적으로 확인하였으므로 유권자들이 루머를 믿는 확률변수는 각각 독립입니다.

루머를 믿을 확률은 0.3이므로 루머를 믿는 경우를 1, 안믿는 경우를 0이라고 한다면 이는 확률 0.3의 베르누이 시행입니다. 이를 5번 반복하면 5명의 유권자 중 루머를 믿는 유권자의 수는  $B(5, 0.3)$ 를 따릅니다.

정확히 한사람만 루머를 믿을 확률은,  $\binom{5}{1} 0.3^1 0.7^4 \approx 36.02\%$

```
choose(5,1)*0.3*0.7^4
```

```
[1] 0.36015
```

## Question 6

Here is the assignment of probabilities that describes the age (in years) and the gender of a randomly selected American College student.

Category	14~18	18~25	25~35	35~
Male	0.01	0.28	0.13	0.04
Femail	0.02	0.3	0.14	0.08

A college student will be selected at random. Let A=[student is Female] and B=[student is at least 25 but less than 35 years old]. Find,

(a)  $P(A)$  and  $P(B)$

$$P(A) = 0.02 + 0.3 + 0.14 + 0.08 = 0.54$$

$$P(B) = 0.13 + 0.14 = 0.27$$

(b)  $P(A \text{ or } B)$

$$P(A \text{ or } B) = P(A \cup B) = P(A) + P(B) - P(A \cap B) = 0.54 + 0.27 - 0.14 = 0.67$$

## Question 7

Show the following statement: If  $X \sim t[k]$ , then  $X^2 \sim F[1, k]$

$$X \sim t[k] \Leftrightarrow X = \frac{Z}{\sqrt{V/k}} \text{ for } Z \sim N(0, 1), V \sim \chi^2(k)$$

$$\Leftrightarrow X^2 = \frac{Z^2}{V/k} = \frac{U/1}{V/k} \text{ for } U = Z^2 \sim \chi^2(1)$$

$$\Leftrightarrow F = X^2 = \frac{U/1}{V/k} \sim F(1, k) \text{ for } U \sim \chi^2(1), V \sim \chi^2(k) \quad \square$$

## 경영통계분석 과제3

### Question 1

if  $X_1, X_2, \dots, X_n$  are i.i.d random variables with mean  $\mu$  and variance  $\sigma^2$ , calculate the covariance  $\bar{X}$  and  $X_i - \bar{X}$  for any  $i = 1, \dots, n$ .

### Answer

먼저, 임의의 확률변수  $U, V$ 에 대하여  $Cov(U, V) = E[(U - E[U])(V - E[V])] = E[UV] - E[U]E[V]$ 입니다.

$\bar{X} = \frac{1}{n} \sum_{i=1}^n X_i \Rightarrow E[\bar{X}] = \mu, E[X_i - \bar{X}] = 0$  for all  $i$  이고, 각각의  $X_i$ 는 독립이므로  $E[X_i X_j] = E[X_i]E[X_j] = \mu^2$  for  $i \neq j$  이며,  $E[X_i^2] = Var[X_i] + E[X_i]^2 = \sigma^2 + \mu^2$ 입니다.

이를 이용하면  $E[\bar{X}^2] = E\left[\frac{\sum_i X_i^2 + \sum_{i,j}^{i \neq j} X_i X_j}{n^2}\right] = \frac{n(\sigma^2 + \mu^2) + n(n-1)\mu^2}{n^2} = \frac{n\sigma^2 + n^2\mu^2}{n^2} = \frac{\sigma^2}{n} + \mu^2$ 입니다.

이제 공분산을 구해보겠습니다.

$$Cov[\bar{X}, X_i - \bar{X}] = E[\bar{X}(X_i - \bar{X})] - \mu \times 0 = E[\bar{X}X_i] - E[\bar{X}^2]$$

$$= \frac{E[X_i(X_1 + \dots + X_n)]}{n} - \frac{\sigma^2}{n} - \mu^2 = \frac{E[X_i^2] + (n-1)\mu^2 - \sigma^2 - n\mu^2}{n}$$

$$= \frac{\sigma^2 + \mu^2 - \mu^2 - \sigma^2}{n} = 0$$

즉, 공분산은 0입니다.

### **i** 간결한 풀이방법

$$\text{Cov}[X_i, X_j] = 0 \text{ for } i \neq j, \text{Cov}[X_i, X_i] = \text{Var}[X_i] = \sigma^2, \text{Var}[\bar{X}] = \frac{\sigma^2}{n} \text{이므로,}$$

$$\text{Cov}[\bar{X}, X_i - \bar{X}] = \text{Cov}[\bar{X}, X_i] - \text{Cov}[\bar{X}, \bar{X}] = \frac{\text{Var}[X_i]}{n} - \text{Var}[\bar{X}] = 0$$

## Question 2

A commercial for a manufacturer of household appliance claims that 3% of all its products require a service call in the first year. A consumer protection association wants to check the claim by surveying 400 households that recently purchased one of the company's appliances. What is the probability that more than 5% require a service call within the first year?

- (a) Calculate the probability using the approximate normal approach.
- (b) Calculate the probability using the binomial distribution.

## Answer

각각의 가구를 조사하는 일은 독립이며, 클레임이 있는 경우를 1, 없는 경우를 0으로 나누면 이는  $Ber(0.03)$ 의 베르누이시행입니다.

이를 400번 반복하므로 분포는  $B(400, 0.03)$ 의 이항분포이며, 정규분포로 근사하면 아래와 같습니다.

$$\text{By CLT, } \frac{\sum_{i=1}^{400}(X_i - 0.03)}{\sqrt{400}\sqrt{0.03 \times 0.97}} \sim N(0, 1) \Rightarrow X \sim N(12, 11.64) \text{ for } X \sim B(400, 0.03)$$

클레임이 있는 가구가 5%를 초과할 확률은 20가구를 초과할 확률  $P(X > 20)$ 이므로, (a) 0.95% 및 (b) 1.05%입니다.

정규분포 근사는  $n$ 이 매우 크고  $p$ 가 0.5와 인접할 때 정확도가 높은데,  $n$ 은 충분히 크지만  $p$ 가 매우 작아 다소 오차가 발생하는 것으로 추정됩니다.

```
# (a) normal dist.  
1-pnorm(20,12,sqrt(11.64))
```

```
[1] 0.009517584
```

```
# (b) binomial dist.  
prob=0  
for(i in 0:20){  
    prob=prob+choose(400,i)*0.03^i*0.97^(400-i)  
}  
1-prob
```

```
[1] 0.01045374
```

### Question 3

Suppose that  $X$  has normal dist. with  $\mu = 10$ ,  $\sigma = 2$

(a)  $P(6 < X < 14) : 0.9545$

```
mean=10; std=2  
pnorm(14,mean,std)-pnorm(6,mean,std)
```

```
[1] 0.9544997
```

**i** Note

이는  $P(X \in (\mu - 2\sigma, \mu + 2\sigma))$  이므로, 95.45%임이 잘 알려져 있습니다.

(b)  $P(X \leq c) = 0.95 : c=13.29$

```

# (a) P(X<=c)=0.95, Using bisection method
i <- mean-3*std
j <- mean+3*std
mid <- (i+j)/2
while(abs(pnorm(mid,mean,std)-0.95)>0.000001){
  if(pnorm(mid,mean,std)-0.95>=0){j <- mid}
  if(pnorm(mid,mean,std)-0.95<0){i <- mid}
  mid=(i+j)/2
}
paste(j, pnorm(j,mean,std), sep=" / ")

```

[1] "13.2897644042969 / 0.950002947051978"

(c) for sample  $X_1, \dots, X_4$ ,  $P(\bar{X} \leq 12) : 0.9772$

$E(\bar{X}) = 10$ ,  $Var(\bar{X}) = \frac{4}{4} = 1$ 이고, 각각의  $X_i$ 는 i.i.d 정규분포이므로  $\bar{X} \sim N(10, 1)$ 입니다.

```
pnorm(12, 10, 1)
```

[1] 0.9772499

## Question 4

Bits are sent over a communications channel in packets of 160. If the probability of a bit being corrupted (one error) over this channel is 0.2 and such errors are independent. Let  $X$  denotes the number of bits that are corrupted over this channels

- (a) What is the distribution of  $X$ ? Can it be approximated as normal distribution?
- (b) Approximately, what is the probability that more than 50 bits in a packet are corrupted?

## Answer

한번의 전송에 에러가 발생하는 경우를 1, 아닌 경우를 0으로 나누면 이는 확률 0.2의 베르누이 시행입니다. 이를 160번 반복하고 에러가 발생하는 횟수인  $X$ 는  $B(160, 0.2)$  이항분포를 따를 것입니다.

해당 이항분포는 시행횟수가 충분히 크지만( $>30$ ), 확률이 0.2로 다소 낮아 정규분포  $N(32, 25.6)$ 를 사용할 때 근소한 오차가 발생할 수 있습니다.

$P(X > 30)$ 을 구해보면, 정규분포는 34.63%, 이항분포는 39.05%으로 오차가 다소 존재하므로 이에 유의해야합니다.

```
prob=0
for(i in 0:30){
  prob=prob+choose(160,i)*0.2^i*0.8^(160-i)
}
paste(pnorm(30,32,sqrt(25.6)),prob,sep=" / ")
```

```
[1] "0.34631639202098 / 0.390465138866041"
```

에러가 50번을 초과하여 발생할 확률은  $P(X > 50)$ 이므로, 정규분포를 이용한 확률은 0.0187%입니다. 한편, 이항분포를 이용한 확률은 0.0265%입니다.

```
prob=0
for(i in 0:50){
  prob=prob+choose(160,i)*0.2^i*0.8^(160-i)
}
paste(1-pnorm(50,32,sqrt(25.6)),1-prob,sep=" / ")
```

```
[1] "0.000187156123554355 / 0.000264520708086802"
```

## Question 5

A large population is described by the probability distribution. Let  $X_1$  and  $X_2$  be a random sample of size 2 from the distribution.

X	f(x)
0	0.1
1	0.2
2	0.7

- (a) Determine the sampling distribution of  $\max(X_1, X_2)$
- (b) Determine the sampling distribution of  $X_1 + X_2$

## Answer

먼저  $X_1, X_2$ 는 모집단으로부터 추출한 표본으로 i.i.d를 만족할 것 입니다. 따라서  $f(X_1 = x_1, X_2 = x_2) = f(x_1)f(x_2)$ 로 산출할 수 있습니다.

$x_m \in \{0, 1, 2\}$  for  $x_m \sim \max(X_1, X_2)$ 이고, 순서쌍  $(x_1, x_2)$ 로 경우의 수를 나타내면  $(0, 0) \Rightarrow x_m = 0$  및  $(0, 1), (1, 0), (1, 1) \Rightarrow x_m = 1$  및  $(2, 0), (2, 1), (2, 2), (0, 2), (1, 2) \Rightarrow x_m = 2$ 입니다. 따라서,  $\max(X_1, X_2)$ 의 분포는,

$x_m$	$\max(X_1, X_2)$
0	0.01
1	$0.02+0.02+0.04=0.08$
2	$0.7+0.21=0.91$

위와 유사한 방법으로  $X_1 + X_2$ 의 경우의 수를 나타내면 다음과 같습니다.

$x_1 \in X_1$	$x_2 \in X_2$	$\Rightarrow x_+ \in X_1 + X_2$
0	0	0

$x_1 \in X_1$	$x_2 \in X_2$	$\Rightarrow x_+ \in X_1 + X_2$
0	1	1
1	0	1
1	1	2
2	0	2
0	2	2
1	2	3
2	1	3
2	2	4

따라서,  $X_1 + X_2$ 의 분포는,

$x_+$	$X_1 + X_2$
0	0.01
1	0.02+0.02=0.04
2	0.04+0.07+0.07=0.18
3	0.14+0.14=0.28
4	0.49

## 경영통계분석 과제4

### Problem 1.

Assume that the standard deviation of the heights of five-year-old boys is 3.5 inches. How many five-year-old need to be sampled if we want to be 90% sure that the population mean height is estimated within .5 inch?

#### Answer

크기가 N인 표본을 임의추출하고, 충분히 크다고 가정하겠습니다.

모표준편차가 3.5(inch)이므로, 크기가 N인 표본평균  $\bar{X}$ 은 중심극한정리(CLT)에 의해 정규분포  $N(\mu, \frac{3.5^2}{N})$ 을 따르게 될 것입니다.

이를 이용하여 신뢰수준이 90%인 모평균의 신뢰구간을 양편으로 추정해보겠습니다.

$$P(z_{0.95} = -z_{0.05} < \frac{\bar{X} - \mu}{3.5/\sqrt{N}} < z_{0.05}) = 0.9 \Rightarrow P(\bar{X} - z_{0.05} \frac{3.5}{\sqrt{N}} < \mu < \bar{X} + z_{0.05} \frac{3.5}{\sqrt{N}}) = 0.9$$

여기서, 모평균을 0.5inch 이내로 추정하는 것의 의미는, 표본오차가 0.5이내, 즉  $z_{0.05} \frac{3.5}{\sqrt{N}} < 0.5$ 라는 뜻입니다.

$$z_{0.05} \approx 1.65 \text{이므로, 이를 정리하면 } 1.65 \times 7 < \sqrt{N} \Rightarrow 133.4025 < N$$

즉, 표본의 수가 134명 이상이라면 90% 신뢰구간을 0.5inch 오차 이내로 추정할 수 있습니다.

또한, 이는 충분히 큰 표본이므로 앞서 CLT를 이용하는 것에 위배되지 않습니다.

## Problem 2.

An employee of an on-campus copy center wants to determine the mean number of copies before a cartridge needs to be replaced. She records the life length in thousands of copies for 43 cartridges and obtains  $n = 43$ ,  $\bar{x} = 8.12$ ,  $s = 1.78$  thousand copies. Obtain a 90% confidence interval for the population mean,  $\mu$ , number of copies in thousands before a cartridge should be replaced.

### Answer

먼저, 43개의 샘플이 충분히 크다고 생각하고 시작하겠습니다.

CLT에 따라 표본평균  $\bar{X} \sim N(\mu, \sigma/\sqrt{43})$ 이며, 표본분산  $\frac{42 \times s^2}{\sigma^2} \sim \chi^2(42)$ 입니다.

이를 정리하면  $\frac{\bar{X} - \mu}{s/\sqrt{43}} \sim t(42)$ 가 되는데, 표본의 크기가 충분히 크므로  $t(n-1) \rightarrow N(0, 1)$ 로 근사할 수 있습니다.

이제, 이를 이용하여 모평균의 90% 신뢰구간을 추정해보겠습니다.

$$P(\mu \in (\bar{X} \pm z_{0.05} \frac{s}{\sqrt{43}})) = 0.90 \Rightarrow P(\mu \in (7.672, 8.568)) = 0.90$$

즉, 신뢰구간은 (7.672, 8.568)입니다.

## Problem 3.

Data on the average weekly earnings were obtained from a survey of 50 nonsupervisory production workers in the mining industry. The sample mean and standard deviation were found to be \$630 and \$35, respectively.

- Estimate the true mean weekly earnings and determine the 95% error margin.
- Construct a 95% confidence interval for the true mean weekly earnings

## Answer

먼저, 표본평균  $\bar{X}$ 의 성질에 따라  $E(\bar{X}) = \mu$ ,  $\bar{X} \rightarrow^p \mu$  이므로  $\bar{X}$ 는 모평균에 대한 불편(unbiased) 및 일치(consistent) 추정량입니다.

따라서 모평균의 estimator로 표본평균을 선정하고, 모평균을 \$630으로 추정하겠습니다.

한편, 표본의 수가 50으로 충분히 크므로 앞서 이용한 논리를 그대로 준용하여  $\frac{\bar{X}-\mu}{s/\sqrt{50}} \sim t(49) \approx N(0, 1)$ 라고 할 수 있습니다.

따라서 95%의 error margin은  $z_{0.025} \frac{s}{\sqrt{50}} = 1.96 \cdot \frac{s}{\sqrt{50}} = 8.167$ 입니다.

마지막으로, 95%의 신뢰수준으로 추정한 모평균의 신뢰구간은 아래와 같습니다.

$$P(\bar{X} - z_{0.025} \frac{s}{\sqrt{50}} < \mu < \bar{X} + z_{0.025} \frac{s}{\sqrt{50}}) \Rightarrow \text{Confidence interval is } (620.299, 639.702)$$