Introduction to Big Data

- Developed by Dr. Keungoui KIM
- https://awekim.github.io/portfolio/

Lecture 2. Pythong Programming - 1

PART 1

Module

```
# import whole package/module
import pandas
```

```
pandas.
```

```
# import whole package/module and name it differently
import pandas as pd
```

```
pd.
```

```
# import specific “variable, function, class” from the package/module
from pandas import DataFrame
```

```
DataFrame.
```

```
pd.DataFrame.
```

```
# import more than one “variables, functions, classes” from package/module, use comma.
from pandas import DataFrame, Series
```

```
from pandas import DataFrame
from pandas import Series
```

```
DataFrame.
Series.
```

```
# import specific “variable, function, class” from the package/module and name it differently
from pandas import DataFrame as DF
```

```
DF.
```

```
# import multiple “variables, functions, classes” from the package/module and name them differently
from pandas import DataFrame as DF, Series as SR
```

```
DF.
SR.
```

```
# import all “variables, functions, classes” from the package/module
from pandas import *
```

Variables

```
False
```

```
1
```

```

```

```
a = 1
```

```
a
```

```
a = 123
```

```
A
```

```
A
```

```
# invalid variable name
100_name = 100
```

```
# reserved words : False, Class, finally, is, return, None, continue, for, lambda, try, True, def, from, nonlocal, while,
False = 1
```

```
import = 1
```

```
a = 123
```

```
a = 123
b = 234

print(a,id(a))
print(b,id(b))
#a
#b
```

Review

1. Import pandas module but rename it to pd

2. Write down the Python command that import DataFrame and Series functions from pandas
3. Which of the following is valid variable name?

- 100_name, class, break, False, false

```
# 1.
import pandas as pd
```

```
# 2.
from pandas import DataFrame, Series
```

```
# 3.
100_name = 1
```

```
# 3.
class = 1
```

```
# 3.
break = 1
```

```
# 3.
False = 1
```

```
# 3. Answer
false = 1
```

Data Types

```
type(123)
```

```
type(123.45)
```

```
type("123")
```

```
type('OneTwoThree')
```

```
123 == 123.0
```

```
123 == '123'
```

```
1 == 'one'
```

Integer Operations

```
123 * 2
```

```
a = 123
a*2
```

```
12 + 4.0
```

```
doublea = a * 2
doublea
```

```
doublea
```

```
type(doublea)
```

Float Operations

```
123.5 * 2
```

```
b = 123.5
b*2
```

```
doubleb = b * 2
doubleb
```

```
type(doubleb)
```

String Operations

```
Handong
```

```
# Handong
```

```
43141
```

```
'Handong'
```

```
print(Handong)
```

```
print(#Handong)
# dfadfadsfasdf afadsfasdf asdfasdfasdfadsf
```

```
print('Handong')
```

```
onetwothree = 1
onetwothree
```

```
print(onetwothree)
```

```
print('onetwothree')
```

```
print("onetwothree")
```

```
print('"onetwothree"')
```

```
print("'onetwothree'")
```

```
print("""onetwothree""")
```

```
print("Handong is God's University")
```

```
print('Handong is God's University')
```

```
print('Handong is God₩'s University')
```

```
'Big'+'Data'
#'Big Data' 1
#'BigData' 2
```

```
var1 = 'Welcome to'
var2 = 'Introduction to Big Data'

print(var1+var2)
```

```
print(var1, var2)
```

```
'BigData'*2
```

```
len('BigData')
```

String Indexing

```
msg = 'God is good.'
print(msg)
```

```
print(msg[1])
```

```
print(msg[3])
```

```
print(msg[-1])
```

```
print(msg[1] + msg[2] + 'd')
```

Review

1. Given Var is a variable with value of 'In the beginning God created the heavens and the earth.'. What is the expected outcome of Var[-2].
2. Var[0]*3
3. print(Var+'Amen')
4. Which one returns "error"?

- print("Mom's Kitchen")
- print("Mom's Kitchen")
- print('Mom"s Kitchen')
- print('Mom's Kitchin')

```
# 1.
Var = 'In the beginning God created the heavens and the earth.'
Var[-2]
```

```
# 2.
Var[0]*3
```

```
# 3.
print(Var+'Amen')
```

```
# 4.
print("Mom's Kitchen")
```

```
# 4.
print("Mom₩'s Kitchen")
```

```
# 4.
print('Mom"s Kitchen')
```

```
# 4. Answer
print('Mom's Kitchin')
```

Converting data type

```
varint = 123
varfloat = 123.0
varstring = '123'

print(varint)
print(varfloat)
print(varstring)
```

```
varstring
```

```
varstring = int(varstring)
```

```
varstring
```

```
type(varstring)
```

```
int(varfloat)
```

```
int(varstring)
```

```
float(varint)
```

```
float(varstring)
```

```
str(varint)
```

```
str(varfloat)
```

Comments

```
BigData = 1
BigData
```

```
''' BigData = 1 '''
BigData
```

```
''' God
is
good.
All
the
time.'''
# God
# is
# good
# all the time
```

```
# God
# is
# good
# all the time
```

```
############################
#### Python Programming ###
############################
# Written by Kim # 21/07/16
```

Review

1. What will be shown on the screen if you run the codes below?
   >> BigData = 1
   >> BigData = str(BigData)
2. What is the data type?
   >> BigData = '1'
   >> int(BigData)
   >> type(BigData)
3. What is the data type?
   >> BigData = '1'

>> type(float(BigData))

4. Fix the following code to make it executable

>> mystring = 'He's Korean and I'm American.'

>> print(mystring)

```
# 1. What will be shown on the screen if you run the codes below?
BigData = 1
BigData = str(BigData)
BigData
```

```
# 2. What is the data type?
BigData = '1'
int(BigData)
type(BigData)
```

```
# 3. What is the data type?
BigData = '1'
print(type(float(BigData)))
print(BigData)
```

```
# 4. Fix the following code to make it executable
# 'He's Korean and I'm American.'
mystring = 'HeW's Korean and IW'm American.'
#mystring = "He's Korean and I'm American."
print(mystring)
```

Lecture 2. Pythong Programming - 2

PART 2

Operations

Arithmetic operators

+, - „ *, /, //, %, +=, -=, *=, /=

```
1 + 1
```

```
1 - 1
```

```
2 * 2
```

```
2 ** 3
```

```
10 / 2
```

```
10 // 2
```

```
10 % 2
```

```
varA = 10
varA
varA + 1
```

```
varA = 10
varA = varA+1
varA
```

```
varA = 10
varA -= 1
varA
```

```
varA = 10
varA = varA-1
varA
```

Relational operators

```
1<2
```

```
1>2
```

```
1<=2
```

```
1>=2
```

```
1==2
```

```
1!=2
```

```
#(bmi: Body mass index)
weight = 100
height = 1.83
bmi = weight / (height * height)

print("My weight is", weight , "kg, and height is", height , "m.")
print("BMI is", bmi)
```

```
if( bmi > 25 ):
  print("overweighted.")
```

Logical operators

```
(1 < 2 ) and ( 2 < 3 )
```

```
(1 == 2 ) and ( 2 < 3 )
```

```
(1 < 2 ) or ( 2 < 3 )
```

```
(1 == 2 ) or ( 2 < 3 )
```

```
1 < 2
```

```
not ( 1 < 2 )
```

Function

Defining a function

```
def plus(num1, num2):
  return num1 + num2
```

```
type(plus)
```

```
plus(2, 12)
```

```
result_plus = plus(2, 12)
result_plus
```

```
def mul(num1, num2):
  return num1 * num2
```

```
mul(2, 12)
```

```
result_mul = mul(2, 12)
result_mul
```

```
# Comparison one without return
def mul(num1, num2):
  result = num1 * num2
  print(result)
```

```
mul(2, 12)
```

```
result_mul = mul(2, 12)
result_mul
```

Review

```
def abc(a,b):
  return a * b
abc(3,5)
```

```
(abc(3,5) == 15) or (abc(3,5) > 15)
```

```
(abc(3,5) == 15) and (abc(3,5) > 15)
```

Review

```
def kim(a,b):
  print(a,b)
kim("Hi","I'm","Kim")
```

```
def kim(a,b):
  print(a,b)
kim(1,2)
```

```
def myname(a):
  return "Messi"
myname('Leo')
```

```
myname('Kim')
```

Class

Defining a class

```
class Calcul:
  def setdata(self, first, second):
    self.first = first
    self.second = second
```

```
a=Calcul()
a.setdata(5,7)
```

```
a.first
```

```
a.second
```

```
b = Calcul()
Calcul.setdata(b,2,3)
```

```
b.first
```

```
b.second
```

```
class Calcul:
  num1 = 5
  num2 = 10
  def add(self):
    result = self.num1 + self.num2
    return result
```

```
Calcul.num1
```

```
Calins = Calcul()
Calins.add()
```

```
class Calcul:
  def __init__(self, first, second):
    self.num1 = first
    self.num2 = second
  def add(self):
    result = self.num1 + self.num2
    return result
```

```
a = Calcul(5,7)
```

```
type(a)
```

```
a.add()
```

Useful comments

Introspection

```
b = [1,2,3,4,5]
```

```
b?
```

```
print?
```

```
import pandas as pd
```

```
pd.DataFrame?
```

```
def mul(num1, num2):
  return num1 * num2
mul?
```

```
def mul(num1, num2):
  '''
  Receive two numbers

  Returns
  -----
  Multiplication of two numbers
  '''
  return num1 * num2
mul?
```

```
def mul(num1, num2):
  '''
  Receive two numbers

  Returns
  -----
  Multiplication of two numbers
  '''
  return num1 * num2
mul??
```

Shortcuts

```
data = 'God is good all the time. All the time God is good.'
hgu = 'Learning Engagement'
```