Lab 1: Basic Python Programming

CPE232 Data Models

[1] Variable

→ 1.1 Number Variable

```
num = 100 #integer variable
num2 = 12.5 #float variable
print(num)
print(num2)

print(num + num2) #addition
print(num - num2) #subtraction
print(num * num2) #multiplication
print( num / num2) #division
```

→ 1.2 String Variable

```
#string variable
string = "Data Models"
print(string) #print complete string

print("Hello " + string)  #print concatenated string
print(string[0])  #print first character of the string
print(string[:4])  #print first to 4th character of the string
print(string[5:])  #print 6th to last character of the string
print(string[1:4])  #print 2nd to 4th character of the string
print(string * 2)  #print string 2 time
```

✓ 1.3 Boolean Variable

1.4 List Variable

```
#list variable
list = ["Data",20,123.23,40,50]
another_list = ["Models",60]
print(list)
                            #print complete list
                           #print first element of the list
print(list[0])
print(list[1:3])
                            #print 2nd to 3rd element of the list
                            #print 3rd to last element of the list
print(list[2:])
print(another_list)
                            #print complete another_list
print(another_list * 2)
                           #print another_list two times
print(list + another_list) #print concatenated list
list[0] = "CPE232"
                            #change first element of the list
print(list)
                            #print complete list
```

→ 1.5 Tuple Variable

```
#tuple variable
tuple = ("Data",20,123.23,40,50)
```

✓ 1.6 Dictionary Variable

```
#dictionary variable
dictionary = {"name":"Alice", "age":21}
another_dictionary = {}
another_dictionary["name"] = "Bob"
another_dictionary["age"] = 21
print(dictionary)
                                   #print complete dictionary
print(dictionary["name"])
                                   #print value for specific key
print(dictionary.keys())
                                   #print all the keys
print(dictionary.values())
                                   #print all the values
print(dictionary.items())
                                   #print all the items
print(another_dictionary)
                                   #print complete another_dictionary
```

√ [2] Control Flow

✓ 2.1 IF ... ELIF ... ELSE

```
number = 123
number2 = 34

if number > number2:
    print("number is greater thanu number2")
elif number < number2:
    print("number is less than number2")
else:
    print("number is equal to number2")</pre>
```

[3] Loop


```
#for loops
for num in range(0,10):
    print(num)

#for loop with list

list = ["Alice","Bob","Charlie","Daisy"]
for name in list:
    print(name)

#continue in for loop

list = [1,23,7,"hello",True,1123,43,23,12]

for element in list:
    if type(element) != int:
        continue
    print(element)

#break in for loop

list = [1,23,7,"hello",True,1123,43,23,12]
```

```
for element in list:
    if type(element) != int:
        break
    print(element)
```

3.2 While loop

```
#while loop
list = ["Alice", "Bob", "Charlie", "Daisy"]
count = 0
while count < len(list):
    print(list[count])
    count += 1
#continue in while loop
list = [1,23,7,"hello",True,1123,43,23,12]
count = 0
while count < len(list):</pre>
    if type(list[count]) != int:
       count += 1
        continue
    print(list[count])
    count += 1
#break in while loop
list = [1,23,7,"hello",True,1123,43,23,12]
count = 0
while count < len(list):</pre>
    if type(list[count]) != int:
        break
    print(list[count])
    count += 1
```

[4] Function

```
#define function
def function_name (arg1, arg2):
   return arg1 + arg2
#calling function
function_name(1,2)
#define function with default argument
def function_with_default_arg(arg1, arg2 = 10, arg3 = 20 , arg4 = 30):
   return arg1 + arg2 + arg3 + arg4
result_1 = function_with_default_arg(1)
result_2 = function_with_default_arg(1,2,5)
result_3 = function_with_default_arg(1,2,5,10)
print(result_1)
print(result_2)
print(result_3)
#multiple agument
def function_with_multiple_arg(*args):
   print(args)
   print(type(args))
    sum = 0
    for num in args:
        sum += num
    return sum
function_with_multiple_arg(1,2,3,4,5)
```

```
#lambda function
lambda_function = lambda arg1, arg2: arg1 + arg2
print(lambda_function(1,2))
```

√ [5] File Handling

✓ 5.1 Text File

```
with open("test.txt","w") as file:
    file.write("Hello World")

with open("test.txt","r") as file:
    print(file.read())
```

▼ 5.2 CSV File

```
import csv
with open("test.csv","w",newline='') as file:
    writer = csv.writer(file)
    writer.writerow(["Name","Surname"])
    writer.writerow(["Alice","Johnson"])
    writer.writerow(["Bob","Smith"])

import csv
with open("test.csv","r") as file:
    reader = csv.reader(file)
    for row in reader:
        print(row)
```

[4] Libraries

- ✓ 4.1 Numpy
- ✓ import numpy library

```
import numpy as np
```

ndarray initialization

Construct using python list

```
# 1d ndarray from 1d python list
list_a1=[1,2,3.5]
arr_a1=np.array(list_a1)
arr_a1

# 2d ndarray from 2d python list (list of list)
list_a2=[[1,2],[3,4],[5,6]]
arr_a2=np.array(list_a2)
arr_a2

list_a3=[[[1,2],[2,3]],[[3,4],[4,5]]]
arr_a3=np.array(list_a3)
arr_a3

or construct using some numpy classes and functions
np.zeros(5)
```

https://colab.research.google.com/drive/1DJTeawP2tCv89cUO1nlWaUJ1oE75zWxk#scrollTo=kMp4qD-6FEwu&printMode=true

```
np.ones((3,4),dtype=float)
np.full((4,),999)
np.arange(3,10,2)
np.linspace(10,15,11)
np.random.choice(['a','b'],9)
np.random.randn(10)
ndarray properties
list_a=[[1,2,3,4],[5,6,7,8],[9,10,11,12]]
arr_a=np.array(list_a)
arr_a
arr_a.ndim
arr_a.shape
→ (3, 4)
arr_a.dtype
arr_a.size
```

Reshaping & Modification

from this original ndarray

arr_a

sometimes you may resize for same dimension where only known some dimension, insert -1 for unknown len

[[7, 8, 9], [10, 11, 12]]])

Would you like to try this?

```
arr_a.reshape((-1,5))
```

```
Traceback (most recent call last)
     <ipython-input-9-286d5aa6424c> in <cell line: 0>()
     ---> 1 arr_a.reshape((-1,5))
     ValueError: cannot reshape array of size 12 into shape (5)
[Q1] From the above cell, explain in your own words why it worked or did not work.
Ans: It not work because 5 can not multiply to 12 in integer number
Next, try to append any value(s) into exist 2darray
np.append(arr_a,13)
\Rightarrow array([ 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13])
np.append(arr_a,arr_a[0])
np.append(arr_a,arr_a[0].reshape((1,-1)),axis=0)
np.append(arr_a,arr_a[:,0].reshape((-1,1)),axis=1)
np.concatenate([arr_a,arr_a])
np.concatenate([arr_a,arr_a],axis=1)
indexing & slicing
from this original array again
arr_a
try to access all element at the first row
arr_a[1]
then you would like to access the second element from the first row
arr_a[1][2]
arr_a[1,2]
Next, try to access all element start from 1th in the first row
arr_a[1,1:]
arr_a[:2,1:]
sometimes you may specify some row number using list within indicing
arr_a[[1,2,1],1:]

    Boolean slicing

based on this original array
arr_a
try to filter all elements which more than 5
```

```
arr_a>5
```

Next, try to filter all elements which more than 5 and less than 10

Run the cell below and answer a question.

```
arr_a[(arr_a>5)&(arr_a<10)]

→ array([6, 7, 8, 9])
```

[Q2] From the above cell, explain in your own words how the output came about?

Ans: we print the filterd values in the arr_a those are greater than 5 and less than 10

Try running the cell below.

```
arr_a[(arr_a>5) and (arr_a<10)]
```

```
ValueError Traceback (most recent call last)
<ipython-input-14-78eb1746bbfd> in <cell line: 0>()
----> 1 arr_a[(arr_a>5) and (arr_a<10)]
```

ValueError: The truth value of an array with more than one element is ambiguous. Use a.any() or a.all()

[Q3] Explain in your own words why the above cell gives an error.

Ans: Because "and" can only use in the simple boolean for this problem can not use this is boolean array not simple boolean

[Q4] And what should be written instead so that the code is error-free?

Ans: replace "and" into "&"

Basic operations

```
list_b=[[1,2,3,4],[1,2,3,4],[1,2,3,4]]
arr_b=np.array(list_b)
arr_b
```

This is some operations for only 1 array

```
np.sqrt(arr_b)
```

This is some operations for 2 arrays with the same shape

```
arr_a-arr_b

np.add(arr_a,arr_b)
```

Next, try to operate with 1 array and one numeric variable

```
arr_a*3
1+arr_a**2
```

Try to play with 2 arrays with different shape

```
arr_c=np.array([1,2,3])
arr_d=np.array([[3],[5],[8]])
arr_c-arr_d
```

→ Basic aggregations

```
arr_a
arr_a.sum()
arr_a.mean()
arr_a.min()
arr_a.max()
arr_a.std()
```

ndarray axis

```
array([15, 18, 21, 24])
```

arr_a.sum(axis=1)

→ array([10, 26, 42])

[Q5] Summarize the value of the argument *axis*, what is the value for row-wise summation and column-wise summation, respectively? Ans: 0: Column-wise operation and 1: Row-wise operation

4.2 Pandas

→ Series

```
import pandas as pd
import numpy as np

pd.Series(np.random.randn(6))

pd.Series(np.random.randn(6), index=['a','b','c','d','e','f'])
```

Constructing Dataframe

Constructing DataFrame from a dictionary

```
d = {'col1':[1,2], 'col2': [3,4]}
df = pd.DataFrame(data=d)
df
```

```
col1 col2 

0 1 3 1 1 2 4 1
```

Next steps: Generate code with df View recommended plots New interactive sheet

df2 = pd.DataFrame(data=d2)
df2

_		Name	Age	
	0	Joe	20	ılı
	1	Nat	21	+/
	2	Harry	19	
	3	Sam	20	
	4	Monica	22	

Next steps: Generate code with df2 View recommended plots New interactive sheet

Constructing DataFrame from a List

```
marks_list = [85.10, 77.80, 91.54, 88.78, 60.55]

df3 = pd.DataFrame(marks_list, columns=['Marks'])
df3
```

Creating DataFrame from file

Read csv file from path and store to df for create dataframe
df = pd.read_csv('nss15.csv')

df

Viewing DataFrame information

 $(.shape, .head, .tail, .info, select \, column, \, .unique, \, .describe, \, select \, low \, with \, .loc \, and \, .iloc)$

Check simple information

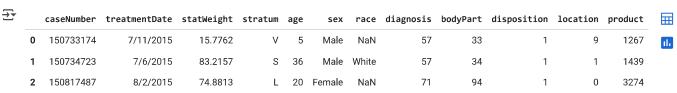
Check dimension by .shape
df.shape

→ (334839, 12)

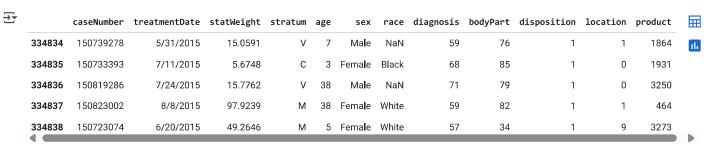
Display the first 5 rows by default
df.head()

₹		caseNumber	treatmentDate	statWeight	stratum	age	sex	race	diagnosis	bodyPart	disposition	location	product	Ħ
				5 cu cinc_8c	50.00	8-	55%		w=wBes=s	204, 4. 4	u_opes_c_e		p. 0444	
	0	150733174	7/11/2015	15.7762	V	5	Male	NaN	57	33	1	9	1267	ılı
	1	150734723	7/6/2015	83.2157	S	36	Male	White	57	34	1	1	1439	
	2	150817487	8/2/2015	74.8813	L	20	Female	NaN	71	94	1	0	3274	
	3	150717776	6/26/2015	15.7762	V	61	Male	NaN	71	35	1	0	611	
	4	150721694	7/4/2015	74.8813	L	88	Female	Other	62	75	1	0	1893	

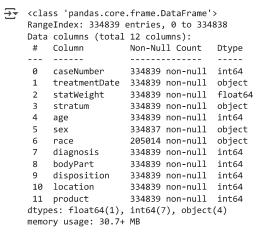
Display the first 3 rows
df.head(3)



Display the last 5 rows by default
df.tail()



Overview information of dataframe
df.info()



Select column, multiple column, with condition

df.columns

#select single column
df['age']

→ *		age	
	0	5	
	1	36	
	2	20	
	3	61	
	4	88	
	•••		
	334834	7	
	334835	3	
	334836	38	
	334837	38	
	334838	5	
	334839 rc	ws × 1	columns

dtype: int64

df.age

$\overline{}$		
→		age
	0	5
	1	36
	2	20
	3	61
	4	88
	•••	
	334834	7
	334835	3
	334836	38
	334837	38
	334838	5

334839 rows × 1 columns

dtype: int64

#select multiple column
df[['treatmentDate','statWeight','age','sex']]

sex	age	statWeight	treatmentDate	
Male	5	15.7762	7/11/2015	0
Male	36	83.2157	7/6/2015	1
Female	20	74.8813	8/2/2015	2
Male	61	15.7762	6/26/2015	3
Female	88	74.8813	7/4/2015	4
Male	7	15.0591	5/31/2015	334834
Female	3	5.6748	7/11/2015	334835
Male	38	15.7762	7/24/2015	334836
Female	38	97.9239	8/8/2015	334837
Female	5	49.2646	6/20/2015	334838
			ws × 4 columns	334839 ro

Viewing the unique value

df.race.unique()

Describe

df['age'].describe()

Select row with condition

```
#select by condition
df[df['sex'] == 'Male']

#select by multiple condition
df[(df['sex'] == 'Male') & (df['age'] > 80)]
```

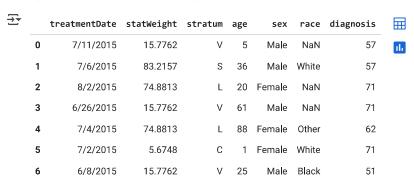
Select row with .iloc

```
# select row by .iloc
df.iloc[10:15]

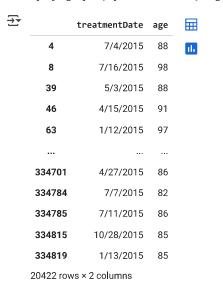
# select column by .iloc
df.iloc[:,[0,1,2,3,4]]
```

Select column and row with .loc

select column and low by .loc
df.loc[:6,'treatmentDate':'diagnosis']



select row by condition
df.loc[df['age']>80, ['treatmentDate', 'age']]



[Q6] What is the difference between .iloc and .loc?

Ans: iloc is the integer based indexing loc is the user's label based indexing