Python Programming Intermediate Level Course v8

SINGAPORE WORKFORCE SKILLS QUALIFICATIONS

Trainer: Truman Ng

Website:www.tertiarycourses.com.sg
Email: enquiry@tertiaryinfotech.com

About the Trainer

Truman Ng graduated with Bachelor Degree in Electrical Engineering from NUS in year 2002. He designed Artificial Intelligence (AI) controller for DC-DC Power Convertor by using Fuzzy Logic and Neural Network (NN) as his university Final Year Project.

Truman has over 15 years project experiences across Database & Web Design, PLC machinery, Data Center Design , Structure Cabling System(SCS) and Enterprise Network Design and Implementation. He used to be a network architect for Hewlett Packard, working with a group of virtual team from the US in handling network design and projects in the States.

Truman is the founder of Nexplore (S) Pte Ltd. He provides solutions of Cloud SaaS, IaaS & PaaS and Software Defined Network (SDN), VoIP and Internet Security. He was engaged by Huawei Global Training Center to provide 60+ consultations and trainings for Internet Service Providers(ISP) from Malaysia, Singapore, Brunei, Philipines, Australia, Poland, Iran, South Africa, Swaziland, Cote Dlvoire, Syria, Uzbekistan, New Zealand and countries over the world.

As achievement, Truman has successfully completed 100+ IT network projects for Bank, Hotel and Factory within 5 years.

Truman is certified in PMP(Project Management Professional), CBP (Certified Bitcoin Professional), Cisco Professional level Certifications: CCNP, CCIP, CCDP, HP Professional Certification: Ase and Huawei ,Huawei Expert and Professional Certifications: HCIE R&S, HCNP, HCNA Cloud, HCNA Security, etc.

Let's Know Each Other...

Say a bit about yourself
- Name
- What Industry you are from?
- Do you have any prior knowledge in programming and Python
- Why do you want to learn Python
- What do you expect to learn from this course?

Ground Rules

- Set your mobile phone to silent mode
- Participate actively in the class. No question is stupid.
- Mutual respect. Agree to disagree.
- One conversation at one time.
- Be punctual. Back from breaks on time.
- Exit the class silently if you need to step out for phone call, toilet break etc.
- 75% attendance is required

Ground Rules for Virtual Training

- Upon entering, mute your mic and turn on the video. Use a headset if you can
- Use the 'raise hand' function to indicate when you want to speak
- Participant actively. Feel free to ask questions on the chat whenever.
- Facilitators can use breakout rooms for private sessions.

WSQ and SSG TG Application Form

Please fill up the following WSQ and SSG TG Application Form for TRACOM survey, e-cert generation and WSQ funding

https://forms.gle/pJ2WxHZ3fyRbDLVu6

Digital Attendance

- You need to take digital attendance in the AM and PM.
- Please download the mySkillsFuture apps below to take your digital attendance for WSQ and SFC courses.

Guidelines for Facilitators

1. Once all the participants are in and introduce themselves
2. Goto gallery mode, take a snapshot of the class photo - makes sure capture the date and time
3. Start the video recording (only for WSQ courses)
4. Continue the class
5. Before the class end on that day, take another snapshot of the class photo - makes sure capture the date and time
6. For NRIC verification, facilitator to create breakout room for individual participant to check (only for WSQ courses)
7. Before the assessment start, take another snapshot of the class photo - makes sure capture the date and time (only for WSQ courses)
8. For Oral Questioning assessment, facilitator to create breakout room for individual participant to OQ (only for WSQ courses)
9. End the video recording and upload to cloud (only for WSQ courses)
10. Assessor to send all the assessment records, assessment plan and photo and video to the staff (only for WSQ courses).

Prerequisite

This is an intermediate course.

- Basic Python is assumed.

Learning Outcomes

By end of the course, learners will be able to

- Understand and code Python comprehensions
- Understand and code Python generators
- Manage files and folders in Python
- Understand and code OOP Classes and Objects
- Understand and code OOP Inheritance
- Setup and use databases in Python
- Understand and code Exceptions to handle errors in Python

Agenda

Topic 1 Comprehensions & Generators

- Comprehension Syntax
- Types of Comprehension
- Generator Syntax
- Types of Generators

Topic 2 File and Directory Handling

- Read and Write Data to Files
- Manage File and Folders with Python OS Module
- Manage Paths with Python Pathlib Module

Topic 3 Object Oriented Programming

- Introduction to Object Oriented Programming
- Create Class and Objects
- Method and Overloading
- Initializer & Destructor
- Inheritance
- Polymorphism

Agenda

Topic 4 – Database

- Setup SQLite3 database
- Apply CRUD operations on SQLite3
- Integrate to external databases

Topic 5 – Error Handling Using Exception

- Exceptions versus Syntax Errors
- Handle Exceptions with Try and Except blocks
- The Else clause
- Clean up with Finally

Practice Session

Final Assessment

- Written Assessment (Q&A)
- Written Assessment (Case Study)
- Oral Questioning

Download Course Material

- You can download the course material from Google classroom https://classroom.google.com
- Enter the class code below to join the class on the top right.
- Goto Classwork>> Course material
- If you cannot access the Google Classroom, please inform trainer or staff.

z4t5rg6

CERTIFICATE

Two e-certificates will be awarded to trainees who have demonstrated competency in the WSQ Python Programming Intermediate Level assessment and achieved at least 75% attendance.

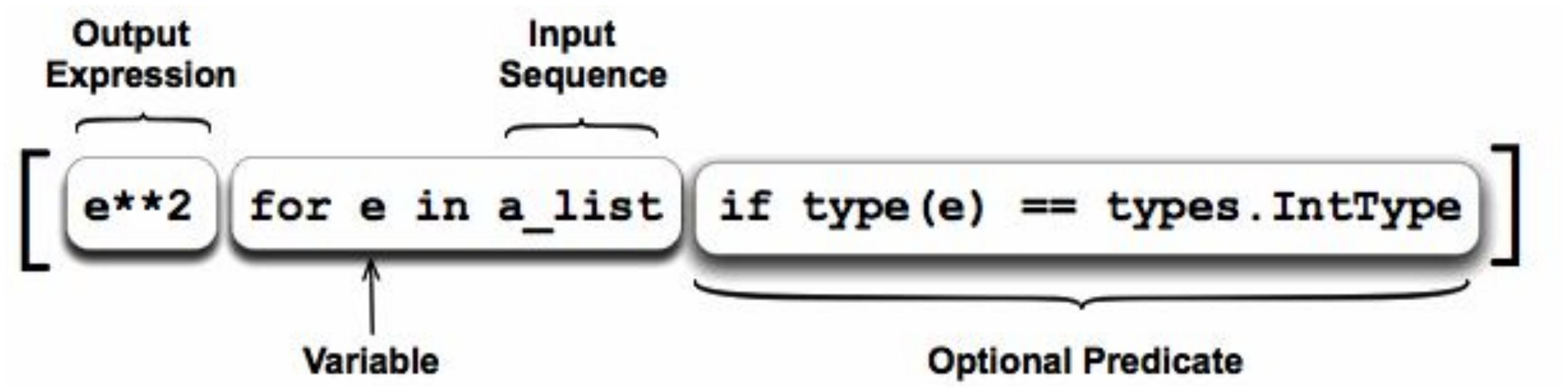
- A SkillsFuture WSQ Statement of Attainment (SOA) – ICT-DIT-3001-1.1 Analytics and Computational Modelling under the National Infocomm Competency Framework (NICF) issued by WSG
- Certification of Completion issued by Tertiary Infotech Pte Ltd

Topic 1
Comprehension & Generators

Comprehension

- Comprehensions provide a concise way to create lists.
- It consists of brackets containing an expression followed by a for clause, then zero or more for or if clauses.
- There are 3 types of comprehensions: list, dict, and set.

List Comprehension Syntax



Examples:

[i**2 for i in range(1,11)]
[i**2 for i in range(1,11) if i%2 != 0]

Ex: List Comprehension

Create a comprehensive list of 20 square numbers that are not divisible by 3 and 5

Output should be
[1,4,16,49,....]

Time: 5 min

Ex: List Comprehension

Use comprehension method to extract the first and last letter of each word in the sentence below

"Today is a sunny day and a great day"

Hint: Use the split function to split out the words in the sentence

Time : 10 min

Double Loop in Comprehension

- You can also apply double loop to the list comprehension.
- The leftmost loop is the outer loop and the rightmost loop is the inner loop.

Examples:

```
[i*j for i in range(1,3) for j in range(1,5)]
```

Set Comprehension

- Comprehension can also create set.
- The syntax for set comprehensions is almost identical to that of list comprehensions, but it uses curly brackets instead of square brackets.

Example

```
{i*i for i in range(1,11)}
```

This is equivalent to

```
set([i*i for i in range(11)])
```

Ex: Set Comprehension

Extract the vocabulary of the sentence below and exclude {is, a, and}

"Today is a sunny day and a great day"

Time : 10 min

Dict Comprehension

A dictionary comprehension takes the form {key: value for (key, value) in iterable}

Example:

```
{i:i*i for i in range(1,11)}


name = ['Ally','Belinda','Steve']
height = [160,158,170]
{name:height for name,height in zip(name,height)}
```

Ex: Dict Comprehension

Given 2 dictionaries

d1 = {'Tom': 14, 'Patrick': 12, 'Sean': 15}

d2 = {'Jeanne': 14, 'Marie': 12}

Merge the two dictionaries into 1 dictionary

Hint: {......for d in (d1,d2) ....in d.items() }

Time : 10 min

Issues with Comprehension

If you want to calculate the sum of the squares of a range of numbers

sum([i**2 for i in range(20)])

That creates a list in memory just to throw it away once the reference to it is no longer needed, which is wasteful.

Also [i*i for i in range(100000...0)] can take up a large memory

Generator

Generator (or Generator Object) is like a Ticket Machine

1. Store value of last ticket
2. Generate the new value upon button press

Generator Expression & Function

There are 2 types of generators:

- Generator Expression
- Generator Function

Generator Expression Example

Generator Expression is enclosed in brackets

```
(i**2 for i in range(20))
sum((i**2 for i in range(20)))
```

This help to save a lot of memory!

Next

a = (n*n for n in range(10))

a is a generator object

next(a) will yield the next element

Ex: Generator Expression

Given the first and last names below:

firstname = ['ally','jane','belinda']

lastname = ['tan','wee','ng']

Create a generator expression to return a capitalized name eg

"Ally Tan"

Time: 5 min

Generator Function

Normal Function

```
def square(n):
    a = []
    for i in range(n):
        a.append(i*i)
    return a
```

Generator Function

```
def square(n):
    for i in range(n):
        yield i*i
```

- return is replaced with yield
- Get rid of the list

Ex: Generator Function

Write a generator function to return a fibonacci number

1,1,2,3,5,8,13,......

Topic 2
File and Directory Handling

OS Module

- The OS module in Python provides a way of using operating system dependent functionality.
- The functions that the OS module provides allows you to interface with the underlying operating system that Python is running on – be that Windows, Mac or Linux
- To import the Python OS module, use

  import os

Mount Google Drive

You can mount your Google Drive on your runtime using an authorization code

```
from google.colab import drive
drive.mount('/content/drive')
```

OS Directory Functions

os.getcwd() - Check current working directory
os.chdir(path) - Change directory
os.mkdir(path)   - Create directory
os.listdir(path) - List the files in the path
os.remove(path) - Remove a file
os.rmdir(path) - Remove directory

OS Path Module

- The OS Path module contains some useful functions on pathnames.
- Some useful functions include:

os.path.dirname - Returns the directory name
os.path.join(path…) - Joins paths

Open a file

```
f = open('file.txt','r')      - reading a file
f = open('file.txt','w')      - writing a file
f = open('file.txt','a')      - appending a file
```

Write Data to a File: Method 1

```
f = open('test.txt','w')
for i in range(10):
    f.write('This is line {}\n'.format(i+1))
```

Append Data to a File

```
f = open('test.txt','a')
for i in range(10):
    f.write('This is line {}\n'.format(i+1))
```

Close a File

To close a file from writing, you need to enter the command:

```
f.close()
```

Write Data to a File: Method 2

```
with open("test.txt",'w') as f:
    for i in range(10):
        f.write("Hello World {}\n".format(i))
```

Ex : Write Data to a File

Prompt user to enter the name, weight and height, and compute the bmi (=weight/height*height)

```
name = input('Enter name: ')
height = float(input('Enter height (m): '))
weight = float(input('Enter weight (kg): '))
```

enter the data into a file in the format below. convert name to lower case and round bmi to 2 decimal places.

```
name     height   weight   bmi
alfred   1.7      72.3     25.02
steve    1.75     68.0     22.2
```

Read Data from a File: Method 1

```
f = open("user.txt","r")
for i in f:
    print(i.strip())
f.close()
```

Read Data from a File: Method 2

```
with open("user.txt",'r') as f:
    for i in f:
        print(i.strip())
```

Read Data from a File: Method 3

```
for i in open('user.txt','r'): print (i.strip())
```

Read Data from File with Pandas

```
import pandas as pd

df = pd.read_csv('test.txt',header=None)
```

Ex: Read Data from a File

Read the users.txt from previous Ex.

Prompt user for the name and return the bmi for the user

Hint:

```
pd.read_csv('user.txt',header=0,sep='\t',index_col='name')
df.loc[name.lower()].bmi
```

Topic 3
Object Oriented Programming

Class

A class is the blueprint of an object. You can create a class as shown below:

```
class Animal:
    color = "black"
    legs = 4

    def walk(self):
        print("walk like an animal")

    def type(self):
        print('This animal has {} legs'.format(self.legs))
```

Property/Attribute

Function/Method/Behavior

Object

An object is an instance of a class eg

```
dog = Animal()

print(dog.color)
print(dog.legs)
dog.walk()
```

Ex: Class & Object

Create a Counter class with a property of count. This class has 3 functions

1. increment the count by 1
2. increment the count by n
3. reset the count to 0

Time: 10 min

Initializer

Initializer (a.k.a constructor) is a special function of a class to initialize the class data or get inputs.

```
def __init__(self,color,legs):
    self.color = color
    self.legs = legs
```

Initializer Examples

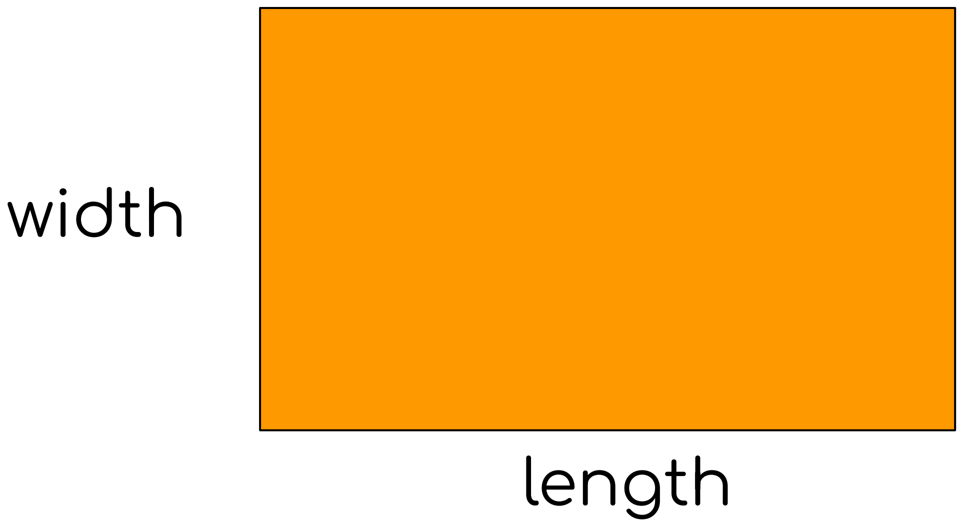
```
class Animal:
    def __init__(self,color,legs):
        self.color = color
        self.legs = legs
    def type(self):
        print('This {} animal has {} legs'.format(self.color,self.legs))

class Person:
    def __init__(self,name, height):
        self.name = name
        self.height = height
    def disp_height(self):
        print("{} height is {}cm".format(self.name, self.height))
```

Ex: Initializer

Write a Rectangle Class with an initializer to accept two properties: length and width. The Rectangle Class has a method to compute the area of the rectangle

Time: 10 min



Destructor

Destructor is a special function which is invoked when the instance is destroyed

```
__del__( self )
```

To delete an object instance,

```
del object_name
```

Destructor Example

```
class Animal:
    def __init__(self,color,legs):
        self.color = color
        self.legs = legs
    def __del__(self):
        print("The animal has destroyed")
    def type(self):
        print('This {} animal has {} legs'.format(self.color,self.legs))


cat = Animal('white',3)
cat.type()
del cat
```

Class vs Instance Variable

- Class variable can be accessed by all object instances
- Instance variable can only be accessed by the instance

Eg

$$\left.\begin{array}{l}\texttt{legs = 4}\\ \texttt{color = "white"}\end{array}\right\}\text{Class Variables}$$

```
def __init__(self,legs,color,name):
```

$$\left.\begin{array}{l}\quad\texttt{self.legs = legs}\\ \quad\texttt{self.color = color}\end{array}\right\}\text{Instance Variables}$$

Class and Instance Variable Examples

```
class Animal:
    color = 'white'
    legs = 4
    def __init__(self,color,legs):
        self.color = color
        self.legs = legs
    def instanceType(self):
        print('This {} animal has {} legs'.format(self.color,self.legs))
    def classType(self):
        print('This {} animal has {} legs'.format(Animal.color,Animal.legs))

spider = Animal('brown',8)
spider.instanceType()
spider.classType()
```

Ex Employee Class

Create a Employee class with the following properties

- empCount : Number of Employees (class variable)
- accept two properties: name, salary (instance variables)
- a method to display employee count
- a method to display employee name and salary

When an employee object is created, the empCount increase by 1. When an employee object is deleted, the empCount reduce by 1

Time: 10 mins

Inheritance

The child class inherits all the attributes and methods from the base (parent) class.

```
class Dog(Animal):
    pass

dog = Dog('brown',4)
dog.type()
```

Overwriting Methods

To overwrite the parent method, use the same method name Eg

```
class Dog(Animal):
    def __init__(self,color):
        super().__init__(color,4)
    def walk(self):
        print("walk like an dog")
    def talk(self):
        print("woof woof woof")
```

Initializing Child Class

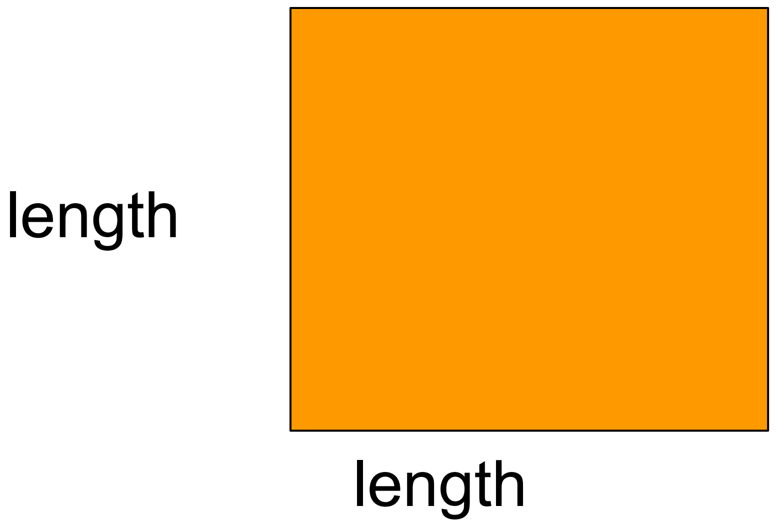
Use super() to initialize the parent

```
class Dog(Animal):
    def __init__(self,color):
        super().__init__(color,4)
```

Ex: Inheritance

- Create a Square Class inherited from Rectangle class and initializes the length.
- Add new method to computer the perimeter = 4*length

Time: 10 min



Ex: Inheritance

Create FullTimeStaff and PartTimeStaff child classes inherited from the Employee class

- FullTimeStaff has extra attribute of leave beside name and salary
- PartTimeStaff has name and hourly rate attributes, but no salary and leave attributes
- Employee Count only count FullTimeStaff

Time : 15 mins

Polymerism

```
class Dog(Animal):
    def __init__(self,color):
        super().__init__(color,4)
    def talk(self):
        print("woof woof woof")
class Cat(Animal):
    def __init__(self,color):
        super().__init__(color,4)
    def talk(self):
        print("meow meow meow")
d1 = Dog("white")
c1 = Cat("black")
def sound(any):
    any.talk()
sound(d1)
sound(c1)
```

The output behavior depends on the input object types

Recap of Day 1 Topics

Topic 4
Database

Database API

The Python standard for database interfaces is the Python DB-API which support many databases such as

- SQLite
- MySQL
- PostgreSQL
- Microsoft SQL
- Oracle SQL
-

Create SQLite Database

```
import sqlite3

db= sqlite3.connect('test.db')
```

Create Table

```
db.execute('CREATE TABLE student
(name text, rank int)')
db.commit()
```

CRUD Operations

There are 4 major operations

- Create/Insert
- Read
- Update
- Delete
-

Create/Insert Data

```
db.execute('INSERT INTO test (name,rank)
values (?,?)',('One',1))

db.commit()
```

Read Data

```
list = db.execute('SELECT * FROM student
ORDER BY rank')

for i in list:
    print(i)
```

Update Data

```
db.execute('UPDATE test SET name=?
WHERE rank=?',('John',3))
```

Delete Data

```
db.execute('DELETE FROM test WHERE
rank = 4')
```

Challenge: Database

Create a table "subjects" to your DB, students, classes. Insert the following records

subject	students	classes
English	200	10
Chinese	50	8
Math	80	12
Science	80	12

Time: 10min

Topic 5
Error Handling Using Exception

What is Exception?

- An exception is an event, which occurs during the execution of a program that disrupts the normal flow of the program's instructions.
- When a Python script raises an exception, it must either handle the exception immediately otherwise it terminates and quits.
- Example, error occur if input is not integer for the following command

```
a = int(input('Enter an integer : '))
print("You enter ",a)
```

Exception Syntax

```
try:
   You do your operations here;
except ExceptionI:
   If there is ExceptionI, then execute this block.
except ExceptionII:
   If there is ExceptionII, then execute this block.
   ......................
else:
   If there is no exception then execute this block.
finally:
     Execute this block whether or not any exception
```

Common Exception Errors

IndexError: index is not found in a sequence.

IOError: input/ output operation fails

TypeError: invalid data type

ValueError: arguments have invalid values specified

Try-Except Example

```
try:
    a = int(input('Enter an integer : '))
    print("You enter ",a)
except:
    print("Please enter an integer")
```

Try-Except-Else Example

```
try:
    a = int(input('Enter an integer : '))
except ValueError:
    print("Please enter an integer")
else:
    print("You enter ",a)
```

Try-Except-Else-Finally Example

```
try:
    a = int(input('Enter an integer : '))
except ValueError:
    print("Please enter an integer")
else:
    print("You enter ",a)
finally:
    print("Please try again")
```

Example of Exception

```
try:
    f = open('test888.txt','r')
except IOError:
    print("file does not exist")
```

Ex: Exceptions

```
import random
die = random.choice([1,2,3,4,5,6])
# print('The correct answer is ',die)
```

- Prompt user to guess an integer number
- if guess correctly, jump out from the program.
- If guess wrongly, ask user to guess again
- Run the program from command line as Sublime Text does not support input

Summary
Q&A

Practice Sesion

- Open up the Practice Section from the Google Classroom
- It is a scenario based question
- Code the python program

Time: mins

Final Assessment

Written Assessment (Q&A)

This is the Written Assessment (Q&A)

Duration: 50 mins

1. The assessor will pass the questions in hardcopy to you. There are 10 questions. You need to answer all the questions.
2. This is an open book exam that must be completed individually.
3. You need to code a Python program in a new Google Colab project.
4. You can use Python IDE or Google Colab to work out the answer.

Written Assessment (Case Study)

This is a Written Assessment (Case Study)

Duration: 80 mins

1. The assessor will pass the case study in hardcopy to you.
2. You need to code a Python program in a new Google Colab project.
3. This is an open book exam that must be completed individually.

Submission Procedure:

1. At the Google Colab, Goto File -> Download as .ipynb
2. Rename your file as this format CRS-Q-0039897-ICT_JohnTan_XXXX567A_
3. Email the Jupyter file to the assessor

Feedback

https://goo.gl/R2eumq

TRAQOM Survey

- You will receive a TRAQOM link after the class.
- Please submit TRAQOM feedback and survey after the class.

Thank You!

Truman Ng
91468768
truman.ng8@nexplore.co