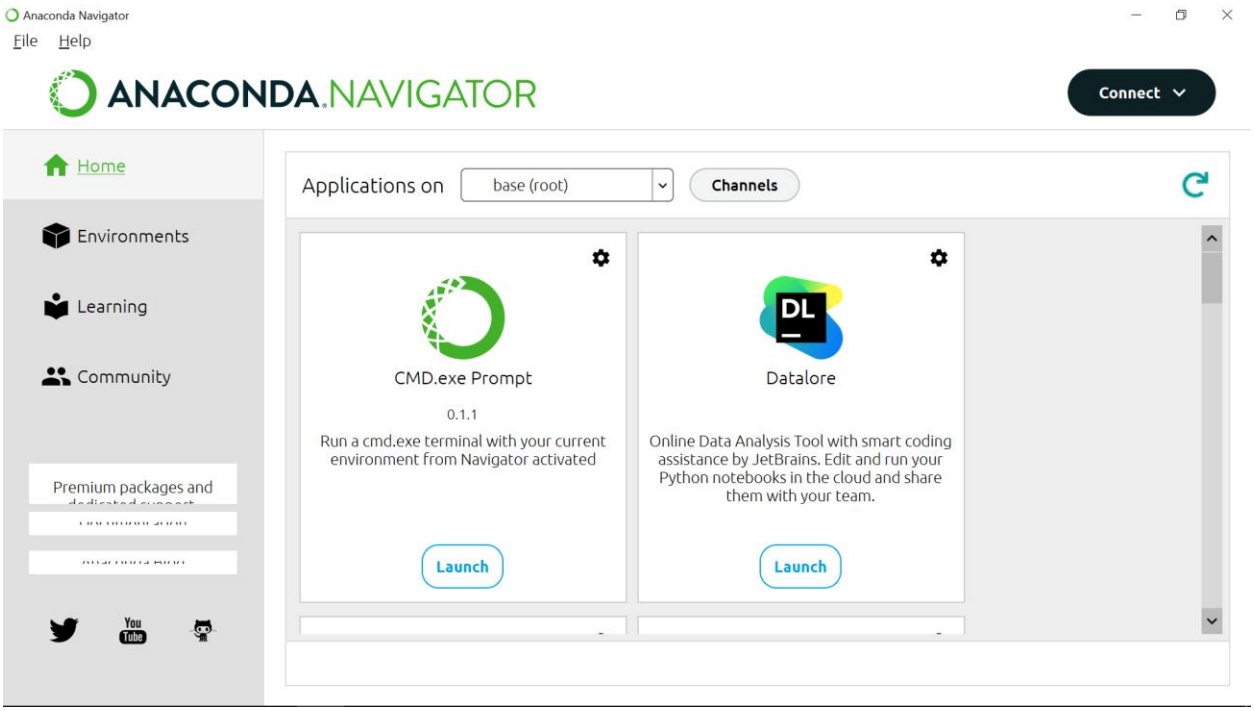
LAB 1

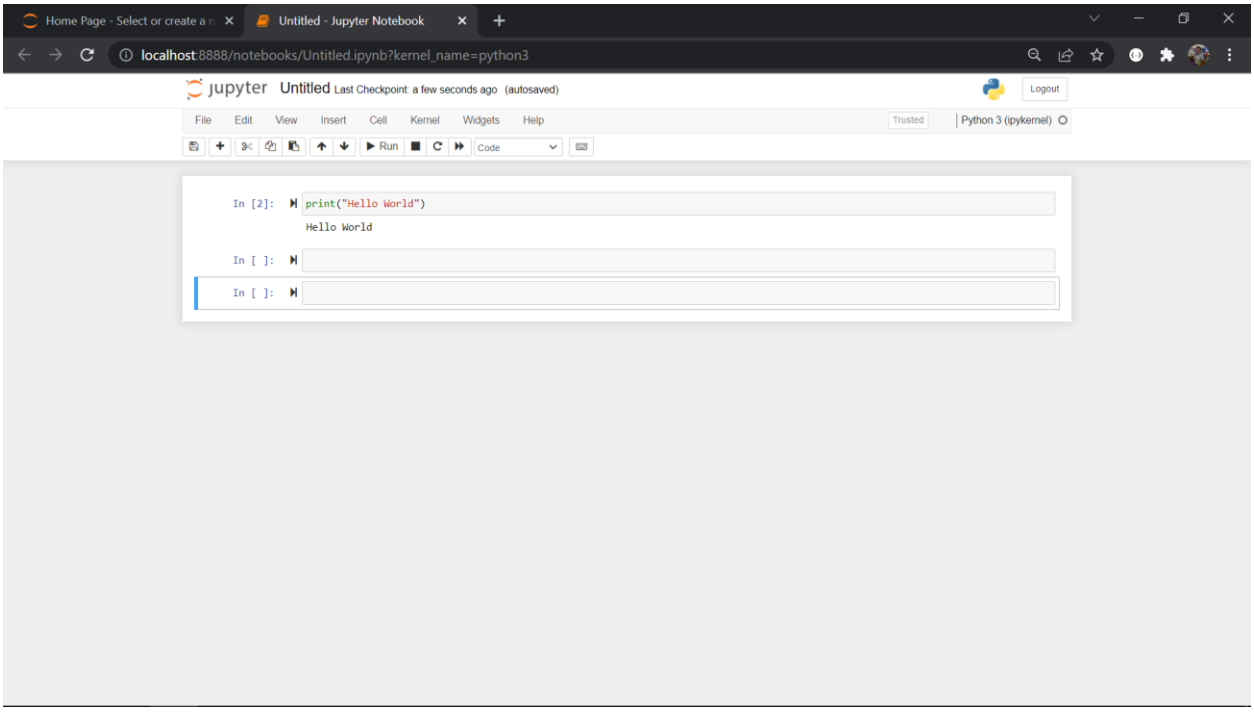
Student code: 18520726

Student name: Do Hoang Hiep

I. Environment set-up: Anaconda



1. Create a Jupyter notebook

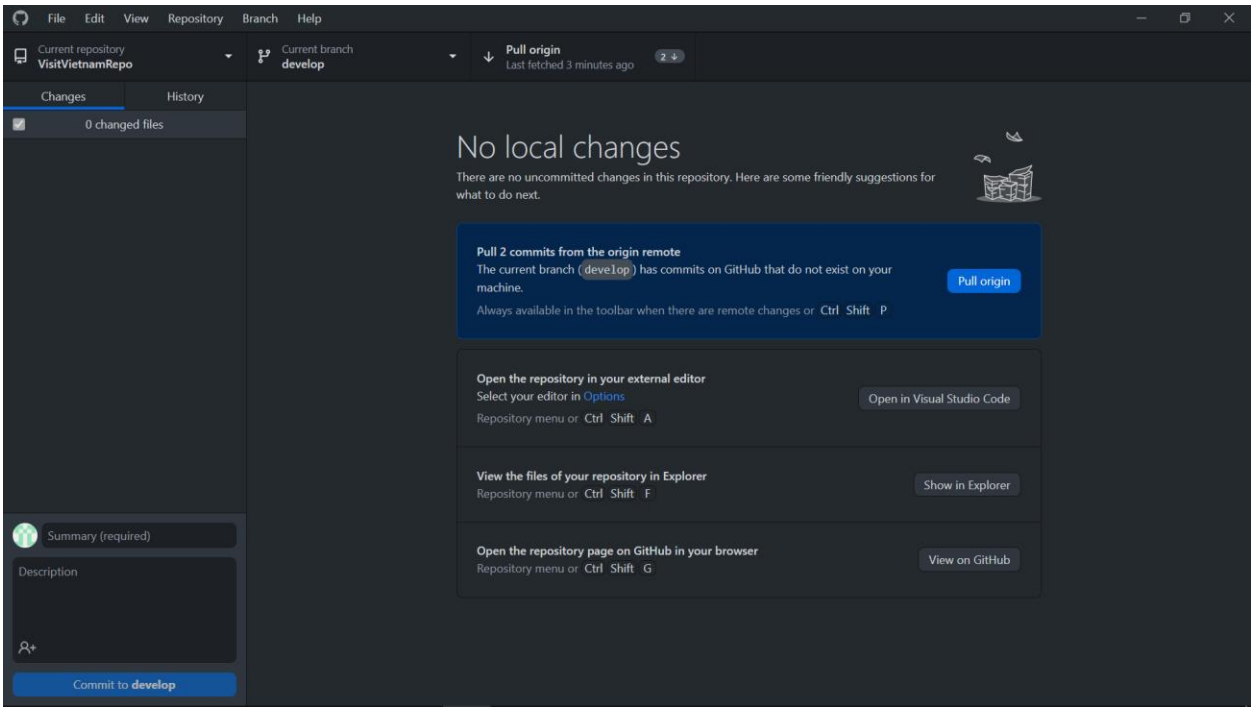


2. Export Jupyter notebook and run it as a Python script

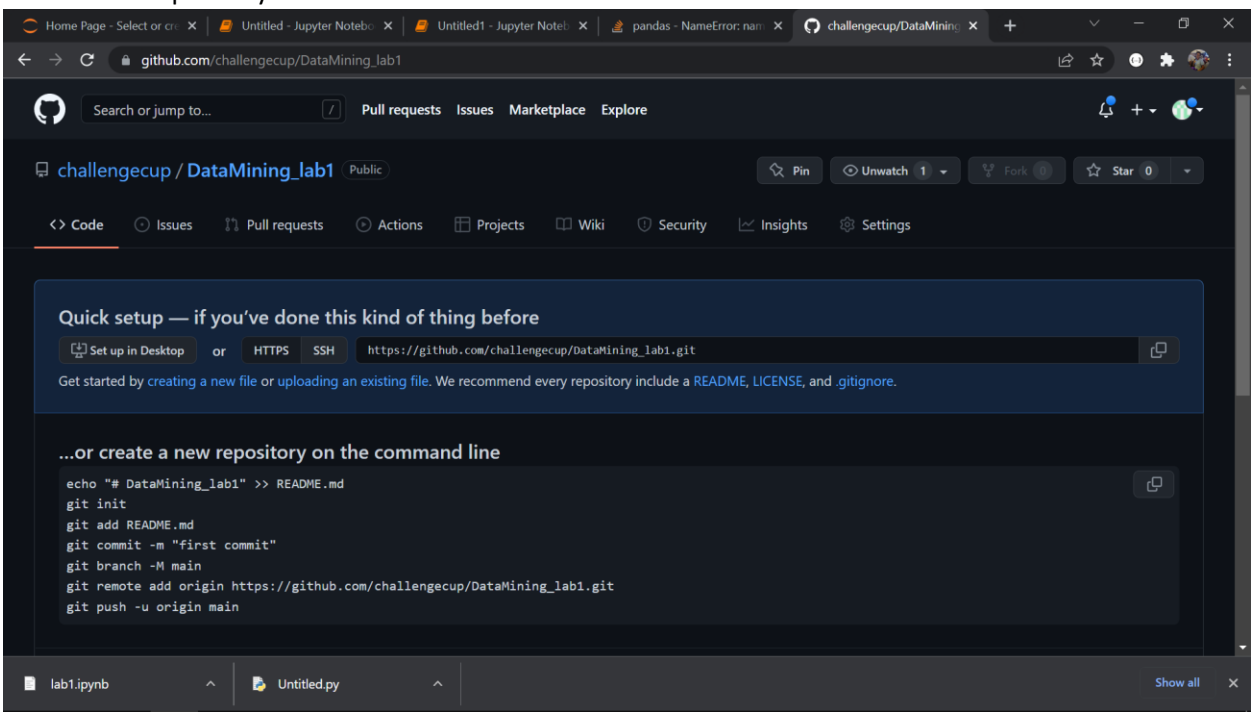
```
(base) PS C:\Users\hp\Downloads> jupyter nbconvert --to script lab1.ipynb
[NbConvertApp] Converting notebook lab1.ipynb to script
[NbConvertApp] Writing 72 bytes to lab1.py
(base) PS C:\Users\hp\Downloads> python lab1.py
Hello World
(base) PS C:\Users\hp\Downloads>
```

II. Github

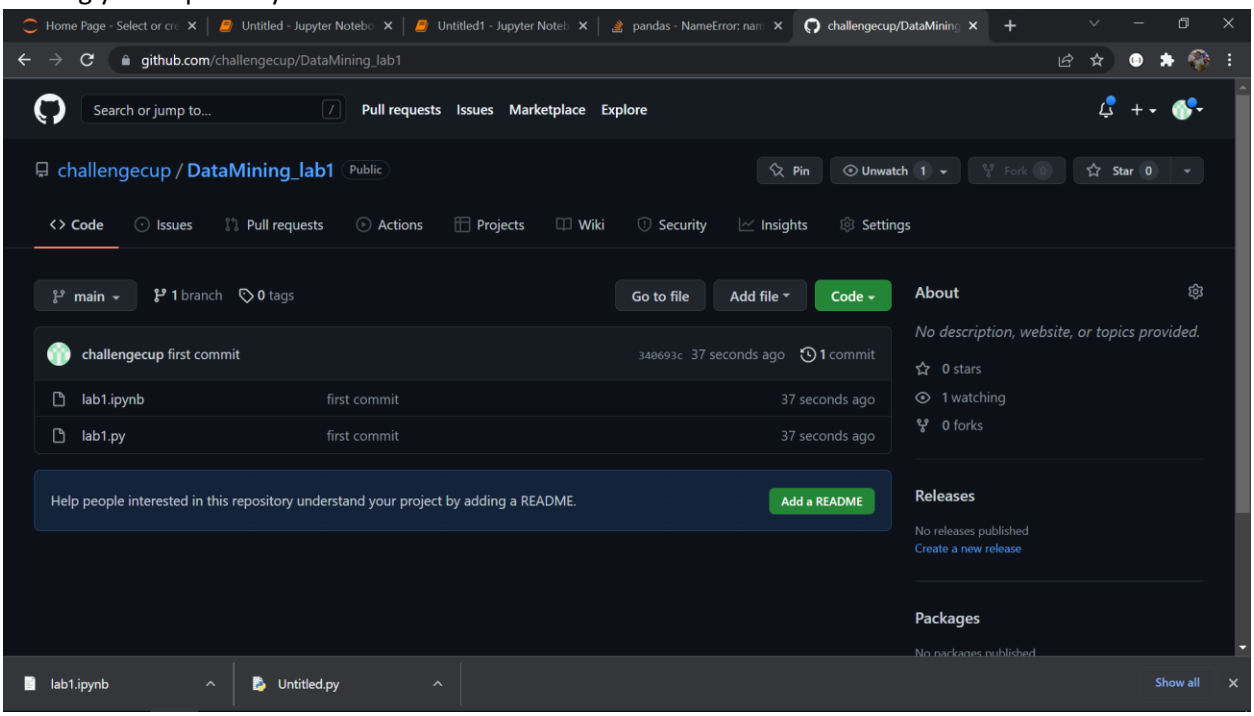
1. Install Github desktop and create a Github account


File Edit View Repository Branch Help
Current repository
VisitVietnamRepo
Current branch
develop
Pull origin
Last fetched 3 minutes ago
2
Changes
History
0 changed files
No local changes
There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.
Pull 2 commits from the origin remote
The current branch (develop) has commits on GitHub that do not exist on your machine.
Always available in the toolbar when there are remote changes or Ctrl Shift P
Pull origin
Open the repository in your external editor
Select your editor in Options
Repository menu or Ctrl Shift A
Open in Visual Studio Code
View the files of your repository in Explorer
Repository menu or Ctrl Shift F
Show in Explorer
Open the repository page on GitHub in your browser
Repository menu or Ctrl Shift G
View on GitHub
Summary (required)
Description
Commit to develop

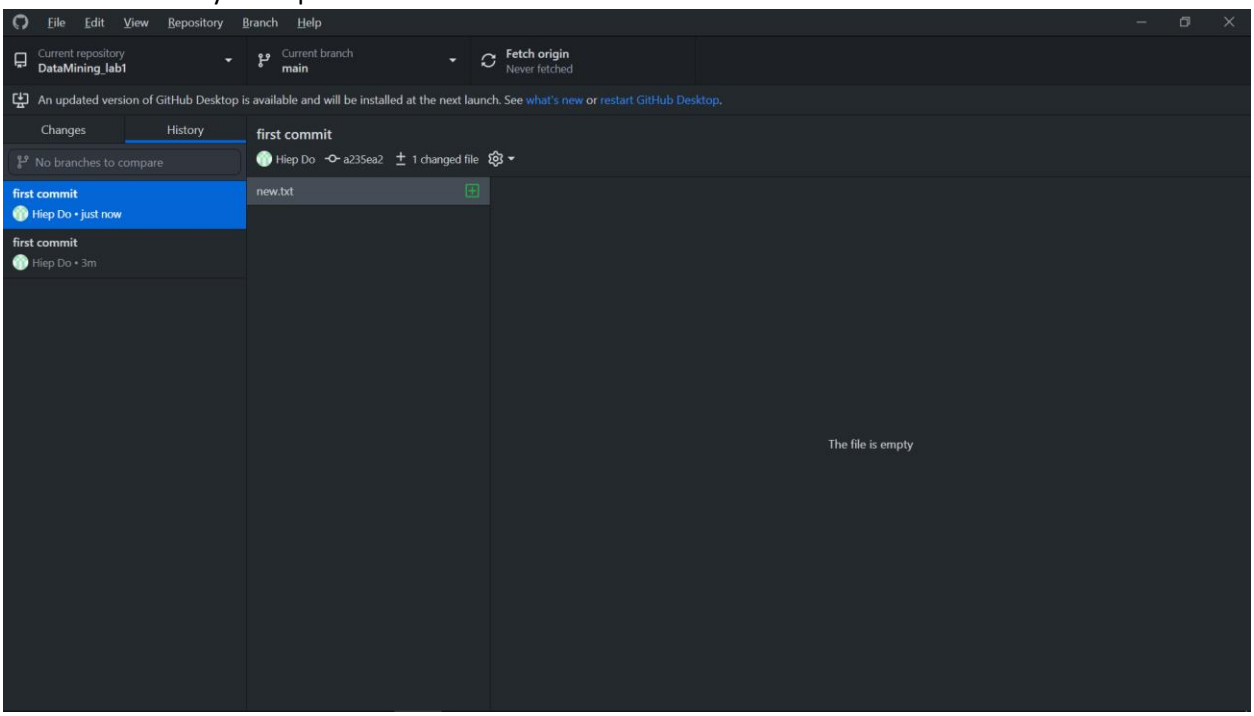
2. Create new repository



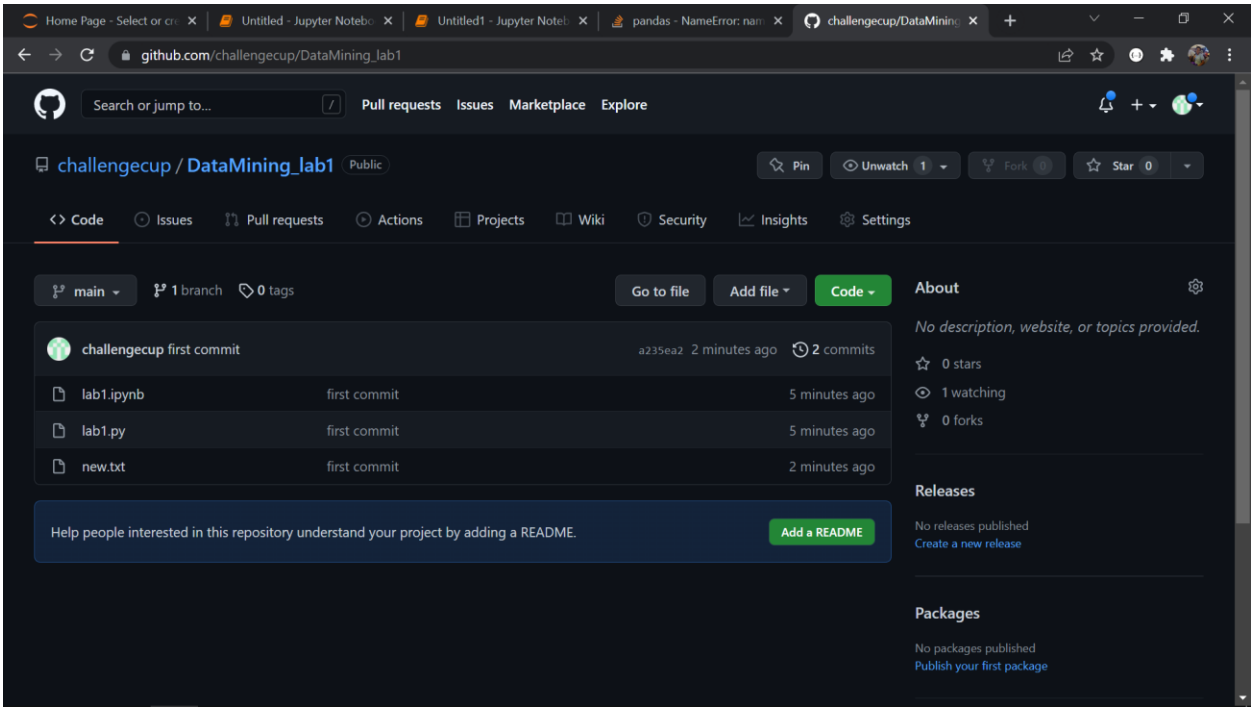
3. Cloning your repository



4. Add a new file to your repo



5. Push a branch to GitHub



III. Python basics

1. Data types
   a. Numbers

```
In [2]: 1 + 1
Out[2]: 2

In [3]: 1 * 3
Out[3]: 3

In [4]: 1 / 2
Out[4]: 0.5

In [5]: 2 ** 4
Out[5]: 16

In [6]: 4 % 2
Out[6]: 0

In [7]: 5 % 2
Out[7]: 1

In [9]: (2 + 3) * (5 + 5)
Out[9]: 50
```

   b. Variable assignment

```
In [10]: # Can not start with number of special characters
         name_of_var = 2

In [11]: x = 2
         y = 3

In [12]: z = x + y

In [13]: z
Out[13]: 5
```

   c. Strings

```
In [14]: 'single quotes'
Out[14]: 'single quotes'

In [16]: "double quotes"
Out[16]: 'double quotes'

In [17]: "wrap lot's of other quotes"
Out[17]: "wrap lot's of other quotes"
```

d. Printing

```
In [18]: x = 'hello'

In [19]: x
Out[19]: 'hello'

In [20]: print(x)
         hello

In [21]: num = 12
         name = 'Sam'

In [23]: print ('My number is: {one}, and my name is: {two}'.format(one=num, two=name))
         My number is: 12, and my name is: Sam

In [24]: print ('My number is: {}, and my name is: {}'.format(num,name))
         My number is: 12, and my name is: Sam
```

e. Lists

```
In [26]: [1,2,3]
Out[26]: [1, 2, 3]

In [27]: ['hi',1,[1,2]]
Out[27]: ['hi', 1, [1, 2]]

In [28]: my_list = ['a','b','c']

In [29]: my_list.append('d')

In [30]: my_list
Out[30]: ['a', 'b', 'c', 'd']

In [31]: my_list[0]
Out[31]: 'a'

In [32]: my_list[1]
Out[32]: 'b'

In [33]: my_list[1:]
Out[33]: ['b', 'c', 'd']

In [34]: my_list[:1]
Out[34]: ['a']

In [35]: my_list[0] = 'NEW'

In [36]: my_list
Out[36]: ['NEW', 'b', 'c', 'd']

In [37]: nest = [1,2,3,[4,5,['target']]]

In [38]: nest[3]
Out[38]: [4, 5, ['target']]

In [39]: nest[3][2]
Out[39]: ['target']

In [40]: nest[3][2][0]
Out[40]: 'target'
```

f. Dictionaries

```
In [41]: d = {'key1':'item1','key2':'item2'}

In [42]: d
Out[42]: {'key1': 'item1', 'key2': 'item2'}

In [43]: d['key1']
Out[43]: 'item1'
```

g. Booleans

```
In [44]: True
Out[44]: True

In [45]: False
Out[45]: False
```

h. Tuples

```
In [46]: t = (1,2,3)

In [47]: t[0]
Out[47]: 1

In [48]: t[0] = 'NEW'
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_10520/2140988817.py in <module>
----> 1 t[0] = 'NEW'

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

i. Sets

```
In [49]: {1,2,3}
Out[49]: {1, 2, 3}

In [50]: {1,2,3,1,2,1,2,3,3,3,3,2,2,2,1,1,2}
Out[50]: {1, 2, 3}
```

j. Comparison Operators

```
In [52]: 1 < 2
Out[52]: True

In [53]: 1 >= 1
Out[53]: True

In [54]: 1 <= 4
Out[54]: True

In [55]: 1 == 1
Out[55]: True

In [56]: 'hi' == 'bye'
Out[56]: False
```

k. Logic Operators

```
In [57]: (1 > 2) and (2 < 3)
Out[57]: False

In [58]: (1 > 2) or (2 < 3)
Out[58]: True

In [59]: (1 == 2) or (2 == 3) or (4 == 4)
Out[59]: True
```

l. If, elseif, else Statement

```
In [60]: if 1 < 2:
             print('Yep!')
Yep!

In [61]: if 1 < 2:
             print('yep!')
yep!

In [63]: if 1 < 2:
             print('first')
         else:
             print('last')
first

In [64]: if 1 > 2:
             print('first')
         else:
             print('last')
last

In [65]: if 1 == 2:
             print('first')
         elif 3 == 3:
             print('middle')
         else:
             print('last')
middle
```

m. For loops

```
In [66]: seq = [1,2,3,4,5]

In [67]: for item in seq:
             print (item)
1
2
3
4
5

In [68]: for item in seq:
             print('Yep')
Yep
Yep
Yep
Yep
Yep

In [69]: for jelly in seq:
             print(jelly+jelly)
2
4
6
8
10
```

n. While loops

In [70]:
```
i = 1
while i < 5:
    print('i is: {}'.format(i))
    i = i + 1
```

```
i is: 1
i is: 2
i is: 3
i is: 4
```

o. Range

In [71]:
```
range(5)
```

Out[71]:
```
range(0, 5)
```

In [72]:
```
for i in range(5):
    print(i)
```

```
0
1
2
3
4
```

In [73]:
```
list(range(5))
```

Out[73]:
```
[0, 1, 2, 3, 4]
```

p. List Comprehension

In [74]:
```
x = [1,2,3,4]
```

In [75]:
```
out = []
for item in x:
    out.append(item**2)
print(out)
```

```
[1, 4, 9, 16]
```

In [76]:
```
[item**2 for item in x]
```

Out[76]:
```
[1, 4, 9, 16]
```

q. Functions

In [77]:
```
def my_func(param1 = 'default'):
    """
    Docstring goes here
    """
    print(param1)
```

In [78]:
```
my_func
```

Out[78]:
```
<function __main__.my_func(param1='default')>
```

In [79]:
```
my_func()
```

```
default
```

In [80]:
```
my_func('new params')
```

```
new params
```

In [81]:
```
my_func(param1='new params')
```

```
new params
```

In [82]:
```
def square(x):
    return x**2
```

In [84]:
```
out = square(2)
```

In [85]:
```
print(out)
```

```
4
```

r. Lambda expression

```
In [86]:
def times2(var):
    return var*2
```

```
In [87]:
times2(2)
```

```
Out[87]: 4
```

```
In [89]:
lambda var: var*2
```

```
Out[89]: <function __main__.<lambda>(var)>
```

s. Map and filter

```
In [90]:
seq = [1,2,3,4,5]
```

```
In [92]:
map(times2,seq)
```

```
Out[92]: <map at 0x20e273e6c70>
```

```
In [93]:
list(map(times2,seq))
```

```
Out[93]: [2, 4, 6, 8, 10]
```

```
In [94]:
list(map(lambda var: var*2, seq))
```

```
Out[94]: [2, 4, 6, 8, 10]
```

```
In [95]:
filter(lambda item: item%2 == 0, seq)
```

```
Out[95]: <filter at 0x20e273f5400>
```

```
In [96]:
list(filter(lambda item: item%2 == 0, seq))
```

```
Out[96]: [2, 4]
```

t. Methods

```
In [97]:
st = 'hello my name is Sam'
```

```
In [98]:
st.lower()
```

```
Out[98]: 'hello my name is sam'
```

```
In [99]:
st.upper()
```

```
Out[99]: 'HELLO MY NAME IS SAM'
```

```
In [100]:
st.split()
```

```
Out[100]: ['hello', 'my', 'name', 'is', 'Sam']
```

```
In [101]:
tweet = 'Go Sports! #Sports'
```

```
In [102]:
tweet.split('#')
```

```
Out[102]: ['Go Sports! ', 'Sports']
```

```
In [103]:
tweet.split('#')[1]
```

```
Out[103]: 'Sports'
```

```
In [104]:
d
```

```
Out[104]: {'key1': 'item1', 'key2': 'item2'}
```

```
In [105]:
d.keys()
```

```
Out[105]: dict_keys(['key1', 'key2'])
```

```
In [106]: d.items()
Out[106]: dict_items([('key1', 'item1'), ('key2', 'item2')])

In [107]: lst = [1,2,3]

In [108]: lst.pop()
Out[108]: 3

In [109]: lst
Out[109]: [1, 2]

In [110]: 'x' in [1,2,3]
Out[110]: False

In [111]: 'x' in ['x','y','z']
Out[111]: True
```

IV. Exercises

Q1:

```
In [112]: 7 ** 4
Out[112]: 2401
```

Q2:

```
In [113]: s = 'Hi there Sam!'

In [114]: s.split()
Out[114]: ['Hi', 'there', 'Sam!']
```

Q3:

```
In [115]: planet = 'Earth'
          diameter = 12742

In [116]: s = "The diameter of {0} is {1} kilometers".format(planet,diameter)
          print(s)
          The diameter of Earth is 12742 kilometers
```

Q4:

```
In [118]: lst = [1,2,[3,4],[5,[100,200,['hello']],23,11],1,7]

In [120]: lst[3][1][2][0]
Out[120]: 'hello'
```

Q5:

```
In [121]: d = {'k1':[1,2,3,{'tricky':['oh','man','inception',{'target':[1,2,3,'hello']}]}]}

In [122]: d['k1'][3]['tricky'][3]['target'][3]
Out[122]: 'hello'
```

Q6:

```
In [123]: # Tuple is immutable

In [124]: def domainGet(email):
              return email.split('@')[1]

In [125]: domainGet('user@domain.com')
Out[125]: 'domain.com'
```

Q7:

```
In [126]: def findDog(str):
              if 'dog' in str.lower():
                  return True
              return False

In [127]: findDog('Is there a dog here?')
Out[127]: True
```

Q8:

```
In [128]: def countDog(str):
              count = 0
              for element in str.lower().split():
                  if element == 'dog':
                      count += 1
              return count

In [129]: countDog('This dog runs faster than the other dog dude!')
Out[129]: 2
```

Q9:

```
In [130]: seq = ['soup','dog','salad','cat','greate']

In [131]: list(filter(lambda element: element[0] == 's', seq))
Out[131]: ['soup', 'salad']
```

Q10:

```
In [132]: def caught_speeding(speed,is_birthday):
              status=''
              if is_birthday == False:
                  if speed <= 60:
                      status = 'No ticket'
                  elif speed > 60 and speed <= 80:
                      status = 'Small ticket'
                  else:
                      status = 'Big ticket'
              else:
                  if speed <= 65:
                      status = 'No ticket'
                  elif speed > 65 and speed <= 85:
                      status = 'Small ticket'
                  else:
                      status = 'Big ticket'
              return status

In [133]: caught_speeding(81, True)
Out[133]: 'Small ticket'

In [134]: caught_speeding(81, False)
Out[134]: 'Big ticket'
```

I uploaded file Python on Github: https://github.com/challengecup/DataMining_lab1