Department of Statistics, The Chinese University of Hong Kong
STAT 5106, Programming Techniques for Data Science (Term 1, 2023-24)

# ASSIGNMENT 1

(Deadline: 21 Sept 2023, 2359)

*[Note: Although the assignment is about Python script, we still welcome to use R script for handling those non-Datacamp questions. But please make sure the scripts are runnable and give the correct answers.]*

Q1. 50% , Please attempt one of the following:

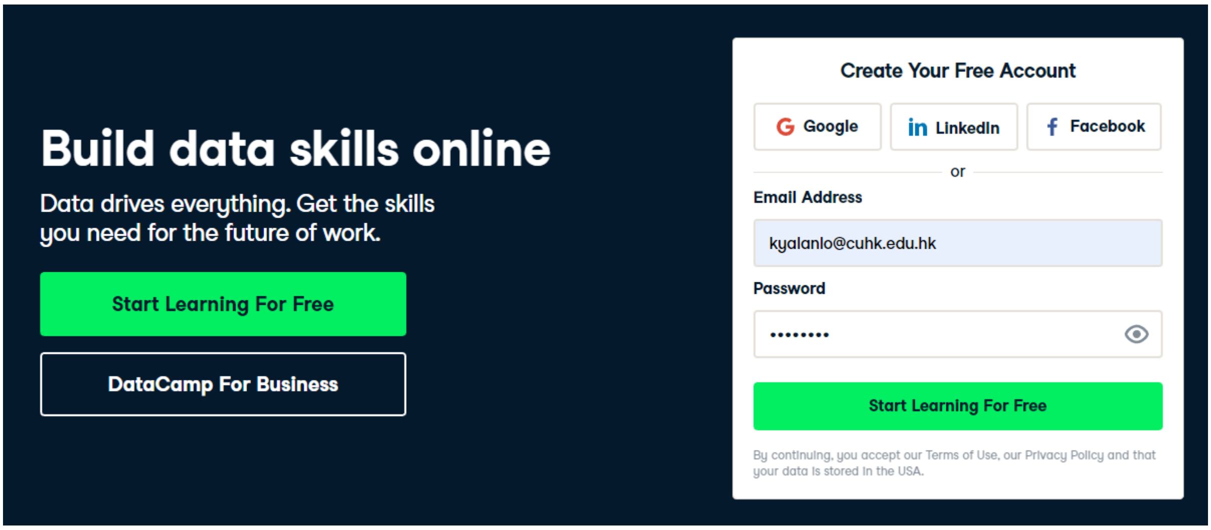
Option A. Time expectation for 5 hours.

Please complete all the following "Assignment 1" chapters in Python Fundamentals Skill Track in Datacamp.
Note for the teaching schedule I still follow to Dr. Chuck (py4e) . But if you need extra examples and training this skill track can help.
Although this is not totally covered what we learn for assignments 1 and 2, the whole skill track is still recommended to all guys completing.

| Course | Chapter | Should be finished in |
|---|---|---|
| Introduction to Python | Python Basics | Assignment 1 |
| Introduction to Python | Python Lists | Assignment 2 |
| Introduction to Python | Functions and Packages | Assignment 1 |
| Introduction to Python | Numpy | Assignment 2 |
| Intermediate Python | Matplotlib | Do this if you want |
| Intermediate Python | Dictionaries and Pandas | Assignment 2 |
| Intermediate Python | Logic, Control Flow and Filtering | Assignment 1 |
| Intermediate Python | Loops | Assignment 1: While Loop, For Loop<br>Assignment 2: Loop Data Structure Part 1/2 |
| Python Data Science Toolbox (Part 1) | Writing your own functions | Assignment 1 |
| Python Data Science Toolbox (Part 1) | Default arguments, variable-length arguments and scope | Do this if you want |
| Python Data Science Toolbox (Part 1) | Lambda functions and error-handling | Assignment 1: Introduction to error-handling<br>Assignment 2: the others |
| Python Data Science Toolbox (Part 2) | Using iterators in PythonLand | Do this if you want |
| Python Data Science Toolbox (Part 2) | List comprehensions and generators | Assignment 2 |

# To access into the datacamp classroom, please:

1. Access into www.datacamp.com
2. Register with your cuhk email account - the one ending with cuhk.edu.hk .
   (Note: don’t use your personal account - since the classroom won’t accept any non cuhk.edu.hk accounts)



## Build data skills online

Data drives everything. Get the skills you need for the future of work.

### Create Your Free Account

Email Address

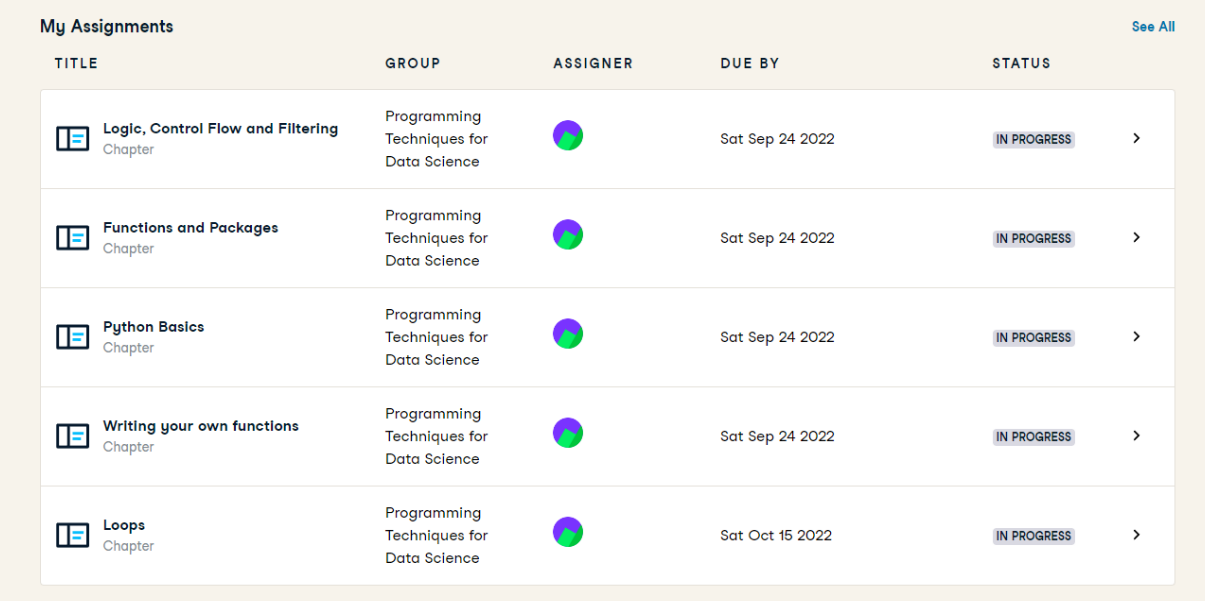
kyalanlo@cuhk.edu.hk

Password

••••••••

By continuing, you accept our Terms of Use, our Privacy Policy and that your data is stored in the USA.

3. After registration, check the following URL out: https://www.datacamp.com/groups/shared_links/ff5dc17433101f5056ecd0215fffe219ae959697357a008a9bd5850129cdef64
4. You can see assignments assigned to you.



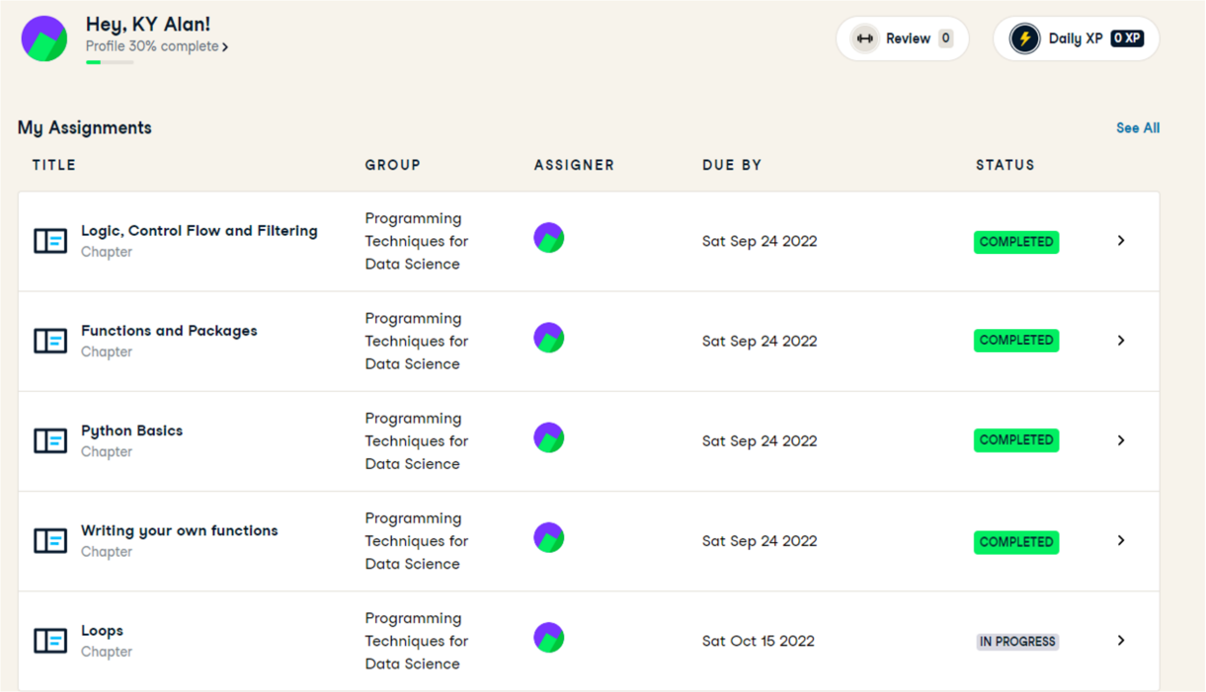
My Assignments

See All

| TITLE | GROUP | ASSIGNER | DUE BY | STATUS | |
|---|---|---|---|---|---|
| Logic, Control Flow and Filtering<br>Chapter | Programming Techniques for Data Science | ● | Sat Sep 24 2022 | IN PROGRESS | › |
| Functions and Packages<br>Chapter | Programming Techniques for Data Science | ● | Sat Sep 24 2022 | IN PROGRESS | › |
| Python Basics<br>Chapter | Programming Techniques for Data Science | ● | Sat Sep 24 2022 | IN PROGRESS | › |
| Writing your own functions<br>Chapter | Programming Techniques for Data Science | ● | Sat Sep 24 2022 | IN PROGRESS | › |
| Loops<br>Chapter | Programming Techniques for Data Science | ● | Sat Oct 15 2022 | IN PROGRESS | › |

Please submit your cap screens for completion proof as the following:
(No need your name, we will check your email registered.)

1.



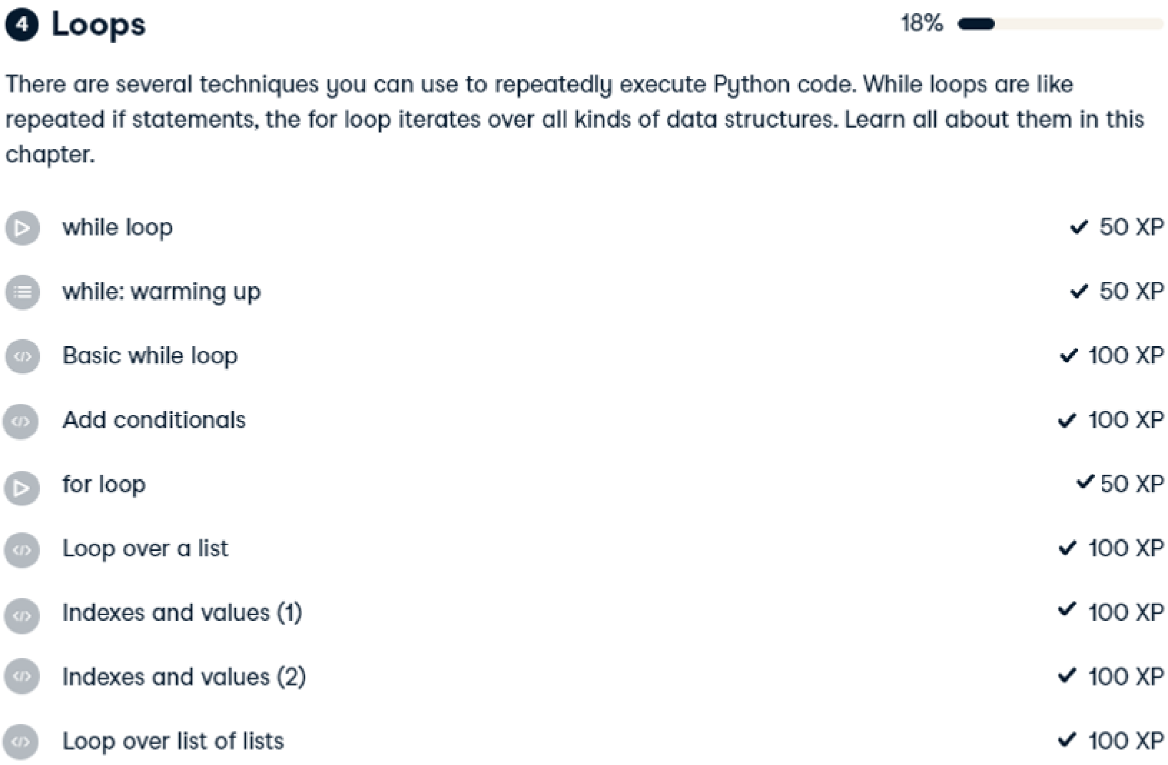
Hey, KY Alan!

Profile 30% complete ›

My Assignments

| TITLE | GROUP | ASSIGNER | DUE BY | STATUS |
|---|---|---|---|---|
| Logic, Control Flow and Filtering<br>Chapter | Programming Techniques for Data Science | | Sat Sep 24 2022 | COMPLETED |
| Functions and Packages<br>Chapter | Programming Techniques for Data Science | | Sat Sep 24 2022 | COMPLETED |
| Python Basics<br>Chapter | Programming Techniques for Data Science | | Sat Sep 24 2022 | COMPLETED |
| Writing your own functions<br>Chapter | Programming Techniques for Data Science | | Sat Sep 24 2022 | COMPLETED |
| Loops<br>Chapter | Programming Techniques for Data Science | | Sat Oct 15 2022 | IN PROGRESS |

2.



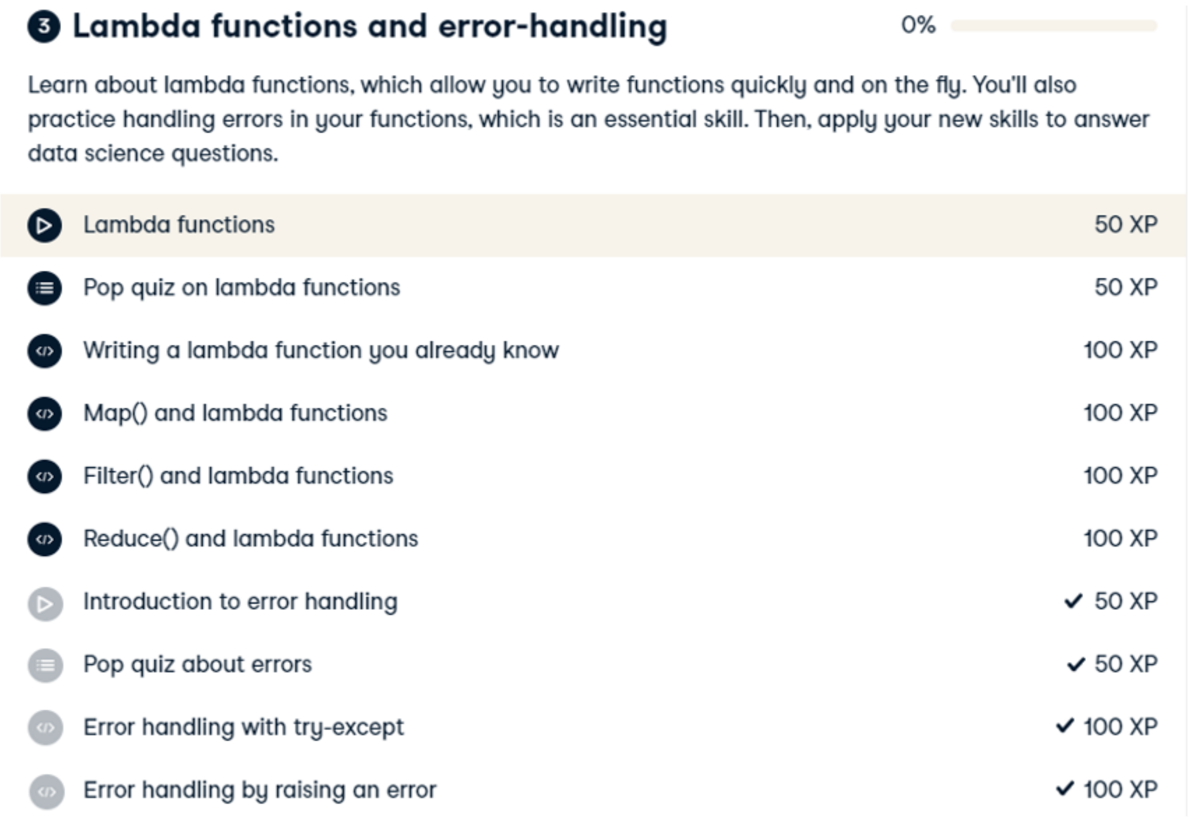
## 4 Loops

18%

There are several techniques you can use to repeatedly execute Python code. While loops are like repeated if statements, the for loop iterates over all kinds of data structures. Learn all about them in this chapter.

| | |
|---|---|
| while loop | ✓ 50 XP |
| while: warming up | ✓ 50 XP |
| Basic while loop | ✓ 100 XP |
| Add conditionals | ✓ 100 XP |
| for loop | ✓ 50 XP |
| Loop over a list | ✓ 100 XP |
| Indexes and values (1) | ✓ 100 XP |
| Indexes and values (2) | ✓ 100 XP |
| Loop over list of lists | ✓ 100 XP |

3.



## ❸ Lambda functions and error-handling

0%

Learn about lambda functions, which allow you to write functions quickly and on the fly. You'll also practice handling errors in your functions, which is an essential skill. Then, apply your new skills to answer data science questions.

| | |
|---|---|
| Lambda functions | 50 XP |
| Pop quiz on lambda functions | 50 XP |
| Writing a lambda function you already know | 100 XP |
| Map() and lambda functions | 100 XP |
| Filter() and lambda functions | 100 XP |
| Reduce() and lambda functions | 100 XP |
| Introduction to error handling | ✓ 50 XP |
| Pop quiz about errors | ✓ 50 XP |
| Error handling with try-except | ✓ 100 XP |
| Error handling by raising an error | ✓ 100 XP |

Option B. Time expectation is 1 hour - given you are already familiar with Python programming, or have already started the career of Data Scientist :)

[Validating Bank Account Numbers - Luhn Algorithm]

You and Hanzawa Naoki are good friends. Yesterday, Mr. Hanzawa received bank accounts from Tokyo Chuo Bank. He wants to verify whether his bank accounts numbers are valid or not. You happen to be great at python so he is asking for your help!

Write a Python function named `validate_credit_card` that takes a credit card number as input and returns True if the number is valid according to the Luhn algorithm, and False otherwise.

The Luhn algorithm is a simple checksum formula used to validate a variety of identification numbers, including credit card numbers. It works as follows:

- Starting from the rightmost digit (excluding the check digit), double the value of every second digit. If the result of doubling exceeds 9, subtract 9 from the product.
- Sum all the digits obtained from the previous step together with the unaffected digits (digits that were not doubled).
- If the total sum is divisible by 10 (i.e., the modulo 10 of the sum equals zero), then the credit card number is considered valid.

For example, given the credit card number 45320151128336, the function should return True because the sum of the digits (4+5+6+2+0+1+1+1+2+8+3+3+6) equals 42, which is divisible by 10.

Write the validate_credit_card function and test it with three credit card numbers: 45320151128336, 6011123456789000, and 379354508162306. Print the result for each credit card number, indicating whether it is valid or not.

Note: Ignore any non-digit characters (e.g., spaces, dashes) in the credit card number and consider it as a string of digits.

You can assume that the credit card numbers provided will have a maximum length of 19 digits.

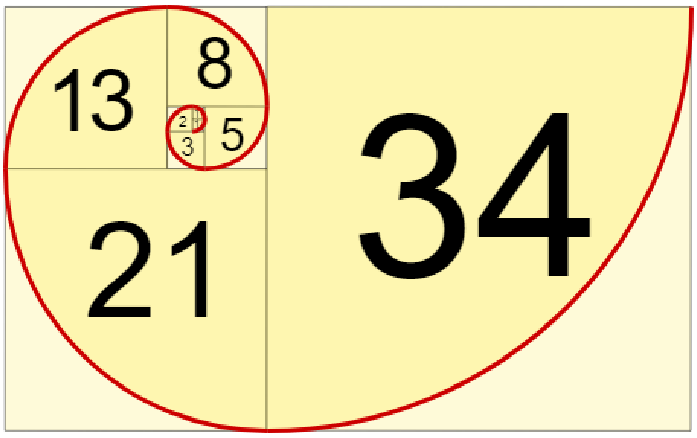
Your function should have the following signature:

```
def validate_credit_card(credit_card_number: str) -> bool:
    # Your code here
```

Example output:

```
45320151128336: Valid
6011123456789000: Invalid
379354508162306: Valid
```

Q2. 25%, Checking mathematical series number



Fibonacci Sequence

Write a Python function named check_mathematical_series that takes a positive integer n as input and checks whether the given number belongs to any of the following important mathematical series: Fibonacci sequence, Triangular sequence, Square sequence.

The function should determine the series type and return a string indicating the result. The possible return values are:

- "Fibonacci" if the number belongs to the Fibonacci sequence.
- "Triangular" if the number belongs to the Triangular sequence.
- "Square" if the number belongs to the Square sequence.
- "None" if the number does not belong to any of the above sequences.

For this question, consider the following definitions for the series:

1. Fibonacci sequence: A series of numbers in which each number (after the first two) is the sum of the two preceding ones. The sequence starts with 0 and 1. For example, the Fibonacci sequence begins as follows: 0, 1, 1, 2, 3, 5, 8, 13, 21, ...
2. Triangular sequence: A sequence of numbers in which each term represents the total number of dots required to form a triangle with that many dots on each side. The nth term of the triangular sequence is given by the formula (n * (n + 1)) / 2. For example, the triangular sequence begins as follows: 1, 3, 6, 10, 15, ...
3. Square number sequence: A sequence of numbers that are the squares of consecutive integers. For example, the square number sequence begins as follows: 1, 4, 9, 16, 25, 36, ...

Write the `check_mathematical_series` function and test it with the following values of n:

- 21
- 9
- 17
- 10

Print the result for each value of n, indicating the series types to which it belongs.
Using commas (,) if it belongs to more than 1 series.
"None" if it doesn't belong to any of the series.

Your function should have the following signature:

```
def check_mathematical_series(n: int) -> str:
    # Your code here
```

Example output:

```
n = 21: Fibonacci, Triangular
n = 9: Square
n = 17: None
n = 10: Triangular
```

Note: You can assume that the given n will be a positive integer.

## Q3. 25%, 超級無敵開口中 aka Guess a number game

Suppose to be trivial - guessing a number between a random number between 1 to N, whereas N is pre-set.
If the number is wrongly guessed, the game will return a message with updated range.
Of course you will win(??? cream as present ?) with special message when you guess correctly.

Please write a program with the following:

- with inputting a maximum integer guessed in the game.
- please refer to the package random for drawing a number.
- try-error exception if input wrongly

The output could be

```
Set a maximum number of this game: 100
Guess a number between 1 and 100 until you get it right : 95
Balloon is still being expanded…
Guess a number between 95 and 100 until you get it right : Don’t want to guess
Please input a NUMBER between 95 and 100 !!!
Guess a number between 95 and 100 until you get it right : 97
BOOM !!!
Congrats ! You have ended the game !
```