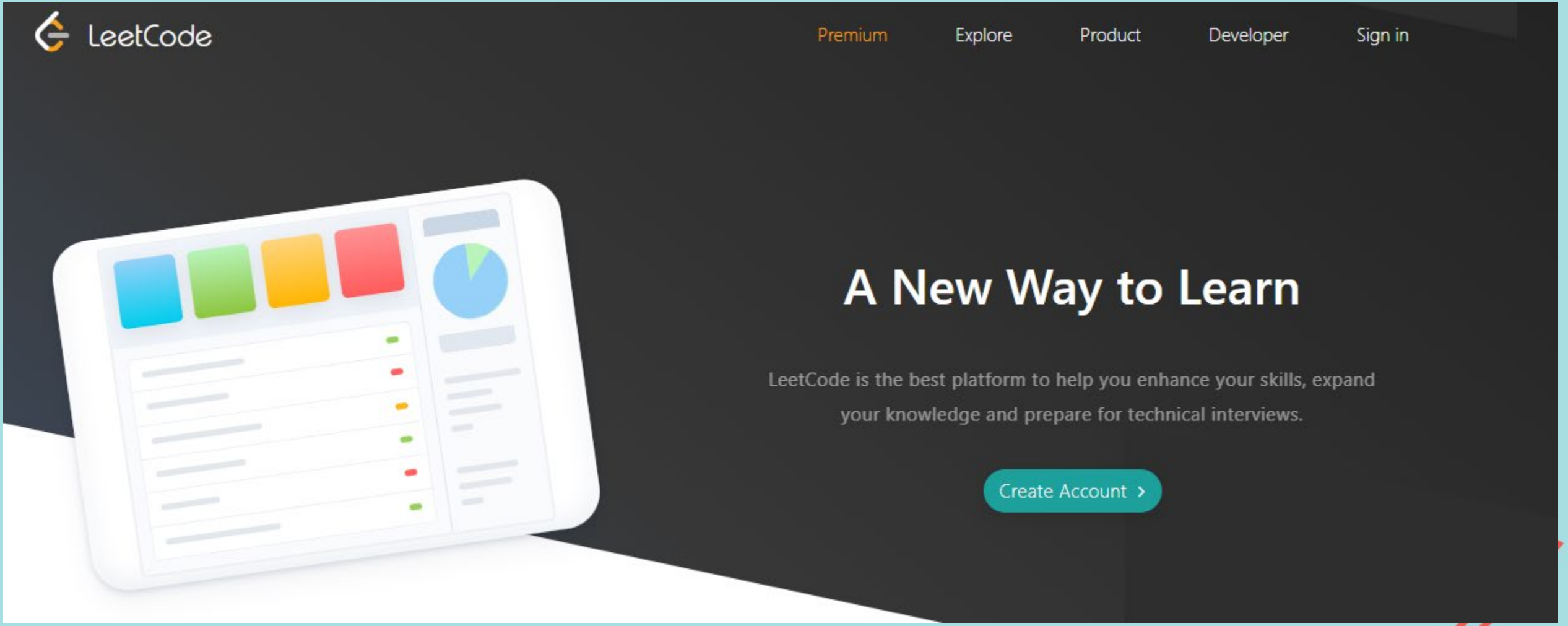
2023-Spring
Advanced Computer Programming (6)

CSIE, Asia Univ.

Course schedule

- W1-Introduction
- W2-Python libraries
- W3-BeautifulSoup(1)
- W4-BeautifulSoup(2)
- W5-
- W6-Scrapy(2)
- W7-Scrapy
- W8-Project development(1)
- W9-Midterm presentation
- W10-Web & HTTP
- W11-Flask
- W12-Flask Routes
- W13-Jinja template
- W14-Flask-form
- W15-Flask-mail
- W16-REST API
- W17-Project development(2)
- W18-Final presentation

leetcode



https://leetcode.com/

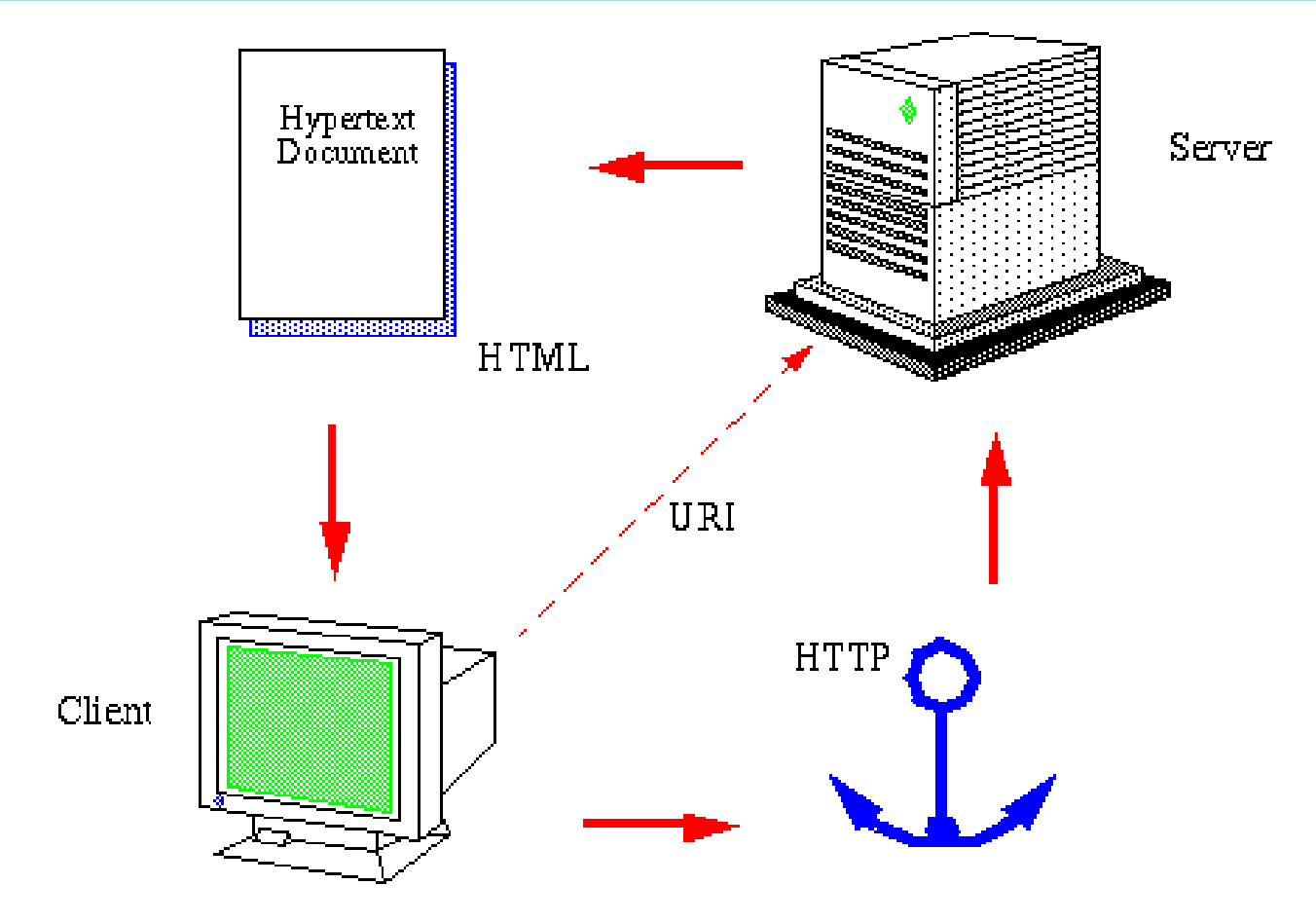
Leetcode problem: **3Sum**

- Given an array of integers nums return all the triplets [nums[i], nums[j], nums[k]] such that i != j, i != k, and j != k, and nums[i] + nums[j] + nums[k] == 0.
- Notice that the solution set must not contain duplicate triplets.
- https://leetcode.com/problems/3sum/

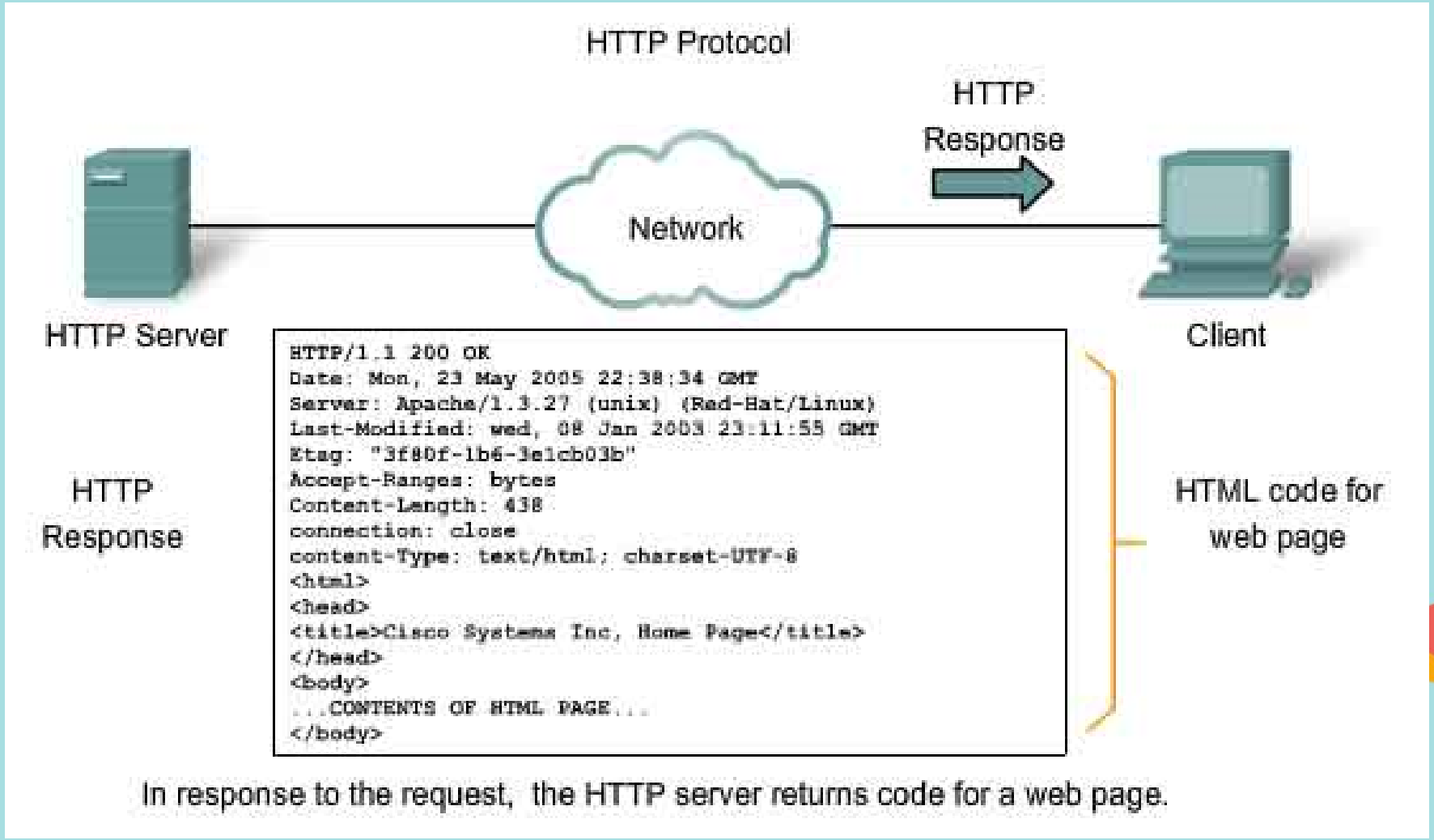
HTTP, URI, HTML

```
<!DOCTYPE html>
<html>
<!-- created 2010-01-01 -->
 <head>
  <title>sample</title>
 </head>
 <body>
  <p>Voluptatem accusantium
   totam rem aperiam.</p>
 </body>
</html>
```

HTML



HTTP


HTTP Protocol
HTTP
Response
Network
HTTP Server
Client
HTTP
Response
HTTP/1.1 200 OK
Date: Mon, 23 May 2005 22:38:34 GMT
Server: Apache/1.3.27 (unix) (Red-Hat/Linux)
Last-Modified: wed, 08 Jan 2003 23:11:55 GMT
Etag: "3f80f-1b6-3e1cb03b"
Accept-Ranges: bytes
Content-Length: 438
connection: close
content-Type: text/html; charset-UTF-8
<html>
<head>
<title>Cisco Systems Inc, Home Page</title>
</head>
<body>
...CONTENTS OF HTML PAGE...
</body>
HTML code for
web page
In response to the request, the HTTP server returns code for a web page.

robots.txt

- A robots.txt file is a simple text file containing rules about which crawlers may access which parts of a site. For example, the robots.txt file for example.com may look like this:

```
# This robots.txt file controls crawling of URLs under https://example.com.
# All crawlers are disallowed to crawl files in the "includes" directory, such
# as .css, .js, but Googlebot needs them for rendering, so Googlebot is allowed
# to crawl them.
User-agent: *
Disallow: /includes/

User-agent: Googlebot
Allow: /includes/

Sitemap: https://example.com/sitemap.xml
```

https://developers.google.com/search/docs/advanced/robots/robots_txt

Beautiful Soup

Beautiful Soup 4.9.0 documentation » Beautiful Soup Documentation

Table of Contents

Beautiful Soup Documentation

Beautiful Soup is a Python library for pulling data out of HTML and XML files. It works with your favorite parser to provide idiomatic ways of navigating, searching, and modifying the parse tree. It commonly saves programmers hours or days of work.

These instructions illustrate all major features of Beautiful Soup 4, with examples. I show you what the library is good for, how it works, how to use it, how to make it do what you want, and what to do when it violates your expectations.

This document covers Beautiful Soup version 4.11.0. The examples in this documentation were written for Python 3.8.

You might be looking for the documentation for Beautiful Soup 3. If so, you should know that Beautiful Soup 3 is no longer being developed and that all support for it was dropped on December 31, 2020. If you want to learn about the differences between Beautiful Soup 3 and Beautiful Soup 4, see Porting code to BS4.

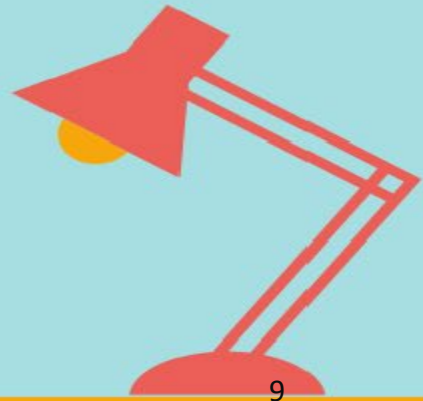
This documentation has been translated into other languages by Beautiful Soup users:

- 这篇文档当然还有中文版.
- このページは日本語で利用できます(外部リンク)
- 이 문서는 한국어 번역도 가능합니다.
- Este documento também está disponível em Português do Brasil.
- Эта документация доступна на русском языке.

https://www.crummy.com/software/BeautifulSoup/bs4/doc/

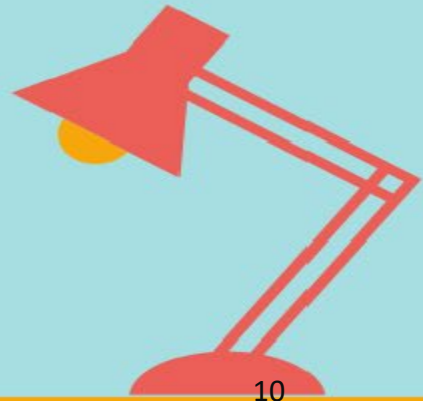
Assignment 1

- Overview:
  - In this assignment, we will use the basic python web scraping tools urllib and BeautifulSoup to scrape data from the focustaiwan website. Please list the scraped news content and submit it to Tronclass's assignment entry.
- Objectives:
  - Learn how to obtain the content of a web page using web scraping.
  - To explore real html files.
  - Reflect on the possible uses of web scraping capabilities for data science.
- Instructions:
  - Go to the focustaiwan website using any browser. https://focustaiwan.tw/
  - Check the tags in the html content.



Key components

- urllib.request.urlopen()
- BeautifulSoup.findAll()



Assignment 2

- Overview:
  - In this assignment, we will use BeautifulSoup to grab all the news content from the cna website and save it to a txtd file to submit to Tronclass's job entry.
- Target:
  - Learn how to get links to web pages using web scraping.
  - Explore crawling an entire site.
  - Reflect on possible uses of web scraping in data science.
- instruct:
  - Visit the focustaiwan website using any browser. https://focustaiwan.tw/
  - Check tags in html content.

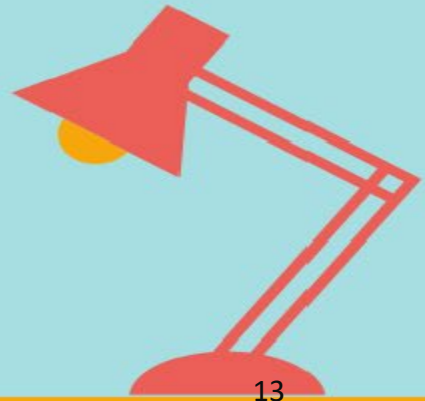
Midterm Report

- Crawling the webpage of the Department of CSIE
- https://csie.asia.edu.tw/en/associate_professors_2
- Read each teacher's specialty / Discipline expertise
- Save as a text file, which can be in txt, csv or json format
- The program part can be submitted as a notebook or py file

robots.txt

- A robots.txt file is a simple text file containing rules about which crawlers may access which parts of a site. For example, the robots.txt file for example.com may look like this:

```
# This robots.txt file controls crawling of URLs under https://example.com.
# All crawlers are disallowed to crawl files in the "includes" directory, such
# as .css, .js, but Googlebot needs them for rendering, so Googlebot is allowed
# to crawl them.
User-agent: *
Disallow: /includes/

User-agent: Googlebot
Allow: /includes/

Sitemap: https://example.com/sitemap.xml
```

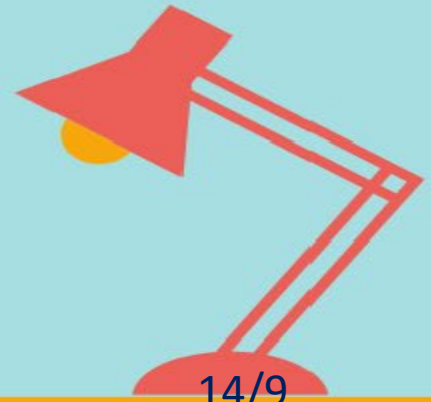


https://developers.google.com/search/docs/advanced/robots/robots_txt

Regular Expressions in Python

Regular expressions (Regex) is a method used to process strings. Regex uses its own set of special notation, which allows us to easily search for strings, replace strings, delete strings or test whether strings conform to the style. rule.
Regex is not a programming language, it is just a "notation" of "string style rules", which is used to express the rules for the appearance of character symbols in strings. Most programming languages support the usage of Regex, and any As long as the tool supports this notation, you can use Regex to process strings on the tool.

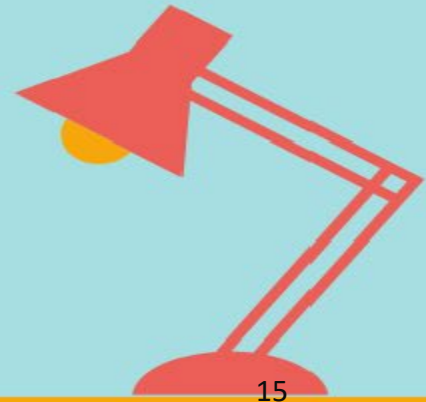
Regex from	Explanation
\d{4}-\d{2}-\d{2}	Find date string in YYYY-MM-DD format from text
cat|dog	find cat or dog string from text
[A-Z]\w+	Find all English words that start with capital letters from the text
^[A-Za-z]\d{9}$	Verify whether the string is a Taiwan ID number



https://www.dataquest.io/blog/regex-cheatsheet/

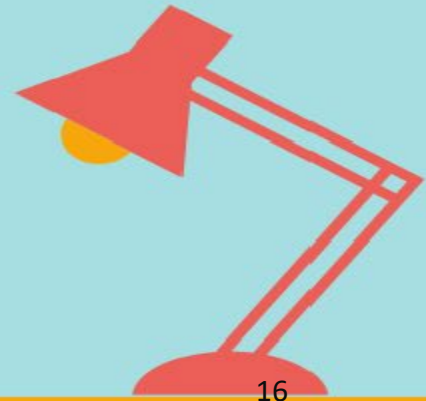
Anchors — ^ and $

- ^The matches any string that starts with The -> Try it!
- end$ matches a string that ends with end
- ^The end$ exact string match (starts and ends with The end)



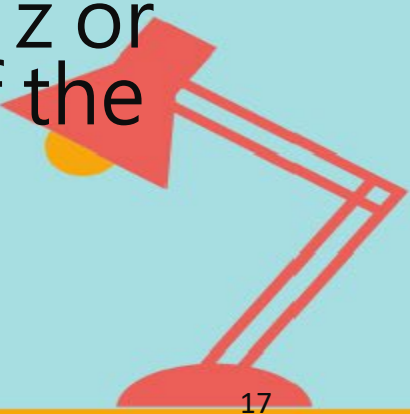
Grouping and capturing

- a(bc) parentheses create a capturing group with value bc
- a(?:bc)* using ?: we disable the capturing group
- a(?<foo>bc) using ?<foo> we put a name to the group



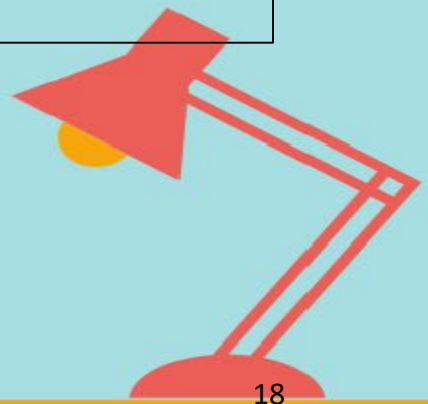
Bracket expressions — []

- [abc] matches a string that has either an a or a b or a c -> is the same as a|b|c
- [a-c] same as previous
- [a-fA-F0-9] a string that represents a single hexadecimal digit, case insensitively
- [0-9]% a string that has a character from 0 to 9 before a % sign
- [^a-zA-Z] a string that has not a letter from a to z or from A to Z. In this case the ^ is used as negation of the expression



Regular Expressions and BeautifulSoup

```
from urllib.request import urlopen
from bs4 import BeautifulSoup
import re

html = urlopen('http://www.pythonscraping.com/pages/page3.html')
bs = BeautifulSoup(html, 'html.parser')
images = bs.find_all('img', {'src':re.compile('\.\.\/img\/gifts/img.*\.jpg')})
for image in images:
    print(image['src'])
```



Lambda Expressions

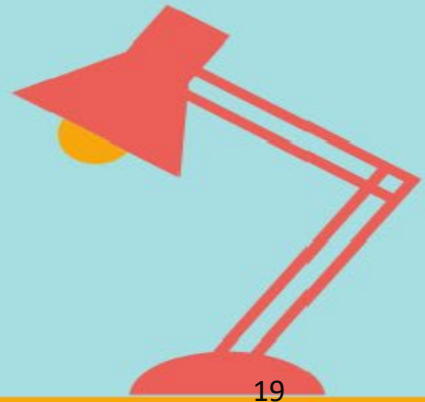
function definition

```
def max(m, n):
    return m if m > n else n

print(max(10, 3)) # 顯示 10
```

Lambda function

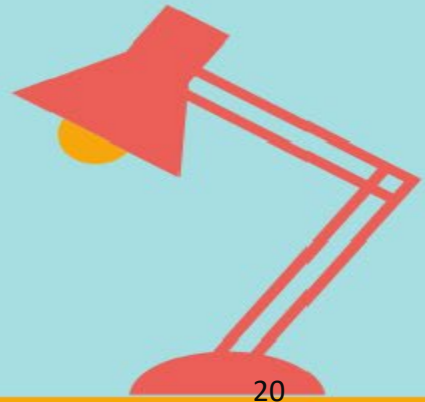
```
max = lambda m, n: m if m > n else n
print(max(10, 3)) # 顯示 10
```

```
bs.find_all(lambda tag: len(tag.attrs) == 2)

bs.find_all(lambda tag: tag.get_text() == 'Or maybe he\'s only resting?')
```



Writing Web Crawlers

```
from urllib.request import urlopen
from bs4 import BeautifulSoup

html = urlopen('http://en.wikipedia.org/wiki/Kevin_Bacon')
bs = BeautifulSoup(html, 'html.parser')
for link in bs.find_all('a'):
    if 'href' in link.attrs:
        print(link.attrs['href'])
```

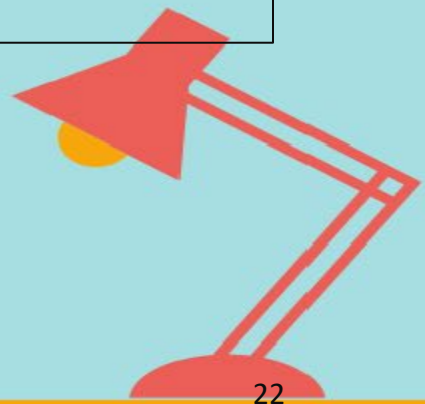


Recursively crawling an entire site

```
from urllib.request import urlopen
from bs4 import BeautifulSoup
import re

pages = set()
def getLinks(pageUrl):
    global pages
    html = urlopen('http://en.wikipedia.org{}'.format(pageUrl))
    bs = BeautifulSoup(html, 'html.parser')
    for link in bs.find_all('a', href=re.compile('^(/wiki/)')):
        if 'href' in link.attrs:
            if link.attrs['href'] not in pages:
                #We have encountered a new page
                newPage = link.attrs['href']
                print(newPage)
                pages.add(newPage)
                getLinks(newPage)
getLinks('')
```

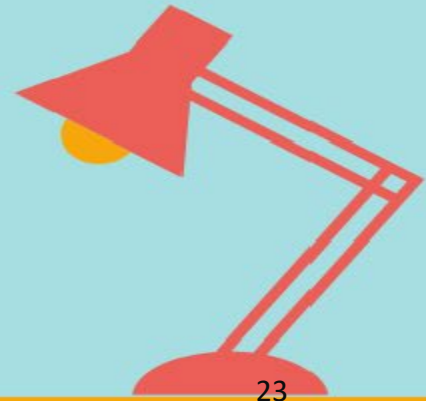
Regular Expressions and BeautifulSoup

```
from urllib.request import urlopen
from bs4 import BeautifulSoup
import re

html = urlopen('http://www.pythonscraping.com/pages/page3.html')
bs = BeautifulSoup(html, 'html.parser')
images = bs.find_all('img', {'src':re.compile('\.\.\/img\/gifts/img.*\.jpg')})
for image in images:
    print(image['src'])
```



Activity-1 (S-S)

- Open the Google Jamboard for the class
- Discuss what you know (K), want to know (W), and have learned (L) about this topic in the KWL chart.



Thanks!

Q&A