Introduction to Python 3:

Link to challenge: https://academy.hackthebox.com/module/88

(log in required)

Class: Tier I | Easy | General

Python Fundamentals

Conditional Statements and Loops:

Question: How long is list_1 ?

Answer: 6

Method:

```
len(list_1)
```

Question: In "Code block 2" the blank should be filled with what, to output all numbers in a terminal?

Answer: print(num)

Method:

Question: What is the result of running the code in "Code block 3"?

Answer: Ac4deMY!

Method:

Keeping It Simple and Smart

Defining Functions:

Question: Write the function signature (def ...) for a function "foo" that has one argument "bar", including the trailing colon.

Answer: def foo(bar):

Method:

Question: When we call a function and explicitly set the value of a parameter, e.g. foo(bar=42), this parameter is called a _____ parameter. (Fill the blank)

Answer: named

Method:

Question: Functions which parameters are not named explicitly are called _____ parameters. (Fill the blank)

Answer: positional

Method:

Word Extractor

The First Iterations:

Question: What is the 3rd most used word on the exercise target website?

Answer: Turbine

Method: we will use the following python script to get the HTML content of the target website, enumerate the words in it and count them to retrieve the 3rd most frequent word:

```
import requests
import re
from bs4 import BeautifulSoup


PAGE_URL = 'http://<target-IP>:<target-port>'


def get_html_of(url):
    resp = requests.get(url)


    if resp.status_code != 200:
        print(f'HTTP status code of {resp.status_code}
returned, but 200 was expected. Exiting...')
        exit(1)


    return resp.content.decode()


html = get_html_of(PAGE_URL)
soup = BeautifulSoup(html, 'html.parser')
raw_text = soup.get_text()


all_words = re.findall(r'\w+', raw_text)
```

```
word_count = {}

for word in all_words:
    if word not in word_count:
        word_count[word] = 1
    else:
        current_count = word_count.get(word)
        word_count[word] = current_count + 1

top_words = sorted(word_count.items(), key=lambda item:
item[1], reverse=True)

for i in range(3):
    print(top_words[i][0])
```

```
┌─[eu-academy-2]─[10.10.15.14]─[htb-ac-1099135@htb-kv6oduiw22]─[~]
└──╼ [★]$ cat word_finder.py
import requests
import re
from bs4 import BeautifulSoup

PAGE_URL = 'http://83.136.254.158:49649'

def get_html_of(url):
    resp = requests.get(url)

    if resp.status_code != 200:
        print(f'HTTP status code of {resp.status_code} returned, but 200 was expected. Exiting...')
        exit(1)

    return resp.content.decode()
```

```
html = get_html_of(PAGE_URL)
soup = BeautifulSoup(html, 'html.parser')
raw_text = soup.get_text()

all_words = re.findall(r'\w+', raw_text)

word_count = {}

for word in all_words:
    if word not in word_count:
        word_count[word] = 1
    else:
        current_count = word_count.get(word)
        word_count[word] = current_count + 1

top_words = sorted(word_count.items(), key=lambda item: item[1], reverse=True)

for i in range(3):
    print(top_words[i][0])
```

And upon running it:

```
┌─[eu-academy-2]─[10.10.15.14]─[htb-ac-1099135@htb-kv6oduiw22]─[~]
└──╼ [★]$ python word_finder.py
and
StarGusts
Turbine ◄──
```

Continuously Improving The Code:

Question - Optional: What's the price per month for a family account, after the initial trial period?

Answer: $5

Method: script is in section's guide.

Further Improvements:

Question: Given a minimum word length of 9, what is the 3rd most frequent word on the target website?

Answer: Unlimited

Method: we will use the following python script:

```
import click
import requests
import re
from bs4 import BeautifulSoup


def get_html_of(url):
    resp = requests.get(url)


    if resp.status_code != 200:
        print(f'HTTP status code of {resp.status_code}
returned, but 200 was expected. Exiting...')
        exit(1)


    return resp.content.decode()


def count_occurrences_in(word_list, min_length):
    word_count = {}


    for word in word_list:
        if len(word) < min_length:
            continue
        if word not in word_count:
            word_count[word] = 1
```

```
        else:
            current_count = word_count.get(word)
            word_count[word] = current_count + 1
    return word_count


def get_all_words_from(url):
    html = get_html_of(url)
    soup = BeautifulSoup(html, 'html.parser')
    raw_text = soup.get_text()
    return re.findall(r'\w+', raw_text)


def get_top_words_from(all_words, min_length):
    occurrences = count_occurrences_in(all_words,
min_length)
    return sorted(occurrences.items(), key=lambda item:
item[1], reverse=True)


@click.command()
@click.option('--url', '-u', prompt='Web URL', help='URL of
webpage to extract from.')
@click.option('--length', '-l', default=0, help='Minimum
word length (default: 0, no limit).')
def main(url, length):
    the_words = get_all_words_from(url)
    top_words = get_top_words_from(the_words, length)


    for i in range(3):
        print(top_words[i][0])
```

```
if __name__ == '__main__':
    main()
```

and save it in the pwnbox under the name ‘script.py’.

```
┌[eu-academy-2]-[10.10.15.123]-[htb-ac-1099135@htb-tb3a9sft7g]-[~]
└──╼ [★]$ cat script.py
import click
import requests
import re
from bs4 import BeautifulSoup

def get_html_of(url):
    resp = requests.get(url)

    if resp.status_code != 200:
        print(f'HTTP status code of {resp.status_code} returned, but 200 was expected. Exiting...')
        exit(1)

    return resp.content.decode()
```

```
def count_occurrences_in(word_list, min_length):
    word_count = {}

    for word in word_list:
        if len(word) < min_length:
            continue
        if word not in word_count:
            word_count[word] = 1
        else:
            current_count = word_count.get(word)
            word_count[word] = current_count + 1
    return word_count

def get_all_words_from(url):
    html = get_html_of(url)
    soup = BeautifulSoup(html, 'html.parser')
    raw_text = soup.get_text()
    return re.findall(r'\w+', raw_text)
```

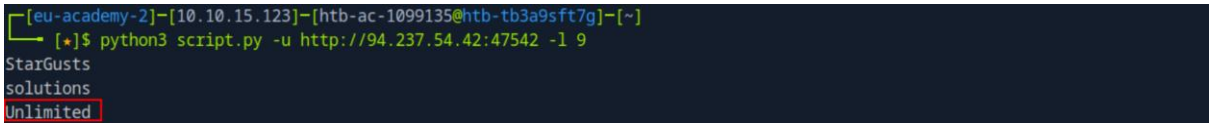
```
def get_top_words_from(all_words, min_length):
    occurrences = count_occurrences_in(all_words, min_length)
    return sorted(occurrences.items(), key=lambda item: item[1], reverse=True)

@click.command()
@click.option('--url', '-u', prompt='Web URL', help='URL of webpage to extract from.')
@click.option('--length', '-l', default=0, help='Minimum word length (default: 0, no limit).')
def main(url, length):
    the_words = get_all_words_from(url)
    top_words = get_top_words_from(the_words, length)

    for i in range(3):
        print(top_words[i][0])

if __name__ == '__main__':
    main()
```

We can observe in the script that there are required execution parameters: ‘-u’ for url, and ‘-l’ for length.

So we will run the script with the following parameters: url of target machine IP and port, and length=9:

```
python3 script.py -u http://<target-IP>:<target-port> -l 9
```

and take the third option:



Turning it up to 11

Managing Libraries in Python (Continued):

Question: How long is foo?

Answer: 3

Method: Run the script:

```
foo = set()


for i in range(42):
    foo.add('Cake')


foo.add('Hello')
foo.add('World')
print(len(foo))
```

Question: The type of foo from question 1 is <class 'set'>. What is the type of x_coordinate?

Answer: <class 'tuple'>

Method: run the script:

```
x_coordinate = (42,)
print(type(x_coordinate))
```

Question: What is the environment variable called which lets us define a search path for external libraries?

Answer: PYTHONPATH

Method: