PYTHON

WORKSHOP

ABOUT ME

Abner Andino - Python & Go developer

Github - https://github.com/bighelmet7

Linkedin - https://www.linkedin.com/in/abner-andino-moran/

Instagram - @bighelmet7

Backend Python Developer

(Actual) DevOps - ITNow

4-whitespace indentation

none brackets

snake_case

PEP8 & PEP20 & PEP257

CODES & MORE

https://github.com/bighelmet7/python-workshop

https://www.pythonanywhere.com/user/bighelmet7/shares/3c30c1943da94977a379f2ac98fcbeda/ (Boring)

https://www.pythonanywhere.com/user/bighelmet7/shares/8f346cd1bab949fdb20ead58bc4d232b/ (Multi-Boring)

https://www.pythonanywhere.com/user/bighelmet7/shares/8f346cd1bab949fdb20ead58bc4d232b/ (FizzBuzz)

https://www.pythonanywhere.com/user/bighelmet7/shares/cd204b6fecfe48e4a3ce7260e1a936aa/ (Instagram)

https://www.pythonanywhere.com/user/bighelmet7/shares/4ed7804f497245ef8501aff45c8934ab/ (Github)

BORING

shebang

import's

function, parameters, default values

docstring

main

```python
#!/usr/bin/python
import time

def boring(sleep_time=0):
    """
    Args:
    sleep_time (int): sleeping boring time.
    """
    print("this is a boring function")
    time.sleep(sleep_time)

def main():
    boring(1)

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

MULTI-BORING

Loops

built-in functions

format string & string adding

```python
#!/usr/bin/python
import time

def boring(sleep_time=0):
    """
    Args:
    sleep_time (int): sleeping boring time.
    """
    print("this is a boring function")
    time.sleep(sleep_time)

def main():
    many_borings = 10
    for i, _ in enumerate(range(many_borings)):
        message = "{iteration} - boring stuff going on".format(iteration=str(i))
        print(message)
        boring(1)

if __name__ == '__main__':
    main()

```

FIZZBUZZ

if, elif, else

```python
#!/usr/bin/python

def fizz_buzz(n_numbers):
    """
    Args:
    n_numbers (int): total of numbers to be treated.

    Prints Fizz when a number is multiple of 3, Buzz when its for 5 and
    FizzBuzz when its of both. Otherwise print the number.
    """
    for i in range(1, n_numbers + 1):
        if i % 3 == 0 and i % 5 == 0:
            print('FizzBuzz')
        elif i % 3 == 0:
            print('Fizz')
        elif i % 5 == 0:
            print('Buzz')
        else:
            print(i)

def main():
    fizz_buzz(15)

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

INSTAGRAM

OOP

magic-functions

args & kwargs

```python
#!/usr/bin/python

class User(object):

    def __init__(self, id, name, age, pic_profile=None):
        self.id = id
        self.name = name
        self.age = age
        self.pic_url = pic_profile
        self.friends = []

    def display_more_info(self, *args, **kwargs):
        print("Args info: ", args, "Kwargs info: ", kwargs)

    def __repr__(self):
        return '<ID %s> Name: %s - Age: %s' % (self.id, self.name, self.age)

def main():
    annie = User(id=0, name='Annie', age=22)
    jhon = User(id=1, name='Jhon', age=28, pic_profile="https://pic.inst.es/1")
    print(annie)
    print(jhon)
    annie.friends.append(jhon)
    annie.display_more_info(
        "Status of today...",
        "Status of 2 seconds ago",
        history=["London", "Denmark", "Poland"],
        blocked=["Danna"]
    )

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

BONUS

try/except/finally/else

Generators

Context manager

Decorators

Duck-typing

packages (__init__.py)

GITHUB CRAWLER

```python
#!/usr/bin/python
from bs4 import BeautifulSoup
import json
import requests

class GithubException(Exception): pass


def search_on_github(query=''):
    url = 'https://github.com/search?q=%s' % query
    resp = requests.get(url)
    if not resp:
        raise GithubException("Could not search the query %s" % query)
    return resp.content

def main():
    content = search_on_github('python')
    soup = BeautifulSoup(content, features='html.parser')
    repo_list = soup.find('ul', {'class': 'repo-list'})
    repo_list_item = repo_list.find_all('li', {'class': 'repo-list-item'})
    for item in repo_list_item:
        div_url = item.find('a', {'class': 'v-align-middle'})
        data_hydro_click = div_url.get('data-hydro-click', {})
        try:
            data_hydro_click = json.loads(data_hydro_click)
        except Exception:
            raise GithubException('Could not parse the URL')
        payload = data_hydro_click.get('payload', {})
        result = payload.get('result', {})
        url = result.get('url', '')
        print(url)

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

https://forms.gle/fZoeTrHX5xugqaJK7