Option [1]

Project Brief: Top Trumps

In this project you'll create a small game where players compare stats, similar to the Top Trumps card game. The basic flow of the games is:

1. You are given a random card with different stats
2. You select one of the card's stats
3. Another random card is selected for your opponent (the computer)
4. The stats of the two cards are compared
5. The player with the stat higher than their opponent wins

The standard project will use the Pokemon API, but you can use a different API if you want after completing the required tasks.

You will not need any additional knowledge beyond what is covered in this course to complete this project.

Required Tasks

These are the required tasks for this project. You should aim to complete these tasks before adding your own ideas to the project.

1. Generate a random number between 1 and 151 to use as the Pokemon ID number
2. Using the Pokemon API get a Pokemon based on its ID number
3. Create a dictionary that contains the returned Pokemon's name, id, height and weight (★ https://pokeapi.co/)
4. Get a random Pokemon for the player and another for their opponent
5. Ask the user which stat they want to use (id, height or weight)
6. Compare the player's and opponent's Pokemon on the chosen stat to decide who wins

Ideas for Extending the Project

Here are a few ideas for extending the project beyond the required tasks. These ideas are just suggestions, feel free to come up with your own ideas and extend the program however you want.

- Use different stats for the Pokemon from the API
- Get multiple random Pokemon and let the player decide which one that they want to use
- Play multiple rounds and record the outcome of each round. The player with most number of rounds won, wins the game

- Allow the opponent (computer) to choose a stat that they would like to compare
- Record high scores for players and store them in a file
- Use a different API (see suggestions below)

Useful Resources

Harry Potter API

- Homepage ★ https://hp-api.herokuapp.com/

Star Wars API

- Homepage ★ https://swapi.co/
- Documentation ★ https://swapi.co/documentation

Anime/Manga API

- Homepage ★ jikan.moe/
- Documentation ★ jikan.docs.apiary.io
- Example API url ★ https://api.jikan.moe/v3/anime/5

Example Project Code

In this section you will find some example code to complete the required tasks. You can use this code for guidance if you are finding it difficult to complete the required tasks for this project.

```
import random

import requests


def random_pokemon():
    pokemon_number = random.randint(1, 151)
    url = 'https://pokeapi.co/api/v2/pokemon/{}/'.format(pokemon_number)
    response = requests.get(url)
    pokemon = response.json()

    return {
        'name': pokemon['name'],
        'id': pokemon['id'],
        'height': pokemon['height'],
        'weight': pokemon['weight'],
    }
```

```
def run():
    my_pokemon = random_pokemon()

    print('You were given {}'.format(my_pokemon['name']))
    stat_choice = input('Which stat do you want to use? (id, height,
weight) ')

    opponent_pokemon = random_pokemon()
    print('The opponent chose {}'.format(opponent_pokemon['name']))

    my_stat = my_pokemon[stat_choice]
    opponent_stat = opponent_pokemon[stat_choice]

    if my_stat > opponent_stat:
        print('You Win!')
    elif my_stat < opponent_stat:
        print('You Lose!')
    else:
        print('Draw!')


run()
```

Option [2]

Project Brief: Search

In this project you'll create a program to search for recipes based on an ingredient. The standard project uses the Food2Fork API, but can be changed to use a different API after completing the required tasks.

You will not need any additional knowledge beyond what is covered in this course to complete this project.

Required Tasks

These are the required tasks for this project. You should aim to complete these tasks before adding your own ideas to the project.

1. Read the Food2Fork API documentation ★ https://www.food2fork.com/about/api
2. Ask the user to enter an ingredient that they want to search for
3. Create a function that makes a request to the Food2Fork API with the required ingredient as part of the search query
4. Get the returned recipes from the API response
5. Display the recipes for each search result

Ideas for Extending the Project

Here are a few ideas for extending the project beyond the required tasks. These ideas are just suggestions, feel free to come up with your own ideas and extend the program however you want.

- Save the results to a file
- Filter the recipes based on rating
- Ask the user additional questions to decide which recipe they should choose
- Use a different searchable API (suggestions in useful resources)

Useful Resources

API for Anime and Manga

- Homepage ★ jikan.moe/
- Documentation ★ jikan.docs.apiary.io
- Example search ★ https://api.jikan.moe/v3/search/anime?q=Sailor%20Moon

Spotify API (can be very difficult to use as it requires a much more complicated setup process)

- Official Documentation ★ https://developer.spotify.com/documentation/web-api/
- Python library for Spotify API ★ https://spotipy.readthedocs.io/en/latest/

Twitter API (can be very difficult to use as it requires a more complicated setup process)

- Official Documentation ★ https://developer.twitter.com/en/docs.html
- Python library for Twitter API ★ http://docs.tweepy.org/en/v3.5.0/api.html

Example Project Code

In this section you will find some example code to complete the required tasks. You can use this code for guidance if you are finding it difficult to complete the required tasks for this project.

```
import requests


def recipe_search(ingredient):
    # Register to get an API key https://www.food2fork.com/about/api
    api_key = ''
    result =
requests.get('https://www.food2fork.com/api/search?key={}&q={}'.format(api_
key, ingredient))
    data = result.json()

    return data['recipes']


def run():
    ingredient = input('Enter an ingredient: ')

    recipes = recipe_search(ingredient)

    for recipe in recipes:
        print(recipe['title'])
        print(recipe['source_url'])
        print()


run()
```

Option [3]

Project Brief: Spreadsheet Analysis

In this project you'll use Python to do very basic data analysis on a spreadsheet. The standard project will use csv file that contains fake sales data. After completing the required tasks you are free to change the csv file that you use.

The csv file for the standard project is called **sales.csv** and is included with your course guide.

You will not need any additional knowledge beyond what is covered in this course to complete this project.

Required Tasks

These are the required tasks for this project. You should aim to complete these tasks before adding your own ideas to the project.

1. Read the data from the spreadsheet
2. Collect all of the sales from each month into a single list
3. Output the total sales across all months

Ideas for Extending the Project

Here are a few ideas for extending the project beyond the required tasks. These ideas are just suggestions, feel free to come up with your own ideas and extend the program however you want.

- Output a summary of the results to a spreadsheet
- Calculate the following:
    - Monthly changes as a percentage
    - The average
    - Months with the highest and lowest sales
- Use a data source from a different spreadsheet (see Useful Resources)

Useful Resources

Free datasets at Kaggle

- Datasets ★ https://www.kaggle.com/datasets
- Documentation ★ https://www.kaggle.com/docs/datasets

Working with Excel files in Python

- Homepage ★ https://github.com/python-excel/xlrd

- Documentation ★ https://xlrd.readthedocs.io/en/latest/

Seaborn graphing library

- Homepage ★ https://seaborn.pydata.org/
- Documentation ★ https://seaborn.pydata.org/tutorial.html
- Tutorial ★ https://www.datacamp.com/community/tutorials/seaborn-python-tutorial
- Tutorial ★ https://www.geeksforgeeks.org/plotting-graph-using-seaborn-python/

Example Project Code

In this section you will find some example code to complete the required tasks. You can use this code for guidance if you are finding it difficult to complete the required tasks for this project.

```
import csv


def read_data():
    data = []

    with open('sales.csv', 'r') as sales_csv:
        spreadsheet = csv.DictReader(sales_csv)
        for row in spreadsheet:
            data.append(row)

    return data


def run():
    data = read_data()

    sales = []
    for row in data:
        sale = int(row['sales'])
        sales.append(sale)

    total = sum(sales)
    print('Total sales: {}'.format(total))


run()
```