

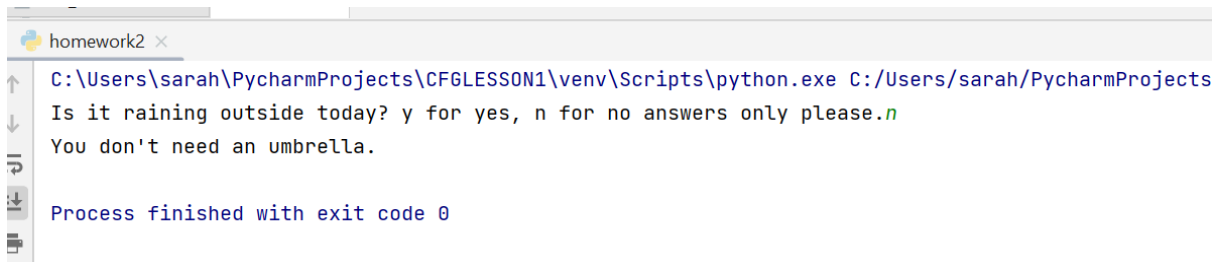
## **TASK 1**

### 1. CODE BELOW

```
is_it_raining = input("Is it raining outside today? y for yes, n for no answers only please.")

if is_it_raining == "y":
    print("Take an umbrella.")
else:
    print("You don't need an umbrella.")
```

### OUTPUT BELOW



```
homework2 x
C:\Users\sarah\PycharmProjects\CFGLESSON1\venv\Scripts\python.exe C:/Users/sarah/PycharmProjects
Is it raining outside today? y for yes, n for no answers only please.n
You don't need an umbrella.

Process finished with exit code 0
```

2. This code isn't quite right for two reasons. First of all, in the if statement, when the amount of money you have is less than the boat cost, it prints "you can afford the boat hire". If the amount of money you have is less than the cost of the boat hire, then you cannot afford it, and so the print statement under this condition should be "you cannot afford the boat hire", with the else statement being "you can afford the boat hire". The second reason is that the comparison operator is comparing an integer with a string, as the input value is always saved as a string. Python will error when comparing an integer with a string as they are not the same object type. To fix this, we should pass the input through the `int()` type to turn it into an integer, allowing Python to compare it. Example of correct code below:

```
# the correct code for question 2
#
my_money = int(input("How much money do you have?"))
boat_cost = 20 + 5

if my_money < boat_cost:
    print("You cannot afford the boat hire")
else:
    print("You can afford the boat hire")
```

3. CODE BELOW - Please note the mistake in the question (in the example giving 1800s as Eighteenth Century). 1800s is Nineteenth Century, and 1900s is Twentieth Century

```
def find_century(year_written):
    if year_written[:2] == "18":
        return "Nineteenth Century" # NB - there is a mistake in the Question - 1800s is 19th century
    elif year_written[:2] == "19":
        return "Twentieth Century" # NB there is a mistake in the question, 1900s is 20th century
    else:
        return "We do not stock books in this century - check again"

def find_decade(year_written):
    if year_written[2:3] == "0":
        return "Noughties"
    elif year_written[2:3] == "1":
        return "Tens"
    elif year_written[2:3] == "2":
        return "Twenties"
    elif year_written[2:3] == "3":
        return "Thirties"
    elif year_written[2:3] == "4":
        return "Fourties"
    elif year_written[2:3] == "5":
        return "Fifties"
    elif year_written[2:3] == "6":
        return "Sixties"
    elif year_written[2:3] == "7":
        return "Seventies"
    elif year_written[2:3] == "8":
        return "Eighties"
    elif year_written[2:3] == "9":
        return "Nineties"
    else:
        return "Something has gone wrong - please check year format is in YYYY"

book_year = input("What year was the book written in?")
century_written = find_century(book_year)
decade_written = find_decade(book_year)

print(f"{century_written}, {decade_written}")
```

OUTPUT BELOW

```
C:\Users\sarah\PycharmProjects\CFGLESSON1\venv\Scripts\python.exe C:/Users/sara
What year was the book written in?1854
Nineteenth Century, Fifties
```

## TASK 2

- There are again 2 problems with this code. The first is that you are using index 1 from the list. The first item of a list is index 0, so using index 1 will retrieve the second item from the list. If you wish to retrieve the first item from the list, you must change the 1 to a 0 like the below example. The second problem is a stray comma at the end of the list next to "cheese board". The final item of the list does not need a comma, and will cause an error as Python is expecting an additional item after a comma:

```
shopping_list = [
    "oranges",
    "cat food",
    "sponge cake",
    "long-grain rice",
    "cheese board",
]

print(shopping_list[0])
```

- CODE BELOW:

```
chocolates = {
    'white': 1.50,
    'milk': 1.20,
    'dark': 1.80,
    'vegan': 2.00
}

chocolate_type = input("Which chocolate are you selling?")
print(chocolates[chocolate_type])
```

OUTPUT BELOW:

```
C:\Users\sarah\PycharmProjects\CFGLESSON1\venv\Scripts\python.exe C:/Users/sarah/Scripts/python.exe
Which chocolate are you selling?milk
1.2
```

```
Process finished with exit code 0
```

6. CODE BELOW:

```
import random

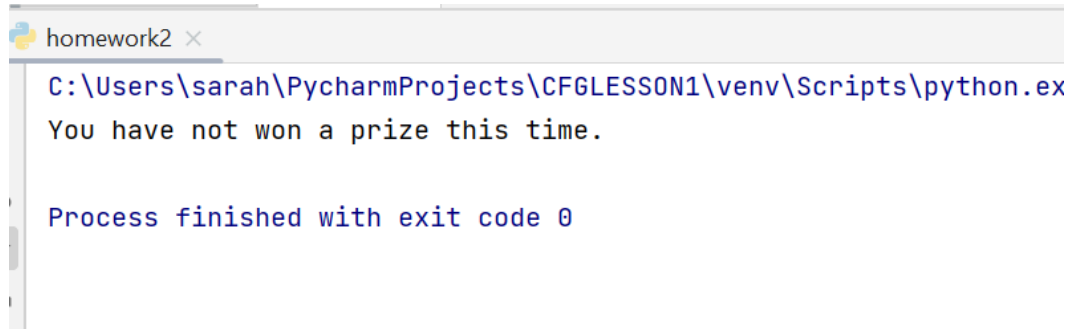
def prize_awarded(number_of_matches):
    if number_of_matches < 3:
        print("You have not won a prize this time.")
    elif number_of_matches == 3:
        print("Congratulations! You have won £20!")
    elif number_of_matches == 4:
        print("Congratulations! You have won £40!")
    elif number_of_matches == 5:
        print("Congratulations! You have won £100!")
    elif number_of_matches == 6:
        print("Congratulations! You have won £10000!")
    elif number_of_matches == 7:
        print("Congratulations! You have won £1000000!")
    else:
        print("Invalid number of matches")

lottery_ticket = [3, 5, 34, 20, 19, 42, 12]
random_numbers = random.sample(range(1,60),7)
print(random_numbers)
match_count = 0

for ticket_number in lottery_ticket:
    if ticket_number in random_numbers:
        match_count += 1

prize_awarded(match_count)
```

OUTPUT BELOW:



```
homework2 x
C:\Users\sarah\PycharmProjects\CFGLESSON1\venv\Scripts\python.exe
You have not won a prize this time.

Process finished with exit code 0
```

### **TASK 3**

7. In python, pip is a package installation manager. It allows you to install modules which are created as an addition to Python that aren't included as part of the standard Python 3 download. This facilitates many additional features and a wide range of uses for Python, as there are many additional tools not included in the default installation which might be more specific. One of the benefits of pip is that it allows you to simply install packages for Python from the command line, using simple commands rather than searching and downloading from the internet.
8. The problem with this code is that it is in 'read' mode rather than 'write' mode. This is signified by the 'r' in the with open brackets, next to the file to be opened. To fix the error, the 'r' should be changed to a 'w' for write mode as below:

```
poem = 'I like Python and I am not very good at poems'

with open('poem.txt', 'w') as poem_file:
    poem_file.write(poem)
```

9. CODE BELOW (next page):

```
import os.path

lyrics = "You could never know what it's like \n" \
        "Your blood like winter freezes just like ice \n" \
        "And there's a cold lonely light that shines from you \n" \
        "You'll wind up like the wreck you hide behind that mask you use \n" \
        "And did you think this fool could never win? \n" \
        "Well look at me, I'm coming back again \n" \
        "I got a taste of love in a simple way \n" \
        "And if you need to know while I'm still standing, you just fade away \n" \
        "Don't you know I'm still standing better than I ever did \n" \
        "Looking like a true survivor, feeling like a little kid \n" \
        "I'm still standing after all this time \n" \
        "Picking up the pieces of my life without you on my mind \n" \
        "I'm still standing (Yeah, yeah, yeah) \n" \
        "I'm still standing (Yeah, yeah, yeah) \n"

with open("song.txt", "w") as file:
    file.write(lyrics) # writing the lyrics to a new file

if os.path.isfile("song.txt"):
    print("File exists")
else:
    print("File not exist") # checking the file has been successfully created

with open("song.txt", "r") as file:
    for line in file:
        line2 = line.strip().lower()
        line2 = line.translate(line.maketrans("", "", '!"#$%&\'()*+,-./:;<=>?@[\\]^_`{|}~'))
        if " still " in line2:
            print(line) # printing the lines with still
```

OUTPUT:

```
C:\Users\sarah\PycharmProjects\CF6LESSON1\venv\Scripts\python.exe C:/Users/sarah/PycharmProjects/CF6LESSON1/homework2.py
File exists
And if you need to know while I'm still standing, you just fade away

Don't you know I'm still standing better than I ever did

I'm still standing after all this time

I'm still standing (Yeah, yeah, yeah)

I'm still standing (Yeah, yeah, yeah)
```

## TASK 4

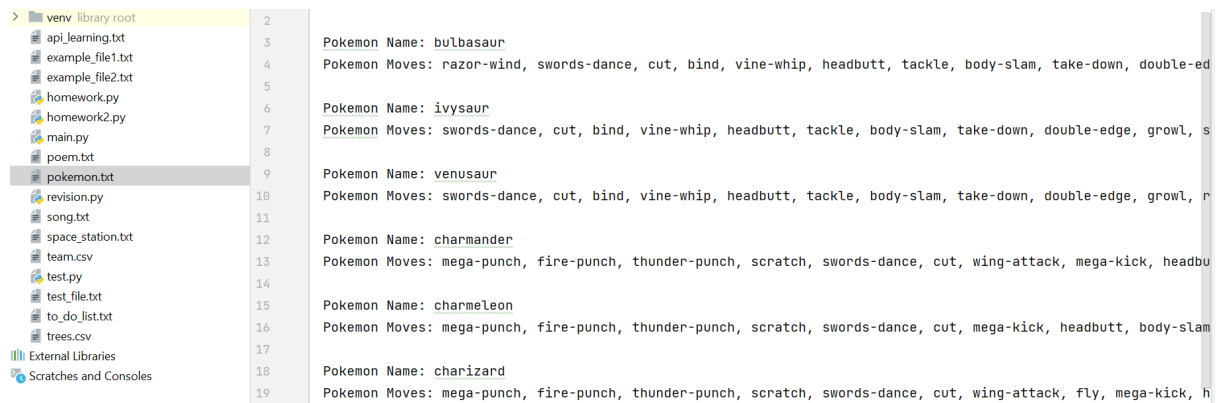
### 10. CODE BELOW:

```
import requests

list_of_pokemon = ["1", "2", "3", "4", "5", "6"]
endpoint_without_number = "https://pokeapi.co/api/v2/pokemon/"

for pokemon in list_of_pokemon:
    endpoint = endpoint_without_number + pokemon
    response = requests.get(endpoint)
    data = response.json()
    pokemon_name = data["name"]
    pokemon_moves_list = [dictionary["move"]["name"] for dictionary in data["moves"]]
    with open("pokemon.txt", "w") as file:
        file.write(f"\n\n Pokemon Name: {pokemon_name} \n "
                  f"Pokemon Moves: " + ", ".join(pokemon_moves_list))
```

### OUTPUT TO FILE BELOW:



```
2
3 Pokemon Name: bulbasaur
4 Pokemon Moves: razor-wind, swords-dance, cut, bind, vine-whip, headbutt, tackle, body-slam, take-down, double-ed
5
6 Pokemon Name: ivysaur
7 Pokemon Moves: swords-dance, cut, bind, vine-whip, headbutt, tackle, body-slam, take-down, double-edge, growl, s
8
9 Pokemon Name: venusaur
10 Pokemon Moves: swords-dance, cut, bind, vine-whip, headbutt, tackle, body-slam, take-down, double-edge, growl, r
11
12 Pokemon Name: charmander
13 Pokemon Moves: mega-punch, fire-punch, thunder-punch, scratch, swords-dance, cut, wing-attack, mega-kick, headbu
14
15 Pokemon Name: charmeleon
16 Pokemon Moves: mega-punch, fire-punch, thunder-punch, scratch, swords-dance, cut, mega-kick, headbutt, body-slam
17
18 Pokemon Name: charizard
19 Pokemon Moves: mega-punch, fire-punch, thunder-punch, scratch, swords-dance, cut, wing-attack, fly, mega-kick, h
```

11. The name of the API is the Open Trivia Database and it provides random quiz questions which you can tailor with difficulty, category and type as well as the number of questions provided. You could make a call to this API with the following code:

The specific URL here is <https://opentdb.com/api.php?amount=3> (for example).

```
import requests

amount_questions = input("How many questions woud you like?")
endpoint1 = f"https://opentdb.com/api.php?amount={amount_questions}"

response = requests.get(endpoint1)
data = response.json()
```

In terms of the output from the API, that looks like a dictionary below after being decoded from JSON:

```
{'response_code': 0,
 'results': [{'category': 'Geography',
              'correct_answer': 'Yangtze',
              'difficulty': 'hard',
              'incorrect_answers': ['Fujian', 'Sichuan', 'Guangdong'],
              'question': 'Which of these is NOT a province in China?',
              'type': 'multiple'},
             {'category': 'History',
              'correct_answer': 'Temüjin',
              'difficulty': 'medium',
              'incorrect_answers': ['Möngke', 'Ögedei', 'Temür'],
              'question': 'What was Genghis Khan's real name?',
              'type': 'multiple'},
             {'category': 'Entertainment: Video Games',
              'correct_answer': 'Jordan "Thermite" Trace',
              'difficulty': 'medium',
              'incorrect_answers': ['Eliza "Ash" Cohen',
                                   'Seamus "Sledge" Cowden',
                                   'Dominic "Bandit" Brunsmeier'],
              'question': 'Which of these operators from "Tom Clancy's Rainbow Six Siege" has the '}
```