Homework week 3

TASK 1 (Conditional flow)

Question 1

Create a program that tells you whether or not you need an umbrella when you leave the house.
The program should:
1. Ask you if it is raining using input()
2. If the input is 'y', it should output 'Take an umbrella'
3. If the input is 'n', it should output 'You don't need an umbrella'

Answer 1

```
is_raining = input('Is it raining? y/n ')

if is_raining == 'y':
    print("Take an umbrella.")
elif is_raining == 'n':
    print("You don't need an umbrella.")
else:
    print("Invalid input")
```

```
Is it raining? y/n y
Take an umbrella.
```

Question 2

I'm on holiday and want to hire a boat. The boat hire costs £20 + a refundable £5 deposit. I've written a program to check that I can afford the cost, but something doesn't seem right. Have a look at my program and work out what I've done wrong

```
my_money = input('How much money do you have? ')
boat cost = 20 + 5
if my_money < boat_cost:
        print('You can afford the boat hire')
else :
        print('You cannot afford the board hire'
```

Answer 2

There are a few mistakes in the program. my_money should be transformed into an integer, there should be underscore in boat_cost, my_money should be greater than boat_cost and there's a bracket missing at the end. Code should look like:

```
my_money = int(input('How much money do you have? '))
boat_cost = 20 + 5
if my_money > boat_cost:
    print('You can afford the boat hire')
else :
    print('You cannot afford the board hire')
```

```
How much money do you have? 70
You can afford the boat hire
```

Question 3

Your friend works for an antique book shop that sells books between **1800** and **1950** and wants to quickly categorise books by the century and decade that they were written.

Write a program that takes a year (e.g. 1872) and outputs the century and decade (e.g. "Eighteenth Century, Seventies") *There's a mistake in the question, 1872 is 19th century, not 18th ;)*

Answer 3

```
acceptable_values = list(range(1800,1951))
date = int(input("What year was the book published? "))

if date in acceptable_values:
    if date <= 1819:
        print("Nineteenth Century, First Decade")
    elif date <= 1829:
        print("Nineteenth Century, Twenties")
    elif date <= 1839:
        print("Nineteenth Century, Thirties")
    elif date <= 1849:
        print("Nineteenth Century, Forties")
    elif date <= 1859:
        print("Nineteenth Century, Fifties")
    elif date <= 1869:
        print("Nineteenth Century, Sixties")
    elif date <= 1879:
        print("Nineteenth Century, Seventies")
    elif date <= 1889:
        print("Nineteenth Century, Eighties")
    elif date <= 1899:
        print("Nineteenth Century, Nineties")
    elif date <= 1919:
        print("Twentieth Century, First Decade")
    elif date <= 1929:
        print("Twentieth Century, Twenties")
    elif date <= 1939:
        print("Twentieth Century, Thirties")
    elif date <= 1949:
        print("Twentieth Century, Forties")
    elif date == 1950:
        print("Twentieth Century, Fifties")
else:
    print("Our bookshop sells only books published between 1800 and 1950.")
```

```
What year was the book published? 1940
Twentieth Century, Forties
```

TASK 2 (Lists and Dictionaries)

Question 1

I have a list of things I need to buy from my supermarket of choice.

```
shopping_list = [
"oranges",
"cat food",
"sponge cake",
"long-grain rice",
"cheese board"
]
print(shopping_list[1])
```

I want to know what the first thing I need to buy is. However, when I run the program it shows me a different answer to what I was expecting? What is the mistake? How do I fix it.

Answer 1

Index in Python starts at zero, so they should print(shopping_list[0]).

Question 2

I'm setting up my own market stall to sell chocolates. I need a basic till to check the prices of different chocolates that I sell. I've started the program and included the chocolates and their prices. Finish the program by **asking the user to input an item and then output its price**.

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

Answer 2

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

user_choice = input('Which chocolate you would like to buy? Available choices are: white, milk, dark or vegan. ')
print(f'The price for this chocolate is: £{chocolates[user_choice]}')
```

```
Which chocolate you would like to buy? Available choices are: white, milk, dark or vegan. vegan
The price for this chocolate is: £2.0
```

Question 3

Write a program that simulates a lottery. The program should have a list of seven numbers that represent a lottery ticket. It should then generate seven random numbers. After comparing the two sets of numbers, the program should output a prize based on the number of matches:

· £20 for three matching numbers
· £40 for four matching numbers
· £100 for five matching numbers
· £10000 for six matching numbers
· £1000000 for seven matching numbers

Answer 3

```python
import random

# list of seven numbers representing lottery ticket
lottery_numbers = [2, 6, 8, 9, 16, 18, 20]

# generate seven unique numbers in a range
random_numbers = random.sample(range(1,21), k=7)

# find matches
common_numbers = set(lottery_numbers).intersection(random_numbers)

# print tidied statements
print(f'The lottery numbers for today are: {", ".join(map(str, lottery_numbers))}')

print(f'Your numbers are: {", ".join(map(str, sorted(random_numbers)))}')

print(f'The numbers that match are: {", ".join(map(str, sorted(common_numbers)))}')

# calculate winnings
if len(common_numbers) < 3:
    print("Better luck next time!")
if len(common_numbers) == 3:
    print("You have won £20!")
if len(common_numbers) == 4:
    print("You have won £40!")
if len(common_numbers) == 5:
    print("You have won £100!")
if len(common_numbers) == 6:
    print("You have won £10000!")
if len(common_numbers) == 7:
    print("JACKPOT! You have won £1000000!")
```

```
The lottery numbers for today are: 2, 6, 8, 9, 16, 18, 20
Your numbers are: 5, 6, 8, 11, 14, 19, 20
The numbers that match are: 6, 8, 20
You have won £20!
```

TASK 3 (Read and Write files)

Question 1

You're having coffee/tea/beverage of your choice with a friend that is learning to program in Python. They're curious about why they would use pip. Explain what pip is and one benefit of using pip.

Answer 1

pip is a package manager that allows to install and manage external libraries. One of the main benefits I can see is that it really expands the programming possibilities by allowing installation of very exciting packages (that can be found on https://pypi.org/).

Question 2

This program should save my data to a file, but it doesn't work when I run it. What is the problem and how do I fix it?

```
poem = 'I like Python and I am not very good at poems'
with open('poem.txt', 'r') as poem_file:
        poem_file.write(poem)
```

Answer 2

The program should be in a write mode, not read mode:

```python
poem = 'I like Python and I am not very good at poems'
with open('poem.txt', 'w') as poem_file:
    poem_file.write(poem)
    print('Poem written to a file')
```

```
Poem written to a file
```

Question 3

Here is a snippet of Elton John's song "I'm Still Standing"
You could never know what it's like
Your blood like winter freezes just like ice
And there's a cold lonely light that shines from you
You'll wind up like the wreck you hide behind that mask you use
And did you think this fool could never win?
Well look at me, I'm coming back again
I got a taste of love in a simple way
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
Looking like a true survivor, feeling like a little kid
I'm still standing after all this time
Picking up the pieces of my life without you on my mind
I'm still standing (Yeah, yeah, yeah)
I'm still standing (Yeah, yeah, yeah)

Tasks:
1. Write the lyrics to a new file called song.txt
2. Check that a file has been created successfully.
3. The read lines from this file and print out ONLY those lines that have a word 'still' in them.

Answer 3

```
# store lyrics in a variable
lyrics = '''You could never know what it's like
Your blood like winter freezes just like ice
And there's a cold lonely light that shines from you
You'll wind up like the wreck you hide behind that mask you use
And did you think this fool could never win?
Well look at me, I'm coming back again
I got a taste of love in a simple way
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
Looking like a true survivor, feeling like a little kid
I'm still standing after all this time
Picking up the pieces of my life without you on my mind
I'm still standing (Yeah, yeah, yeah)
I'm still standing (Yeah, yeah, yeah)'''

# write lyrics to a file
with open('song.txt', 'w') as file:
    file.write(lyrics)
    print('Lyrics written to a file')
```

```
Lyrics written to a file
```

```
# check if file has been created
with open('song.txt', 'r') as file:
    [print(line.strip()) for line in file.readlines()]
```

```
You could never know what it's like
Your blood like winter freezes just like ice
And there's a cold lonely light that shines from you
You'll wind up like the wreck you hide behind that mask you use
And did you think this fool could never win?
Well look at me, I'm coming back again
I got a taste of love in a simple way
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
Looking like a true survivor, feeling like a little kid
I'm still standing after all this time
Picking up the pieces of my life without you on my mind
I'm still standing (Yeah, yeah, yeah)
I'm still standing (Yeah, yeah, yeah)
```

```
# print lines with word 'still' only
with open('song.txt', 'r') as file:
    [print(line.strip()) for line in file.readlines() if 'still' in line]
```

```
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)
```

```
# If we also wanted to include possibility of 'Still':
with open('song.txt', 'r') as file:
    [print(line.strip()) for line in file.readlines() if 'still' in line.lower()]
```

```
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 (API)

Question 1

In this session you used the Pokémon API to retrieve a single Pokémon. I want a program that can retrieve multiple Pokémon and save their names and moves to a file. Use a list to store about 6 Pokémon IDs. Then in a for loop call the API to retrieve the data for each Pokémon. Save their names and moves into a file called 'pokemon.txt'

Answer 1

```
import requests
import random

# create a list of 6 unique pokemon numbers from a sample of 100
pokemon_number = sorted(random.sample(range(1,101), k=6))

# with file open, loop through the numbers and retrieve needed information. Format the answer and write it to a file.
with open('pokemon.txt', 'w') as file:
    for number in pokemon_number:
        pokemon = requests.get(f'https://pokeapi.co/api/v2/pokemon/{number}/').json()
        file.write("Pokemon name: " + pokemon['name'].capitalize() + ".\n")
        file.write("Its moves are: " )
        [file.write(move['move']['name'].capitalize() + ". ") for move in pokemon['moves']]
        file.write("\n")
    print("Written to a file.")
```

```
Written to a file.
```

```
# to check if it's there:
with open('pokemon.txt', 'r') as file:
    [print(line.strip()) for line in file]
```

```
Pokemon name: Charizard.
Its moves are: Mega-punch. Fire-punch. Thunder-punch. Scratch. Swords-dance. Cut. Wing-attack. Fly. Mega-kick. Headbutt. Body-s
lam. Take-down. Double-edge. Leer. Growl. Roar. Ember. Flamethrower. Hyper-beam. Submission. Counter. Seismic-toss. Strength. S
olar-beam. Dragon-rage. Fire-spin. Earthquake. Fissure. Dig. Toxic. Rage. Mimic. Double-team. Smokescreen. Defense-curl. Reflec
t. Bide. Fire-blast. Swift. Skull-bash. Fury-swipes. Rest. Rock-slide. Slash. Substitute. Snore. Curse. Protect. Scary-face. Mu
d-slap. Outrage. Sandstorm. Endure. False-swipe. Swagger. Fury-cutter. Steel-wing. Attract. Sleep-talk. Return. Frustration. Dy
namic-punch. Dragon-breath. Iron-tail. Metal-claw. Hidden-power. Twister. Sunny-day. Crunch. Rock-smash. Beat-up. Heat-wave. Wi
ll-o-wisp. Facade. Focus-punch. Helping-hand. Brick-break. Secret-power. Blaze-kick. Blast-burn. Weather-ball. Air-cutter. Over
heat. Rock-tomb. Aerial-ace. Dragon-claw. Dragon-dance. Roost. Natural-gift. Tailwind. Fling. Flare-blitz. Air-slash. Dragon-pu
lse. Focus-blast. Giga-impact. Shadow-claw. Fire-fang. Defog. Captivate. Ominous-wind. Hone-claws. Flame-burst. Flame-charge. R
ound. Echoed-voice. Sky-drop. Incinerate. Acrobatics. Inferno. Fire-pledge. Bulldoze. Dragon-tail. Work-up. Heat-crash. Hurrica
ne. Confide. Mystical-fire. Power-up-punch. Brutal-swing. Breaking-swipe. Scale-shot. Dual-wingbeat. Scorching-sands.
Pokemon name: Pidgey.
Its moves are: Razor-wind. Gust. Wing-attack. Whirlwind. Fly. Sand-attack. Headbutt. Tackle. Take-down. Double-edge. Toxic. Agi
lity. Quick-attack. Rage. Mimic. Double-team. Reflect. Bide. Mirror-move. Swift. Sky-attack. Rest. Substitute. Thief. Snore. Cu
rse. Protect. Feint-attack. Mud-slap. Foresight. Detect. Endure. Swagger. Steel-wing. Attract. Sleep-talk. Return. Frustration.
Pursuit. Hidden-power. Twister. Rain-dance. Sunny-day. Uproar. Heat-wave. Facade. Secret-power. Feather-dance. Air-cutter. Aeri
al-ace. Roost. Natural-gift. Pluck. Tailwind. U-turn. Air-slash. Brave-bird. Defog. Captivate. Ominous-wind. Round. Work-up. Hu
rricane. Confide.
Pokemon name: Pidgeotto.
Its moves are: Razor-wind. Gust. Wing-attack. Whirlwind. Fly. Sand-attack. Headbutt. Tackle. Take-down. Double-edge. Toxic. Agi
lity. Quick-attack. Rage. Mimic. Double-team. Reflect. Bide. Mirror-move. Swift. Sky-attack. Rest. Substitute. Thief. Snore. Cu
rse. Protect. Mud-slap. Detect. Endure. Swagger. Steel-wing. Attract. Sleep-talk. Return. Frustration. Hidden-power. Twister. R
ain-dance. Sunny-day. Uproar. Heat-wave. Facade. Secret-power. Feather-dance. Air-cutter. Aerial-ace. Roost. Natural-gift. Pluc
k. Tailwind. U-turn. Air-slash. Defog. Captivate. Ominous-wind. Round. Work-up. Hurricane. Confide.
Pokemon name: Sandslash.
Its moves are: Scratch. Swords-dance. Cut. Sand-attack. Headbutt. Body-slam. Take-down. Double-edge. Poison-sting. Pin-missile.
Hyper-beam. Submission. Counter. Seismic-toss. Strength. Earthquake. Fissure. Dig. Toxic. Agility. Rage. Mimic. Double-team. De
fense-curl. Bide. Swift. Skull-bash. Amnesia. Leech-life. Fury-swipes. Rest. Rock-slide. Super-fang. Slash. Substitute. Thief.
Snore. Curse. Protect. Mud-slap. Spikes. Detect. Sandstorm. Endure. Rollout. Swagger. Fury-cutter. Attract. Sleep-talk. Return.
Frustration. Safeguard. Magnitude. Dynamic-punch. Rapid-spin. Iron-tail. Hidden-power. Sunny-day. Rock-smash. Facade. Focus-pun
ch. Brick-break. Knock-off. Secret-power. Crush-claw. Rock-tomb. Sand-tomb. Aerial-ace. Mud-shot. Covet. Gyro-ball. Natural-gif
t. Fling. Poison-jab. X-scissor. Focus-blast. Earth-power. Giga-impact. Shadow-claw. Rock-climb. Stone-edge. Captivate. Stealth
-rock. Hone-claws. Round. Bulldoze. Work-up. Drill-run. Confide. Throat-chop. Stomping-tantrum. Steel-roller. Scorching-sands.
Pokemon name: Diglett.
Its moves are: Scratch. Cut. Sand-attack. Headbutt. Body-slam. Take-down. Double-edge. Growl. Earthquake. Fissure. Dig. Toxic.
Agility. Rage. Mimic. Screech. Double-team. Bide. Fury-swipes. Rest. Rock-slide. Slash. Substitute. Thief. Snore. Curse. Revers
al. Protect. Feint-attack. Sludge-bomb. Mud-slap. Sandstorm. Endure. Swagger. Attract. Sleep-talk. Return. Frustration. Magnitu
de. Pursuit. Hidden-power. Sunny-day. Ancient-power. Rock-smash. Beat-up. Uproar. Memento. Facade. Secret-power. Astonish. Rock
-tomb. Aerial-ace. Natural-gift. Assurance. Sucker-punch. Earth-power. Shadow-claw. Mud-bomb. Captivate. Stealth-rock. Hone-cla
ws. Round. Echoed-voice. Ally-switch. Final-gambit. Bulldoze. Work-up. Confide. Stomping-tantrum. Scorching-sands.
Pokemon name: Mankey.
Its moves are: Karate-chop. Mega-punch. Pay-day. Fire-punch. Ice-punch. Thunder-punch. Scratch. Mega-kick. Headbutt. Body-slam.
Take-down. Thrash. Double-edge. Leer. Submission. Low-kick. Counter. Seismic-toss. Strength. Thunderbolt. Thunder. Earthquake.
Dig. Toxic. Meditate. Rage. Mimic. Screech. Double-team. Defense-curl. Focus-energy. Bide. Metronome. Swift. Skull-bash. Fury-s
wipes. Rest. Rock-slide. Substitute. Thief. Snore. Curse. Reversal. Spite. Protect. Mud-slap. Foresight. Detect. Outrage. Endur
e. Swagger. Attract. Sleep-talk. Return. Frustration. Dynamic-punch. Encore. Pursuit. Iron-tail. Hidden-power. Cross-chop. Rain
-dance. Sunny-day. Psych-up. Rock-smash. Beat-up. Uproar. Facade. Focus-punch. Smelling-salts. Taunt. Helping-hand. Role-play.
Revenge. Brick-break. Endeavor. Secret-power. Overheat. Rock-tomb. Aerial-ace. Bulk-up. Covet. Natural-gift. U-turn. Close-comb
at. Payback. Assurance. Fling. Punishment. Poison-jab. Night-slash. Seed-bomb. Vacuum-wave. Focus-blast. Rock-climb. Gunk-shot.
Captivate. Hone-claws. Smack-down. Low-sweep. Round. Acrobatics. Retaliate. Final-gambit. Bulldoze. Work-up. Dual-chop. Confid
e. Power-up-punch. Power-trip. Stomping-tantrum.
```

Question 2 (optional)

Here is a link to a really cool API: https://opentdb.com/
Answer the following questions:
· What is the name of this API?
· What does it do?
· Example URL to make a call to this API?
· Example output?

Answer 2

· What is the name of this API?
It's called "Open Trivia Database"
· What does it do?
It gives the requester specified by them number of trivia questions with answers, categories, incorrect answers, difficulty and type.
· Example URL to make a call to this API?

```
response = requests.get('https://opentdb.com/api.php?amount=3').json()
```

· Example output?

```python
from pprint import pprint as pp
import requests

response = requests.get('https://opentdb.com/api.php?amount=3').json()

pp(response)
```

```
{'response_code': 0,
 'results': [{'category': 'Entertainment: Cartoon & Animations',
              'correct_answer': 'Max Tennyson (Ben 10)',
              'difficulty': 'medium',
              'incorrect_answers': ['Carl Wheezer (Jimmy Neutron)',
                                    'Yakko Warner (Animaniacs)',
                                    'The Mask (The Mask, TV Series)'],
              'question': 'Which one of these cartoon characters is NOT voiced '
                          'by Rob Paulsen?',
              'type': 'multiple'},
             {'category': 'Science: Computers',
              'correct_answer': '23',
              'difficulty': 'medium',
              'incorrect_answers': ['8', '53', '15'],
              'question': 'How many bits make up the significand portion of a '
                          'single precision floating point number?',
              'type': 'multiple'},
             {'category': 'Entertainment: Books',
              'correct_answer': 'Zhao Yun',
              'difficulty': 'easy',
              'incorrect_answers': ['Liu Bei', 'Guan Yu', 'Zhang Fei'],
              'question': 'In Romance of the Three Kingdoms, who was not a '
                          'member of the Peach Garden Oath?',
              'type': 'multiple'}]}
```