

HOMEWORK WEEK 3

(handout for students)

TASK 1 (Conditional flow)

Question 1

Create a program that tells you whether or not you need an umbrella when you leave the house.

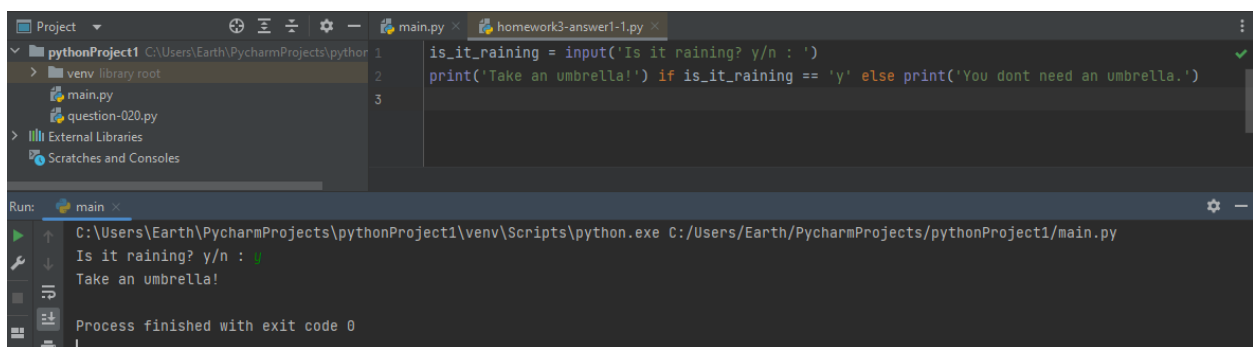
The program should:

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 1.1

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

```
print('Take an umbrella!') if is_it_raining == 'y' else print('You dont need an umbrella.')
```



The screenshot shows a PyCharm IDE with a project named 'pythonProject1'. The file explorer on the left shows a 'venv' directory and a 'main.py' file. The main editor window displays the following Python code:

```
1 is_it_raining = input('Is it raining? y/n : ')
2 print('Take an umbrella!') if is_it_raining == 'y' else print('You dont need an umbrella.')
3
```

The 'Run' window at the bottom shows the execution of the program. The command used is 'C:\Users\Earth\PycharmProjects\pythonProject1\venv\Scripts\python.exe C:\Users\Earth\PycharmProjects\pythonProject1/main.py'. The output shows the prompt 'Is it raining? y/n : ' followed by the user input 'y', and then the output 'Take an umbrella!'. The process finished with exit code 0.

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 1.2

In the name of variable “boat_cost” there is no floor “_” but a space “ ” where the variable is declared and initialised in line 2 of code. Then this variable is attempted to be referenced in line 4, but “boat cost” and “boat_cost” are two different names, so it’s a wrong reference, and also, variable name cannot contain space marks, so it’s a wrong variable name to begin with.

In line 7 there is no closing parentheses after ‘ mark – is “print(‘You cannot afford the boat hire’ “ while it should be “print(‘You cannot afford the boat hire’)”.

In line 6 there is also a space between else and : but that’s a stylistic error, doesn’t affect the code run.

Additionally, apparently the input function in Python isn’t as “type-less” as Python poses to be, because the code won’t run unless the value received in the input is cast as int, i.e. `int(input(‘can we afford the boat?’))` - without indicating the type Python wrongly guesses the value input as str not as int and returns an error at runtime saying logical operator < cannot be used to compare str and int values – it doesn’t recognise 18 or 25 or 36 as int, but as str. That’s why Java is better, it removes the ambiguity of types and doesn’t allow for guessing either on the user side (i.e. “is the value answering the question “is it raining?” a bool or a str?”) nor on the machine side (i.e. like here – input is interpreted as a str even if we input a number).

Lastly, the program contains a logical flaw: if “my_money” is LESS than “boat_cost”, meaning we cannot afford to rent it, then it should print “can’t afford” answer, not “can afford one”. So the correct logic is either:

A - inverted logical operand to produce a logically correct result:

```
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')
```

or B – inverted result of the logic to produce a logically correct result:

```
my_money = int(input('How much money do you have? '))
```

```
boat_cost = 20 + 5
```

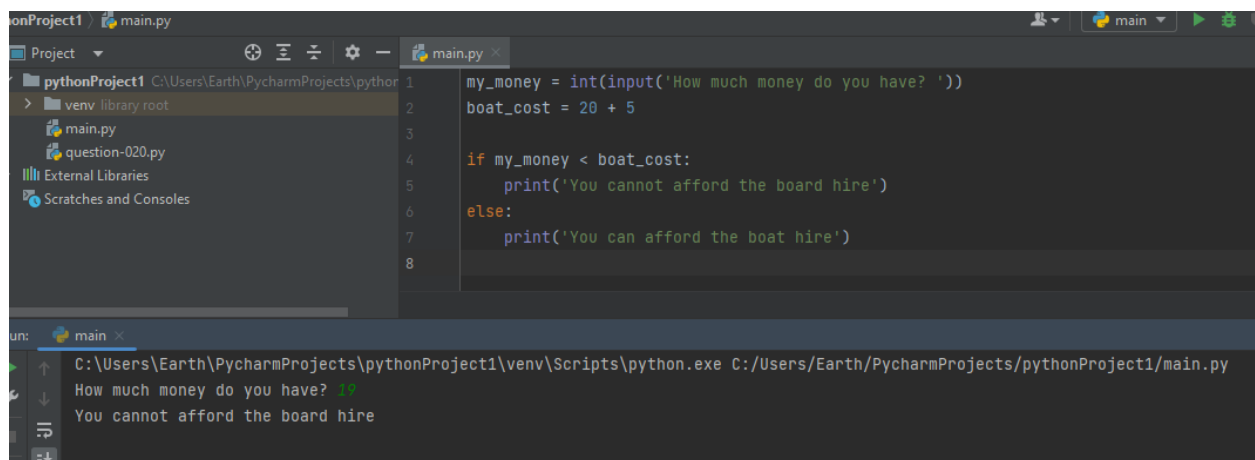
```
if my_money < boat_cost:
```

```
    print('You cannot afford the board hire')
```

else:

```
    print('You can afford the boat hire')
```

Bot A & B will work in the same way – saying “you can’t afford” when “your money” is less than “required money” and saying “you can afford” when “your money” exceeds the “required money”.

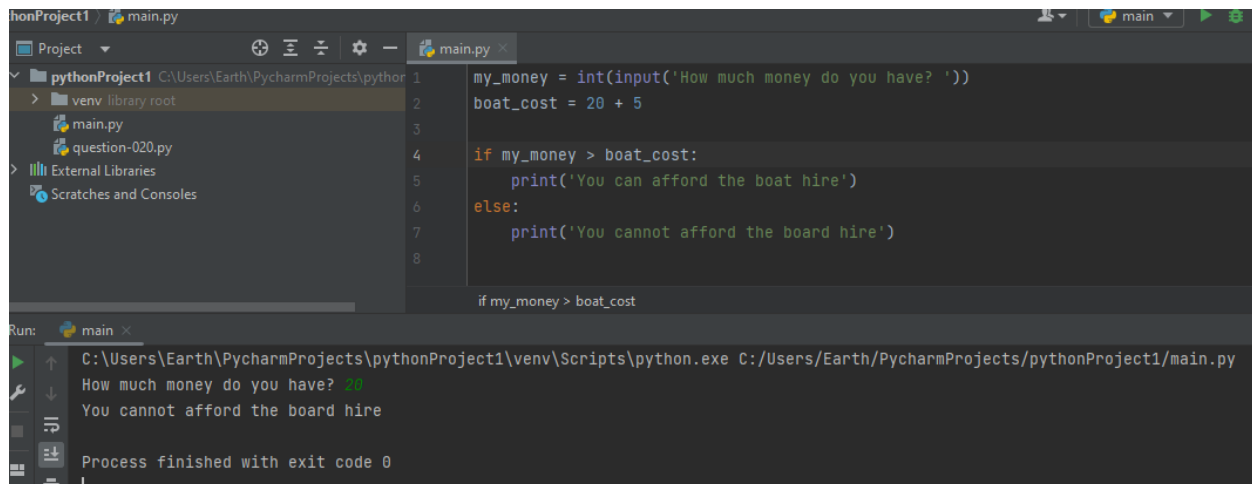


The screenshot shows the PyCharm IDE interface. The top pane displays the code for `main.py`:

```
1 my_money = int(input('How much money do you have? '))
2 boat_cost = 20 + 5
3
4 if my_money < boat_cost:
5     print('You cannot afford the board hire')
6 else:
7     print('You can afford the boat hire')
8
```

The bottom pane shows the console output for the `main` run:

```
run: main
C:\Users\Earth\PycharmProjects\pythonProject1\venv\Scripts\python.exe C:/Users/Earth/PycharmProjects/pythonProject1/main.py
How much money do you have? 17
You cannot afford the board hire
```



The screenshot shows the PyCharm IDE with a project named 'pythonProject1'. The file explorer on the left shows a 'venv' directory and a 'main.py' file. The main editor displays the following Python code:

```
1 my_money = int(input('How much money do you have? '))
2 boat_cost = 20 + 5
3
4 if my_money > boat_cost:
5     print('You can afford the boat hire')
6 else:
7     print('You cannot afford the board hire')
8
9 if my_money > boat_cost
```

The Run window at the bottom shows the execution of the program. The command prompt displays the following output:

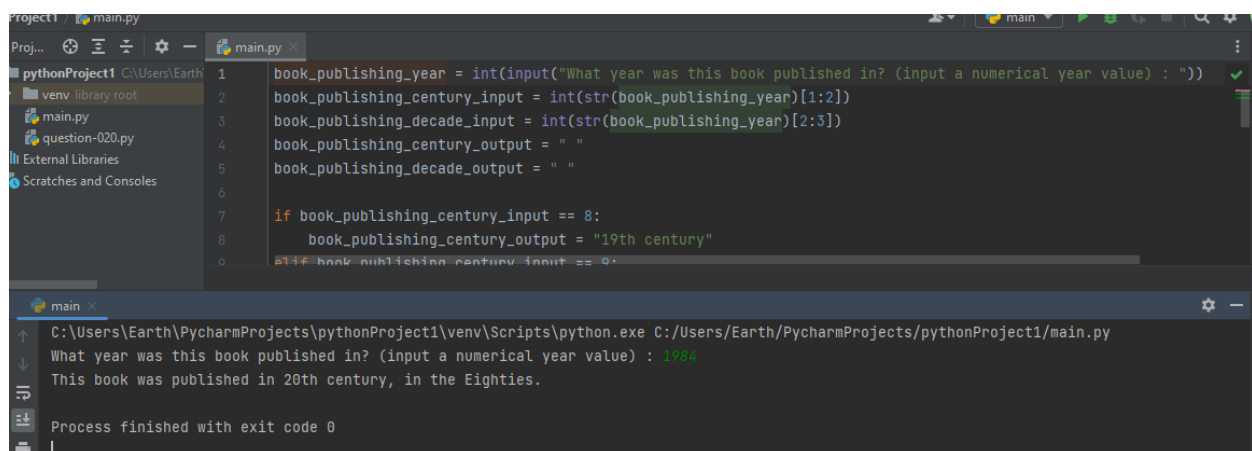
```
C:\Users\Earth\PycharmProjects\pythonProject1\venv\Scripts\python.exe C:\Users\Earth\PycharmProjects\pythonProject1/main.py
How much money do you have? 20
You cannot afford the board hire
Process finished with exit code 0
```

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")

ANSWER 1.3

- 1 – pull out the century & decade determining values (2nd & 3rd digit respectively)
- 2 – name possible centuries and decades by ascribing them string name value
- 3 – put all together into 1 normal language sentence with Python string formatting



The screenshot shows the PyCharm IDE with a project named 'pythonProject1'. The file explorer on the left shows a 'venv' directory and a 'main.py' file. The main editor displays the following Python code:

```
1 book_publishing_year = int(input("What year was this book published in? (input a numerical year value) : "))
2 book_publishing_century_input = int(str(book_publishing_year)[1:2])
3 book_publishing_decade_input = int(str(book_publishing_year)[2:3])
4 book_publishing_century_output = " "
5 book_publishing_decade_output = " "
6
7 if book_publishing_century_input == 8:
8     book_publishing_century_output = "19th century"
9 elif book_publishing_century_input == 9:
```

The Run window at the bottom shows the execution of the program. The command prompt displays the following output:

```
C:\Users\Earth\PycharmProjects\pythonProject1\venv\Scripts\python.exe C:\Users\Earth\PycharmProjects\pythonProject1/main.py
What year was this book published in? (input a numerical year value) : 1984
This book was published in 20th century, in the Eighties.
Process finished with exit code 0
```

```
book_publishing_year = int(input("What year was this book published in? "))
```

```

book_publishing_century_input = int(str(book_publishing_year)[1:2])
book_publishing_decade_input = int(str(book_publishing_year)[2:3])
book_publishing_century_output = " "
book_publishing_decade_output = " "

if book_publishing_century_input == 8:
    book_publishing_century_output = "19th century"
elif book_publishing_century_input == 9:
    book_publishing_century_output = "20th century"
else:
    book_publishing_century_output = "a different century"

match book_publishing_decade_input:
    case 0:
        book_publishing_decade_output = "Aughts"
    case 1:
        book_publishing_decade_output = "Teens"
    case 2:
        book_publishing_decade_output = "Twenties"
    case 3:
        book_publishing_decade_output = "Thirties"
    case 4:
        book_publishing_decade_output = "Forties"
    case 5:
        book_publishing_decade_output = "Fifties"
    case 6:
        book_publishing_decade_output = "Sixties"
    case 7:
        book_publishing_decade_output = "Seventies"
    case 8:
        book_publishing_decade_output = "Eighties"
    case 9:
        book_publishing_decade_output = "Nineties"

print("This book was published in { }, in the
{ }.".format(book_publishing_century_output, book_publishing_decade_output))

```

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 2.1

First of all, we can only guess “what were you expecting” since it isn’t explicitly written, so I cannot be sure what exactly it is that “you were expecting”.

I am guessing that the you expected “oranges” to be the “first thing you need to buy”. In which case, the program is fine, just calling particular objects from the list you should remember that indexing starts with 0, not with 1, so “oranges” are at first place on your list with an index of “0” – if you want to call the 3rd item on your list, namely the “sponge cake”, you should call for the item number 2, as this is the index of the third element on your list. So if you want to get the answer “orange” the correct query is with 0:

```
print(shopping_list[0])
```

The mistake: calling 1 instead of 0 in `print(shopping_list[1])`.

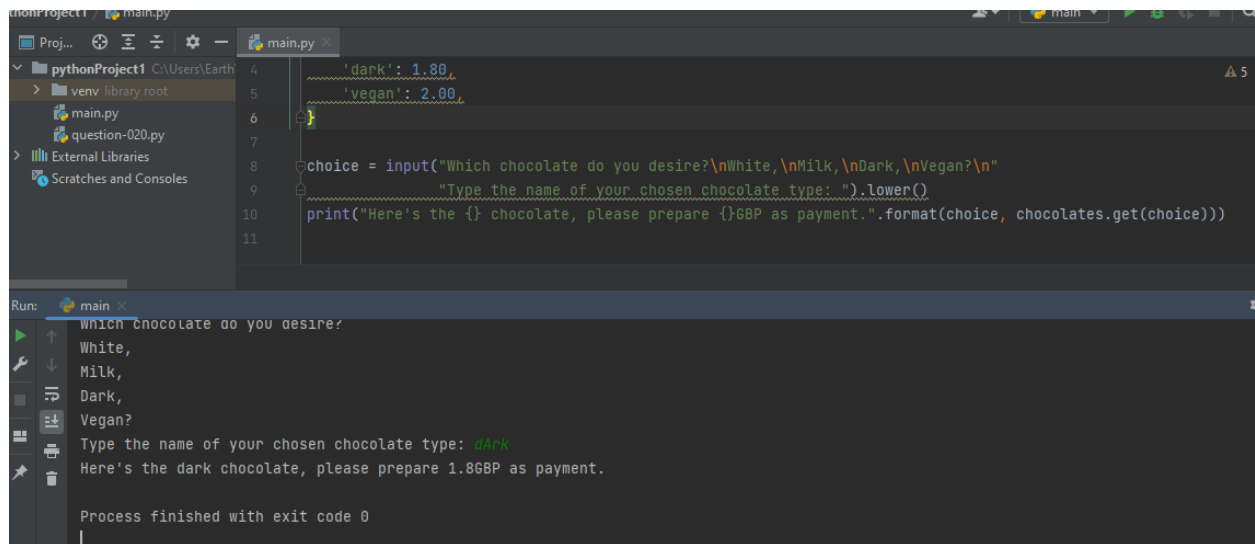
How to fix it: calling 0 instead of 1 in `print(shopping_list[1])` + remember indexing of lists starts at 0 point, so each thing you see is -1, so 3rd thing you see is actually on index 2.

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 2.2



The screenshot shows a Python IDE with a file named `main.py`. The code defines a dictionary `chocolates` with four items: 'white' (1.50), 'milk' (1.20), 'dark' (1.80), and 'vegan' (2.00). It then prompts the user to choose a chocolate type from a list of options. The user has entered 'dark', and the program has output the price of 1.80 GBP. The console output shows the following sequence of events:

```
Which chocolate do you desire?  
White,  
Milk,  
Dark,  
Vegan?  
Type the name of your chosen chocolate type: dark  
Here's the dark chocolate, please prepare 1.80GBP as payment.  
Process finished with exit code 0
```

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

```
choice = input("Which chocolate do you desire?\nWhite,\nMilk,\nDark,\nVegan?\n")
    "Type the name of your chosen chocolate type: ").lower()
print("Here's the {} chocolate, please prepare {}GBP as payment.".format(choice,
chocolates.get(choice)))
```

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 2.3

```
import random
```

```
print("Please choose 7 numbers for your lottery ticket. Hit ENTER key when ready.")
```

```
# declaring and initialising side variables used in lottery ticket generating logic
```

```
which_number = 0
```

```
which_number_ordinal = "
```

```
def match_function(which_number):
```

```
    match which_number:
```

```
        case 0:
```

```
            return '1st'
```

```
        case 1:
```

```
            return '2nd'
```

```
        case 2:
```

```
            return '3rd'
```

```
        case 3:
```



```

        return '4th'
    case 4:
        return '5th'
    case 5:
        return '6th'
    case 6:
        return '7th'

```

```

# # LOTTERY TICKET GENERATING LOGIC - METHOD 1 - random sample of 7
# numbers >100 auto generated with random function
#
# lottery_ticket = random.sample(range(100), 7)

```

```

# LOTTERY TICKET GENERATING LOGIC - METHOD 2 - list of 7 zeros, user
# updates value of each list item by input

```

```

lottery_ticket = [0, 0, 0, 0, 0, 0, 0]
for i in range(len(lottery_ticket)):
    which_number_ordinal = match_function(which_number)
    lottery_ticket[i] = int(input("Please choose your {} lottery ticket number:
".format(which_number_ordinal)))
    which_number += 1

```

```

# # LOTTERY TICKET GENERATING LOGIC - METHOD 3 - list size 0 (empty),
# each user input number appends to the list
#
# lottery_ticket = []
# i = 0
# while i < 7:
#     which_number_ordinal = match_function(which_number)
#     lottery_ticket.append(int(input("Please choose your {} lottery ticket number:
#.format(which_number_ordinal))))
#     which_number += 1
#     i += 1

```

```

print("Your lottery ticket numbers are: ", lottery_ticket)

```

```

# lottery result generating logic (random sample, non repeatable)

```

```

lottery_result = random.sample(range(100), 7)
print("The results of today's draw are: ", lottery_result)

```

```
# lottery match number deciding logic
```

```
number_of_matches = 0
```

```
for i in lottery_ticket:
    for j in lottery_result:
        if j == i:
            number_of_matches += 1
        else:
            continue
```

```
your_prize = 0
```

```
# lottery prize deciding logic
```

```
match number_of_matches:
    case 3:
        your_prize = 20
    case 4:
        your_prize = 40
    case 5:
        your_prize = 100
    case 6:
        your_prize = 10000
    case 7:
        your_prize = 1000000
```

```
# lottery result & prize announcement
```

```
print("You had { } matching numbers and have won the { }GBP prize! Congratulations!"
      .format(number_of_matches, your_prize)) if your_prize > 0 else print(
    "You had no matching numbers and have not won anything at this lottery. Better luck
    next time!")
```

```
# the code works but the rng is cruel, increase lottery result range to 30 results for more
matches -> faster validation
```

```
1 import random
2
3 lottery_ticket = random.sample(range(100), 7)
4 print("Your lottery ticket numbers are: ", lottery_ticket)
5
6 lottery_result = random.sample(range(100), 7)
7 print("The results of today's draw are: ", lottery_result)
8
9 number_of_matches = 0
10
11 for i in lottery_ticket:
12     for j in lottery_result:
13         if j == i:
14             number_of_matches += 1
15         else:
16             continue
17
18 your_prize = 0
19
20 match number_of_matches:
21     case 3:
22         your_prize = 20
23     case 4:
24         your_prize = 40
25     case 5:
26         your_prize = 100
27     case 6:
28         your_prize = 10000
29     case 7:
30         your_prize = 1000000
31
32 print("You have won {}GBP! Congratulations!".format(your_prize)) if your_prize > 0 else print("You have lost")
33
34 # the code works but the rng is cruel, increase lottery result range to 30 results for more matches -> f
35
```

Run: main ×

C:\Users\Earth\PycharmProjects\pythonProject1\venv\Scripts\python.exe C:\Users\Earth\PycharmProjects\pythonProject1/main.py

Your lottery ticket numbers are: [67, 15, 42, 64, 33, 32, 30]

The results of today's draw are: [45, 40, 67, 42, 11, 58, 32]

You have won 20GBP! Congratulations!

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 3.1

Pip means Package Import for Python. Package is a set of files, in Java it's called a module, otherwise known as library / libraries. The point of using

modules/libraries/packages is to save time, make life easier, avoid mistakes other programmers have already encountered, been through and resolved – the module/library/package can be seen as the knowledge gathered from that experience, expressed in code. The files in that library/module/package already are the ready-made answer, solution to a given problem, they allow us to do the thing we want by simply importing the solution which works, instead of trying to invent it by ourselves. They are very useful. Example of a package: a package for Korean language processing, for Korean word segmentation or for encoding Hangeul to Kana and back. If you're writing a program which would require a use of Korean language, importing such a package that aptly enables you to use Hangeul with proper encoding would be useful. After you have imported a package, you can use it in your program (reference it – reference the methods and variables from that package and any of its content). There's a package with advanced or improved math functions – after importing it, you can call on them to use them in your program.

In layman's terms, I'd compare this (a bit) to the Wikipedia – you do not have to have the knowledge (you might not be able to write the code to enable proper Korean language processing or encoding between Hangeul and Kana) but if you know what are you looking for, where to find it, and how to reference it, you can find the thing you want in the Wiki and use it. With packages, you can find the thing you want, import it and use it, i.e. reference/call it in your program. It is enough to understand what you are looking for so to know if the package you found does what you want (is what you are looking for) + how to use it in practice (apply it).

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)
```

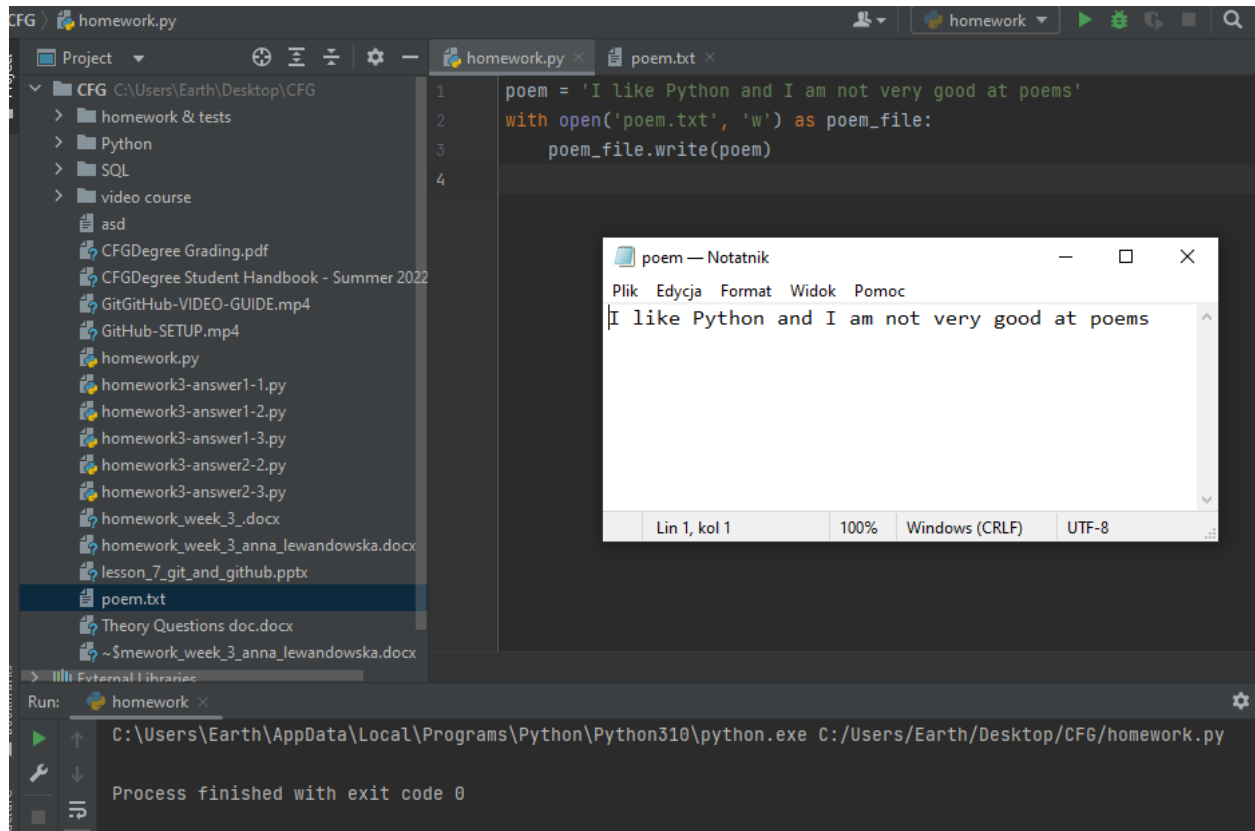
ANSWER 3.2

This code, if run, will produce an error saying “there is no “poem.txt”” if there is no “poem.txt” in the local location.

But the main issue is that it does not write the poem line into the file – this is because with this code, we open the file in “read only” mode – ‘r’ – which means we can read the file's content, but not add to it nor alter it in any way.

If we change the mode to ‘w’ for “write”, then the code works – writes the “poem” line to the “poem.txt” file – even if the file does not exist in the local location, the file with name “poem.txt” will be created and the line “poem” will be written in it.

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



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 3.3

```
song = "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\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\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\n" \
"I'm still standing (Yeah, yeah, yeah)\n" \
"I'm still standing (Yeah, yeah, yeah)"

```

```

with open('./song.txt', 'w') as song_file:
    song_file.write(song)

```

READ ONLY LINES CONTAINING "STILL" - METHOD #1 - List Comprehension

```

with open('./song.txt', 'r') as song_file:
    keyword = 'still'
    matching_lines = [line for line in open('./song.txt').readlines() if keyword in line]
    print(matching_lines)

```

```

## READ ONLY LINES CONTAINING "STILL" - METHOD #2 - For Loop with If
#
# matching_lines = []
# with open('./song.txt', 'r') as song_file:
#     for line in song_file:
#         if 'still' in line:
#             matching_lines.append(line)
# print(matching_lines)

```

```

CFG > homework.py
15 with open('./song.txt', 'w') as song_file:
16     song_file.write(song)
17
18 # READ ONLY LINES CONTAINING "STILL" - METHOD #1 - List Comprehension
19
20
21 with open('./song.txt', 'r') as song_file:
22     keyword = 'still'
23     matching_lines = [line for line in open('./song.txt').readlines() if keyword in line]
24     print(matching_lines)
25
26 ## READ ONLY LINES CONTAINING "STILL" - METHOD #2 - For Loop with If Condition
27 #
28 # matching_lines = []
29 # with open('./song.txt', 'r') as song_file:
30 #     for line in song_file:
31 #         if 'still' in line:
32 #             matching_lines.append(line)
33 # print(matching_lines)
34
with open('./song.txt', 'w') as...
Run: homework
C:\Users\Earth\AppData\Local\Programs\Python\Python310\python.exe C:/Users/Earth/Desktop/CFG/homework.py
["And if you need to know while I'm still standing, you just fade away\n", "Don't you know I'm still standing bette
Process finished with exit code 0

```

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 4.1

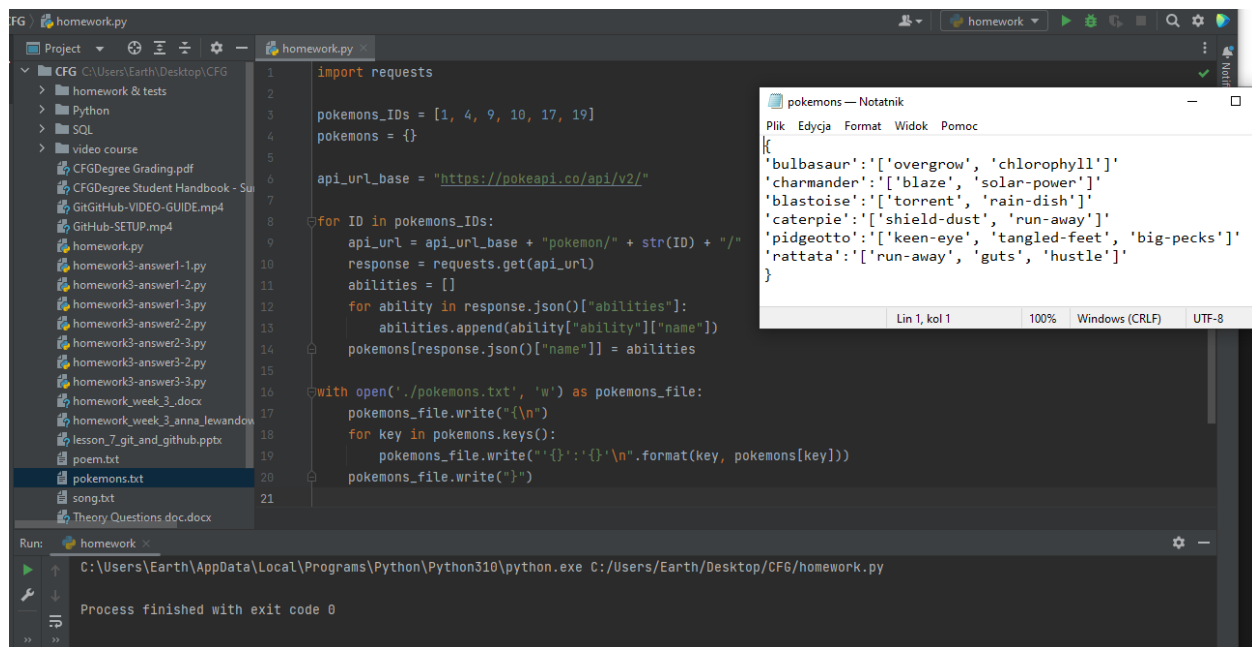
```
import requests
```

```
pokemons_IDs = [1, 4, 9, 10, 17, 19]
pokemons = { }
```

```
api_url_base = "https://pokeapi.co/api/v2/"
```

```
for ID in pokemons_IDs:
    api_url = api_url_base + "pokemon/" + str(ID) + "/"
    response = requests.get(api_url)
    abilities = []
    for ability in response.json()["abilities"]:
        abilities.append(ability["ability"]["name"])
    pokemons[response.json()["name"]] = abilities
```

```
with open('./pokemons.txt', 'w') as pokemons_file:
    pokemons_file.write("{\n")
    for key in pokemons.keys():
        pokemons_file.write("{}': '{}\n".format(key, pokemons[key]))
    pokemons_file.write("}")
```

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?

ANSWER 4.2.1

“The Open Trivia Database provides a completely free JSON API” - so it’s name is “Open Trivia Database API”.

- What does it do?

ANSWER 4.2.2

“use it in your own application to retrieve trivia questions” – meaning this API provides trivia questions to our program. These questions are provided in JSON output format.

- Example URL to make a call to this API?

ANSWER 4.2.3

<https://opentdb.com/api.php?amount=10>

- Example output?

ANSWER 4.2.4

```
{ "response_code":0,"results":[{"category":"Sports","type":"multiple","difficulty":"easy",
"question":"Which team won the 2015-16 English Premier
League?","correct_answer":"Leicester
City","incorrect_answers":["Liverpool","Cheslea","Manchester
United"]},{ "category":"Celebrities","type":"multiple","difficulty":"easy","question":"By
what name is Carlos Estevez better known? ","correct_answer":"Charlie
Sheen","incorrect_answers":["Ricky Martin","Bruno Mars","Joaquin
Phoenix"]},{ "category":"Entertainment:
Television","type":"multiple","difficulty":"easy","question":"In the original Star Trek TV
series, what was Captain James T. Kirk's middle
name?","correct_answer":"Tiberius","incorrect_answers":["Trevor","Travis","Tyrone"]},
{ "category":"Entertainment: Cartoon &
Animations","type":"multiple","difficulty":"easy","question":"When did the last episode
of &quot;Futurama&quot; air?","correct_answer":"September 4,
2013","incorrect_answers":["December 25, 2010","March 28, 1999","On
Going"]},{ "category":"General
Knowledge","type":"multiple","difficulty":"hard","question":"What year was Queen
Elizabeth II
born?","correct_answer":"1926","incorrect_answers":["1923","1929","1930"]},{ "category":"Entertainment: Video Games","type":"multiple","difficulty":"hard","question":"In
World of Warcraft lore, who organized the creation of the
Paladins?","correct_answer":"Alonsus Faol","incorrect_answers":["Uther the
Lightbringer","Alexandros Mograine","Sargeras, The Daemon
Lord"]},{ "category":"Entertainment:
Television","type":"multiple","difficulty":"medium","question":"In the television show
&quot;Lazy Town&quot;, who is the actor of Robbie
Rotten?","correct_answer":"Stefan
Stefansson","incorrect_answers":["Adam Sandler","Magnus
Scheving","Stephen Carl"]},{ "category":"Entertainment: Video
Games","type":"boolean","difficulty":"easy","question":"Several characters in
&quot;Super Mario 64&quot; blink their eyes, including Mario
himself.","correct_answer":"True","incorrect_answers":["False"]},{ "category":"Geograp
hy","type":"multiple","difficulty":"medium","question":"What is the capital of
Lithuania?","correct_answer":"Vilnius","incorrect_answers":["Tallinn","Helsinki","Riga"
]},{ "category":"History","type":"multiple","difficulty":"medium","question":"During
```

which American Civil War campaign did Union troops dig a tunnel beneath Confederate troops to detonate explosives underneath them?", "correct_answer": "Siege of Petersburg", "incorrect_answers": ["Siege of Vicksburg", "Antietam Campaign", "Gettysburg Campaign"]}]}