Dictionaries, Stacks And Queues

# Before Class

1. Familiarise yourself with dictionary data structure. Explain the concepts: key and value.

<https://www.w3schools.com/python/python_dictionaries.asp>

1. Watch the video on using dictionaries in Python:

<https://youtube.com/playlist?list=PLi01XoE8jYohWFPpC17Z-wWhPOSuh8Er->

1. JSON is a lightweight text format for computer data interchange. From the Internet, download any JSON file and open it in a text editor. Pay attention to the description of the data contained in the file. How are they structured?
2. Watch the video on how to deal with JSON files in Python:

<https://youtu.be/pTT7HMqDnJw?feature=shared>

1. A stack is a linear data structure in which data is added to the top of the stack and is retrieved from the top of the stack. Familiarize yourself in more detail with this data structure. Explain the concept of LIFO.
2. A queue is a linear data structure in which new data is added to the end of the queue, and data is retrieved from the beginning of the queue for further processing. Familiarize yourself in more detail with this data structure. Explain the concept of FIFO.

# During Class

## Dictionary

1. Create a dictionary as in the example below. Note the structure of the dictionary (key-value) and the value types in the example below. What type of value was used in each of the six key-value pairs?

person = {  
 "name": "Marek",  
 "surname": "Banach",  
 "age": 25,  
 "hobby": ["swimming","excursions"],  
 "married": True,  
 "phone":{"landline":"123444321","mobile":"777888999"}  
 }

Then, create a program that:

* 1. Displays contents of the dictionary
  2. Displays name
  3. Displays hobby
  4. Changes surname to 'Nowak'
  5. Changes person's marriage status
  6. Adds gender: 'male'
  7. Adds a new hobby: 'bicycle'
  8. Adds work phone to existing phones: '313131444'

1. Create a dictionary describing your mobile phone. Use at least 6 key-value pairs of data of different types. Then, using 'for' loop, display the contents of the dictionary. To read a key and value, use the items() method. Sample result:

mobile = {  
 "OS":"Android",  
 …  
 …  
 …  
 …  
 …  
}  
for key,value in mobile.items():  
 print(f"{key} : {…}")

1. Create an array with 5 dictionaries, each containing a country and its population. Then, using a ‘while’ loop, display the array contents. Sample result:

countries = [  
 {"name":"Poland", "population":38000000},  
 …  
 …  
 …  
 …  
]

COUNTRY POPULATION  
Poland 38000000  
… …  
… …  
… …  
… …

## JSON

1. Find any JSON file on the Internet and download it to your computer. Open the file in any character editor and read its contents. Then, write a program that displays the contents of the JSON file. Sample result:

import json  
  
with open("filename.json") as file:  
 data = json.load(file)  
  
for key,value in data.items():  
 print(f"{key} : {value}")

1. Create a dictionary that describes your favorite book or movie with at least five key-value pairs. Then, create a program that writes the dictionary data to the favourite.json file. Use the dump() method. Pay attention to the formatting of the data in the json file. Use the 'indent' parameter in the dump() method. Sample result:

movie = {  
 "title":"…",  
 "year": …,  
 "actor":{"leading":"…","supporting":"…"},  
 "oscar":False,  
 …  
 …  
 …  
}

1. Write a program in which you create a dictionary containing student data. Try to describe a student in detail, using different data types that can be used in the dictionary. Then, save the data about student in the file student.json, in a readable form.

# After Class

## Dictionary and JSON

1. A program contains any defined dictionary, with any number of key-value pairs of information. Write a program that displays the number of pairs of information available in the dictionary.
2. A dictionary contains course names along with the number of hours. Write a program that calculates and displays the total number of hours. Sample results:

winter\_semester = {  
 "math":60,  
 "programming":30,  
 "history":15  
}

The total number of hours in the winter semester is …

1. The program contains two dictionaries with personal data:

basic\_data = {  
 "name":"Barbara",  
 "age":21  
}  
  
advanced\_data = {  
 "status":"student",  
 "married":False,  
 "interest":["reading","swimming"]  
}

Write a program that creates a dictionary called ‘person’ containing data from the two given dictionaries (five key-value pairs). Display the contents of the ‘person’ dictionary.

1. A program contains two functions:
   1. hotel\_list(hotels) that returns a list of hotels names, separated by a comma sign
   2. avg\_price(hotels) that returns the average room price for a given list of hotels, rounded to an integer value

Write a program that calculates and displays the average price for a room in hotels in Krakow and Sopot and indicates in which cities hotels are cheaper.

Hotels\_in\_Krakow = [  
 {"name":"Sky","price":320.00},  
 {"name":"Metropol","price":480.00},  
 {"name":"New Port","price":420.00},  
 {"name":"Aparthotel","price":390.00}  
]

hotels\_in\_Sopot = [  
 {"name":"Focus","price":510.00},  
 {"name":"Aqua","price":345.00},  
 {"name":"La Boutique","price":390.00},  
 {"name":"Marina","price":410.00}  
]

Sample result:

Hotels in Krakow: …,…,…,…  
Average hotel price in Krakow: …  
Hotels in Sopot: …,…,…,…  
Average hotel price in Sopot: …  
Cheaper hotels in: …

1. Write a program that spells any text entered from the keyboard, using ICAO spelling alphabet:

<https://en.wikipedia.org/wiki/NATO_phonetic_alphabet>

Create a dictionary in which you put all the letters and the corresponding words. Then, try to spell your name and three other words. Sample result:

Enter text: uek  
Spelled text: Uniform Echo Kilo

1. Create a program that writes to a file ICAO.txt the contents of a dictionary containing ICAO spelling alphabet. Sample file content:

A Alfa  
B Bravo  
C Charlie  
D Delta  
…  
…  
Z Zulu

1. Using the website https://mockaroo.com, generate a list of 500 students, containing the following data: name, surname, student ID, gender, age, year of study, email. Write the data to the students.json file. Then, write a program that creates the limited.json file containing the list of students limited to data: first name, last name, student id.
2. The website http://api.nbp.pl contains data on exchange rates published by the National Bank of Poland. The service provides data in json or xml formats. Display the last ten Euro exchange rates in json format in the browser window. Save the data to the euro.json file. Then, write a program that displays the data from the euro.json file in the following format:

Date Buying Rate Selling Rate  
============================================  
2019-10-25 3.8150 3.9820  
... ... ...

1. The products.csv file contains data about the products sold. Create the file in a text editor.

Product,Quantity,Price  
milk,8,4.25  
cheese,5,17.90  
bread,21,6.15  
juice,12,5.90

Then, write a program to convert data from CSV to JSON. The program reads product data from the CSV file and writes the data to a JSON file.

## Stack and Queue

1. The following functions are necessary to handle the stack: push(), pop() and empty(). Below is a simple implementation of the stack using a Python list. Note the definition of the listed functions. What actions do these functions perform? Copy and paste the program code below into a module named stack.py.

#####  
# Stack definition  
##  
  
stack = []  
  
# add value at the top of the stack  
def push(value):  
 stack.append(value)  
   
# remove the topmost element of the stack  
# and return its value   
def pop():  
 if not empty():  
 return stack.pop()  
 else:  
 return None  
   
# return true if the stack is empty  
def empty():  
 return len(stack) == 0  
  
# display stack  
def display():  
 for i in range(len(stack)-1,-1,-1):  
 print(stack[i])  
 print()

1. Write a program, in which, import the module stack.py. Then, do the following:
   1. Display stack
   2. Put the number 2 on the stack
   3. Put the number 14 on the stack
   4. Put the number 9 on the stack
   5. Display stack
   6. Get element from stack
   7. Display stack
   8. Put the number 31 on the stack
   9. Put the number 6 on the stack
   10. Display stack
   11. Get two elements from stack
   12. Display stack
2. Write a program that converts any natural number to a binary number. Use the stack. To convert a number, divide the number by 2, each time taking the remainder of the division and putting the remainder on the stack. Repeat the division until the number you are dividing is zero. Then pop and display all values from the stack. Sample result for number 18:

|  |  |
| --- | --- |
| Division | Remainder |
| 18 / 2 = 9 | 0 |
| 9 / 2 = 4 | 1 |
| 4 / 2 = 2 | 0 |
| 2 / 2 = 1 | 0 |
| 1 / 2 = 0 | 1 |

Natural number: 18  
Binary number: 10010

1. Search the Internet and familiarise yourself with RPN (Reverse Polish Notation). Then, write a program that calculates RPN expressions. RPN can be conveniently evaluated using a stack structure. A user can enter from the keyboard any number, an operator (+ - \* / ) or the equal sign (=).
   1. If the entered value is a number, push the number on to the stack
   2. If the entered value is an operator, pop two items from the top of the stack, do the calculation, and push the result of the operation on to the stack.
   3. If the entered value is an equal sigh, pop the final result from the stack and display the result of calculation.

Using the program, calculate the value of RPN expressions:

|  |  |
| --- | --- |
| Expression | RPN (Reverse Polish Notation) |
| 2 + 3 = | 2 3 + = |
| 2 \* (4 + 1) | 2 4 1 + \* = |
| (2 + 3) \* ( 4 + 5) = | 2 3 + 4 5 + \* = |
| 8 / (3 + 1) \* (3 - 2 + 4) = | 8 3 1 + / 3 2 – 4 + \* = |

1. Following the example of stack.py, create a queue.py module in which define queue handling. Then write a program that imports the queue.py module. Add and remove values from the queue. Display its content.