Handling data in Python

Python Dictionaries

So, far we have seen three in-built data-types in Python that can be used to store data, viz lists, tuples and sets. Each have their own characteristics.

Today, we look at the fourth type. A Python **dictionary**. A dictionary stores data values in key-data pairs. In latest Python version, dictionaries are ordered, changeable and cannot have duplicate keys.

In [5]:
```
import numpy as np
from scipy.integrate import solve_ivp
import matplotlib.pyplot as plt
```

In [33]:
```
new_dict = {
  "Name": ['Rahul','Sourav','Sachin'],  # You can have any data type as v
  "Debut": (1996,1992,1989),
  "Century": np.array([48,38,100])
}

print(new_dict)
```

```
{'Name': ['Rahul', 'Sourav', 'Sachin'], 'Debut': (1996, 1992, 1989), 'Cent
ury': array([ 48,  38, 100])}
```

In [3]:
```
print (len(new_dict))  # the length of the dictionary or number of keys
print (type(new_dict))  # the type of the variable -- is a dictionary
print (type(new_dict["Debut"]))  # the type of the variable -- is a tuple
print (type(new_dict["Century"]))  # the type of the variable -- is a num

new_dict.keys()  # gives the keys of the dictionary
```

```
3
<class 'dict'>
<class 'tuple'>
<class 'numpy.ndarray'>
```

Out[3]:
```
dict_keys(['Name', 'Debut', 'Century'])
```

Accessing and adding elements and keys in a dictionary

In [25]:
```
print (new_dict.get("Name"))
print (new_dict.get("Name")[1])
print ()

new_dict["Name"][1] = 'Ganguly'  # Edit a specific value in a specific ke
print(new_dict["Name"])
```

```
['Rahul', 'Sourav', 'Sachin']
Sourav

['Rahul', 'Ganguly', 'Sachin']
```

```
In [26]: new_dict["Wicket"] = [5,132,200]  # Add a new key
         print(new_dict)
```

```
{'Name': ['Rahul', 'Ganguly', 'Sachin'], 'Debut': (1996, 1992, 1989), 'Cen
tury': array([ 48,  38, 100]), 'Wicket': [5, 132, 200]}
```

```
In [4]: # Adding new elements to each list
        new_dict["Debut"].append(1996)
        # new_dict["Name"].append('Laxman')
        # new_dict["Century"].append(23)
        # new_dict["Wicket"].append(2)
```

```
---------------------------------------------------------------------------
-
AttributeError                            Traceback (most recent call las
t)
Cell In[4], line 2
      1 # Adding new elements to each list
----> 2 new_dict["Debut"].append(1996)

AttributeError: 'tuple' object has no attribute 'append'
```

```
In [28]: new_dict
```

```
Out[28]: {'Name': ['Rahul', 'Ganguly', 'Sachin'],
          'Debut': (1996, 1992, 1989),
          'Century': array([ 48,  38, 100]),
          'Wicket': [5, 132, 200]}
```

```
In [29]: new_dict.items() # Gives you the key and value pair as a tuple
```

```
Out[29]: dict_items([('Name', ['Rahul', 'Ganguly', 'Sachin']), ('Debut', (1996, 1
         992, 1989)), ('Century', array([ 48,  38, 100])), ('Wicket', [5, 132, 20
         0])])
```

Practise problem

The spread of the viral infection in a body with an initial infection is approximated with balance equations on the number of healthy cells (H), infected cells (I), and virus count (V), which are governed by:

$$\frac{dH}{dt} = r_1 - r_2 H - r_3 HV$$
$$\frac{dV}{dt} = -r_3 HV - r_4 V + r_5 I,$$
$$\frac{dI}{dt} = r_3 HV - r_6 I$$

where, $r_1 = 10^5$ is the growth rate of healthy cells, $r_2 = 0.1$ is the death rate, $r_3 = 2 * 10^{-7}$ is the rate of conversion of healthy cells into infected cells, $r_4 = 5$ is the death rate of virus, $r_5 = 100$ is the production of virus by infected cells, and $r_6 = 0.5$ is the death rate of infected cells. All rates are per month.

Plot the healthy cell, infected cell, and the virus count over the course of 15 months, if the initial counts are: $H(0) = 10^6$, $V(0) = 100$, and $I(0) = 0$.

Create a dictionary: dict = {'time':[],'healthy cells':[], 'virus count':[], 'infected cells':[] }, and store the output of the differential equation in the dictionary.

Saving data in Python

We look at ways in which data can be saved in Python. There are 3 types we will discuss 1) simple **.txt** for simple text format, 2) **csv** file for data that can be written as a table (think MS Excel), and 3) using **pickle**, **NumPy** and **json** for saving objects, dictionaries or more structured data.

Using txt

In [11]:
```
## Saving text data

txtdata = ['Axe','Bat','Cod','Dash']         # Here is the text data
file = open("filename.txt", "w")    # Creating a file -- "w" stands for w

for i in txtdata:
    file.writelines([i,'\n'])            # Writing text -- use "write" if
file.close()
```

In [14]:
```
file = open("filename.txt", "r")
print (file.read())
file.close()
```

```
Axe
Bat
Cod
Dash
Elephant
```

In [13]:
```
file = open("filename.txt", "a")
file.write('Elephant')
file.close()
```

Using CSV

CSV stands for comma separated values. Works well with pandas.

In [20]:
```
import csv

heads = ['Name', 'Roll', 'Marks']

data = [['Arup', '090', '88'],
         ['Lata', '112', '95'],
         ['Varun', '202', '92'],
       ]

file = "midsems.csv"

# writing to csv file
with open(file, 'w') as csvfile:
    csvwriter = csv.writer(csvfile)

    csvwriter.writerow(heads)
    csvwriter.writerows(data)
```

In [23]:
```
# reading a csv file

with open("midsems.csv", mode='r')as file:
    f = csv.reader(file)
    for rows in f:
        print (rows)
```

```
['Name', 'Roll', 'Marks']
['Arup', '090', '88']
['Lata', '112', '95']
['Varun', '202', '92']
['Raj', '212', '100']
```

In [22]:
```
with open("midsems.csv", 'a') as csvfile:
    csvwriter = csv.writer(csvfile)

    csvwriter.writerow(['Raj', '212', '100'])
```

Saving data using pickle

The pickle module implements binary protocols for serializing and deserializing a Python object structure. “Pickling” is the process whereby a Python object hierarchy is converted into a byte stream, and “unpickling” is the inverse operation, whereby a byte stream (from a binary file or bytes-like object) is converted back into an object hierarchy.

In [24]:
```
import pickle
data_np = np.arange(100000)
```

In [25]:
```
# Create the pickle file

with open("numpy_data.pickle", "wb") as f:
    pickle.dump(data_np, f, protocol=pickle.HIGHEST_PROTOCOL)
```

In [13]:
```
# Read from the pickle file

with open("numpy_data.pickle", "rb") as f:
    out = pickle.load(f)
out
```

Out[13]:
```
array([    0,     1,     2, ..., 99997, 99998, 99999])
```

In [104…
```
with open("dict.pickle", "wb") as f1:
    pickle.dump(new_dict, f1, protocol=pickle.HIGHEST_PROTOCOL)
```

In [110…
```
file = open("dict.pickle", "rb")
pickle.load(file)
# print (pickle.load(file))
# file.close()
```

Out[110…
```
{'Name': ['Dravid', 'Ganguly', 'Tendulkar'],
 'Debut': (1996, 1992, 1989),
 'Century': [48, 38, 100],
 'Wicket': [5, 132, 200]}
```

Practise

From the above problem, where one created a dictionary: dict = {'time':[],'healthy cells':[], 'virus count':[], 'infected cells':[]}, to store the output of the differential equation in the dictionary. Save it as both a pickle and numpy file.

Load the pickle file that you have saved and, using matplotlib, plot the change of I, H and V with time.

Saving Python data and dictionaries

using NumPy

In [29]:
```
string = np.arange(10000)
np.savetxt('strings_numpy.txt', string)
```

In [30]:
```
string2 = np.loadtxt('strings_numpy.txt')
string2
```

Out[30]:
```
array([0.000e+00, 1.000e+00, 2.000e+00, ..., 9.997e+03, 9.998e+03,
       9.999e+03], shape=(10000,))
```

In [31]:
```
np.save('numpy_data.npy',data_np,allow_pickle = True)
```

In [34]:
```
np.save('cricket.npy',new_dict,allow_pickle = True) # The pickle module i
                                                    # serializing and de-

new_dict["Name"][0] = 'Dravid'  # Edit a specific value in a specific key
new_dict["Name"][2] = 'Tendulkar'  # Edit a specific value in a specific

print(new_dict["Name"])
```

```
['Dravid', 'Sourav', 'Tendulkar']
```

In [35]:
```
old_dict = np.load('cricket.npy',allow_pickle = True).item(0) # set allow
```

In [36]:
```
print (old_dict)
print ()
print (new_dict)
```

```
{'Name': ['Rahul', 'Sourav', 'Sachin'], 'Debut': (1996, 1992, 1989), 'Cent
ury': array([ 48,  38, 100])}

{'Name': ['Dravid', 'Sourav', 'Tendulkar'], 'Debut': (1996, 1992, 1989),
'Century': array([ 48,  38, 100])}
```

In [38]:
```
np.save('cricket.npy',new_dict,allow_pickle = True) # will overwrite the
check_dict = np.load('cricket.npy',allow_pickle = True).item(0) # set all
print (check_dict)
```

```
{'Name': ['Dravid', 'Sourav', 'Tendulkar'], 'Debut': (1996, 1992, 1989),
'Century': array([ 48,  38, 100])}
```

using json

What is json? JavaScript Object Notation (JSON) is a standard text-based format for representing structured data based on JavaScript object syntax but applicable to other platforms. It is commonly used for transmitting data in various applications (e.g., sending some data from the server to the client or vice versa).

In [40]:
```
import pickle
import json
```

In [45]:
```
# Load this file from last class -- attached with code

new_dict = np.load('cricket.npy',allow_pickle = True).item(0)

new_dict
```

Out[45]:
```
{'Name': ['Dravid', 'Sourav', 'Tendulkar'],
 'Debut': (1996, 1992, 1989),
 'Century': array([ 48,  38, 100])}
```

In [46]:
```
with open("dict.json", "w") as outfile:        # Open a new json file, fo
    json.dump(new_dict,outfile,indent=4)       # Dump the new_dict dictio

new_dict
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[46], line 2
      1 with open("dict.json", "w") as outfile:        # Open a new json file, for writing "w"
----> 2     json.dump(new_dict,outfile,indent=4)       # Dump the new_dict dictionary to the open json file
      4 new_dict

File /opt/anaconda3/envs/Py3/lib/python3.13/json/__init__.py:179, in dump(obj, fp, skipkeys, ensure_ascii, check_circular, allow_nan, cls, indent, separators, default, sort_keys, **kw)
    173     iterable = cls(skipkeys=skipkeys, ensure_ascii=ensure_ascii,
    174         check_circular=check_circular, allow_nan=allow_nan, indent=indent,
    175         separators=separators,
    176         default=default, sort_keys=sort_keys, **kw).iterencode(obj)
    177 # could accelerate with writelines in some versions of Python, at
    178 # a debuggability cost
--> 179 for chunk in iterable:
    180     fp.write(chunk)

File /opt/anaconda3/envs/Py3/lib/python3.13/json/encoder.py:433, in _make_iterencode.<locals>._iterencode(o, _current_indent_level)
    431     yield from _iterencode_list(o, _current_indent_level)
    432 elif isinstance(o, dict):
--> 433     yield from _iterencode_dict(o, _current_indent_level)
    434 else:
    435     if markers is not None:

File /opt/anaconda3/envs/Py3/lib/python3.13/json/encoder.py:407, in _make_iterencode.<locals>._iterencode_dict(dct, _current_indent_level)
    405         else:
    406             chunks = _iterencode(value, _current_indent_level)
--> 407         yield from chunks
    408 if not first and newline_indent is not None:
    409     _current_indent_level -= 1

File /opt/anaconda3/envs/Py3/lib/python3.13/json/encoder.py:440, in _make_iterencode.<locals>._iterencode(o, _current_indent_level)
    438         raise ValueError("Circular reference detected")
    439     markers[markerid] = o
--> 440 o = _default(o)
    441 yield from _iterencode(o, _current_indent_level)
    442 if markers is not None:

File /opt/anaconda3/envs/Py3/lib/python3.13/json/encoder.py:180, in JSONEncoder.default(self, o)
    161 def default(self, o):
    162     """Implement this method in a subclass such that it returns
    163     a serializable object for ``o``, or calls the base implementation
    164     (to raise a ``TypeError``).
   (...)    178
    179     """
--> 180     raise TypeError(f'Object of type {o.__class__.__name__} '
    181                     f'is not JSON serializable')
```

```
TypeError: Object of type ndarray is not JSON serializable
```

In [47]:
```
# nd.arrays are not "json" serializable

new_dict["Century"]= new_dict["Century"].tolist()   # Convert it into a l
print (new_dict)
```

```
{'Name': ['Dravid', 'Sourav', 'Tendulkar'], 'Debut': (1996, 1992, 1989),
'Century': [48, 38, 100]}
```

In [48]:
```
with open("dict.json", "w") as outfile:             # Open a new json file
    json.dump(new_dict,outfile,indent=4)            # Dump the new_dict di
```

In [49]:
```
loadfile = open("dict.json")                     # Open a saved json file
json_dict = json.load(loadfile)                  # Dump the opened a file to a v
json_dict
```

Out[49]:
```
{'Name': ['Dravid', 'Sourav', 'Tendulkar'],
 'Debut': [1996, 1992, 1989],
 'Century': [48, 38, 100]}
```

In [50]:
```
json_dict["City"] = ['Bengaluru','Kolkata','Mumbai']

# json_dict
with open("dict.json","w") as loadfile:
    json.dump(json_dict,loadfile,indent=4)
```

In [51]:
```
with open("dict.json") as loadfile:                  # Open a saved json file
    json_dict = json.load(loadfile)                  # Dump the opened a file
json_dict
```

Out[51]:
```
{'Name': ['Dravid', 'Sourav', 'Tendulkar'],
 'Debut': [1996, 1992, 1989],
 'Century': [48, 38, 100],
 'City': ['Bengaluru', 'Kolkata', 'Mumbai']}
```

In [55]:
```
students = {
            "2022": [{'Name':'Ajay',
                      'Age':18,
                      'Roll':'22ND075'},
                     {'Name':'Vijay',
                      'Age':19,
                      'Roll':'22ND867'},
                     {'Name':'Tanuj',
                      'Age':19,
                      'Roll':'22ND105'}
                    ]
}

with open("student.json", "w") as outfile:   # Create a new json file "w"
    json.dump(students,outfile,indent=2)

students['2021']=[]
print (students)

with open("student.json", "w") as outfile:   # If I want to overwrite the
    json.dump(students,outfile,indent=2)

def append_jsonfile(add_key,add_value,filename='stud.json'):
    with open(filename, "r+") as outfile:   # Open the json file, for rea
        dict_ = json.load(outfile)
        dict_[add_key].append(add_value)
    with open(filename, "w") as outfile:
        json.dump(dict_,outfile,indent=2)

add_val = {'Name':'Alex',
           'Age': 21,
           'Roll':'21ND005'}

add_val2 = {'Name':'Raj',
           'Age': 24,
           'Roll':'21ND105'}

append_jsonfile('2021',add_val,'student.json')

append_jsonfile('2021',add_val2,'student.json')

with open("student.json") as outfile:   # If I want to overwrite then alw
    new_student = json.load(outfile)

new_student
```

```
{'2022': [{'Name': 'Ajay', 'Age': 18, 'Roll': '22ND075'}, {'Name': 'Vija
y', 'Age': 19, 'Roll': '22ND867'}, {'Name': 'Tanuj', 'Age': 19, 'Roll': '2
2ND105'}], '2021': []}
```

Out[55]:
```
{'2022': [{'Name': 'Ajay', 'Age': 18, 'Roll': '22ND075'},
  {'Name': 'Vijay', 'Age': 19, 'Roll': '22ND867'},
  {'Name': 'Tanuj', 'Age': 19, 'Roll': '22ND105'}],
 '2021': [{'Name': 'Alex', 'Age': 21, 'Roll': '21ND005'},
  {'Name': 'Raj', 'Age': 24, 'Roll': '21ND105'}]}
```

Using pandas

Pandas is a Python library that provides various data structures and operations for manipulating numerical data. Built on top of the NumPy library, Pandas is fast, productive and high performing.

https://www.geeksforgeeks.org/introduction-to-pandas-in-python/

In [56]:
```
import pandas as pd

list0 = (students['2022'][1])
print (list0,'\n')

print("This is a Panda series,\n")
A = pd.Series(list0)  # Converting the above list to a Panda series
print(A)
```

```
{'Name': 'Vijay', 'Age': 19, 'Roll': '22ND867'}

This is a Panda series,

Name      Vijay
Age          19
Roll    22ND867
dtype: object
```

```
In [62]: list_2022 = pd.DataFrame(new_student['2022'])  # Converting our dictionar
         print("This is a Panda Dataframe,\n")
         print (list_2022)

         # print(list_2022[['Name','Age']])
```

```
This is a Panda Dataframe,

    Name  Age     Roll
0   Ajay   18  22ND075
1  Vijay   19  22ND867
2  Tanuj   19  22ND105
```

```
In [76]: large_data = pd.read_csv("ipl_data.csv")  # Loading a large data set
         ipl = pd.DataFrame(large_data)  # Converting our dictionary object to a P
         print("This is a Panda Dataframe,\n")
         print (ipl)
```

This is a Panda Dataframe,

```
      id  season        city        date                  team1  \
0      1    2008   Bangalore  2008-04-18  Kolkata Knight Riders
1      2    2008  Chandigarh  2008-04-19    Chennai Super Kings
2      3    2008       Delhi  2008-04-19       Rajasthan Royals
3      4    2008      Mumbai  2008-04-20         Mumbai Indians
4      5    2008     Kolkata  2008-04-20        Deccan Chargers
..   ...     ...         ...         ...                    ...
572  573    2016      Raipur  2016-05-22       Delhi Daredevils
573  574    2016   Bangalore  2016-05-24          Gujarat Lions
574  575    2016       Delhi  2016-05-25    Sunrisers Hyderabad
575  576    2016       Delhi  2016-05-27          Gujarat Lions
576  577    2016   Bangalore  2016-05-29    Sunrisers Hyderabad

                           team2                  toss_winner toss_decisio
n  \
0    Royal Challengers Bangalore  Royal Challengers Bangalore         fiel
d
1                Kings XI Punjab          Chennai Super Kings           ba
t
2               Delhi Daredevils             Rajasthan Royals           ba
t
3    Royal Challengers Bangalore               Mumbai Indians           ba
t
4          Kolkata Knight Riders              Deccan Chargers           ba
t
..                           ...                          ...
...
572  Royal Challengers Bangalore  Royal Challengers Bangalore         fiel
d
573  Royal Challengers Bangalore  Royal Challengers Bangalore         fiel
d
574        Kolkata Knight Riders        Kolkata Knight Riders         fiel
d
575          Sunrisers Hyderabad          Sunrisers Hyderabad         fiel
d
576  Royal Challengers Bangalore          Sunrisers Hyderabad           ba
t

     result  dl_applied                       winner  win_by_runs  \
0    normal           0        Kolkata Knight Riders          140
1    normal           0          Chennai Super Kings           33
2    normal           0             Delhi Daredevils            0
3    normal           0  Royal Challengers Bangalore            0
4    normal           0        Kolkata Knight Riders            0
..      ...         ...                          ...          ...
572  normal           0  Royal Challengers Bangalore            0
573  normal           0  Royal Challengers Bangalore            0
574  normal           0          Sunrisers Hyderabad           22
575  normal           0          Sunrisers Hyderabad            0
576  normal           0          Sunrisers Hyderabad            8

     win_by_wickets player_of_match  \
0                 0     BB McCullum
1                 0      MEK Hussey
2                 9     MF Maharoof
3                 5      MV Boucher
4                 5       DJ Hussey
..              ...             ...
```

```
572                6         V Kohli
573                4  AB de Villiers
574                0    MC Henriques
575                4       DA Warner
576                0     BCJ Cutting

                                                venue         umpire1  \
0                               M Chinnaswamy Stadium       Asad Rauf
1          Punjab Cricket Association Stadium, Mohali       MR Benson
2                                    Feroz Shah Kotla       Aleem Dar
3                                    Wankhede Stadium        SJ Davis
4                                        Eden Gardens       BF Bowden
..                                                ...             ...
572  Shaheed Veer Narayan Singh International Stadium   A Nand Kishore
573                             M Chinnaswamy Stadium     AK Chaudhary
574                                  Feroz Shah Kotla       M Erasmus
575                                  Feroz Shah Kotla       M Erasmus
576                             M Chinnaswamy Stadium  HDPK Dharmasena

             umpire2  umpire3
0        RE Koertzen      NaN
1         SL Shastri      NaN
2      GA Pratapkumar     NaN
3          DJ Harper      NaN
4        K Hariharan      NaN
..               ...      ...
572      BNJ Oxenford     NaN
573  HDPK Dharmasena      NaN
574     C Shamshuddin     NaN
575         CK Nandan     NaN
576      BNJ Oxenford     NaN

[577 rows x 18 columns]
```

```
In [77]: print (large_data[['season','umpire1']])
```

```
     season          umpire1
0      2008        Asad Rauf
1      2008        MR Benson
2      2008        Aleem Dar
3      2008         SJ Davis
4      2008        BF Bowden
..      ...              ...
572    2016   A Nand Kishore
573    2016     AK Chaudhary
574    2016        M Erasmus
575    2016        M Erasmus
576    2016  HDPK Dharmasena

[577 rows x 2 columns]
```

```
In [70]: print (large_data.loc[list(range(20,25)),['toss_winner','winner']])
```

```
            toss_winner                       winner
20      Deccan Chargers  Royal Challengers Bangalore
21      Kings XI Punjab              Kings XI Punjab
22     Delhi Daredevils               Mumbai Indians
23  Chennai Super Kings             Rajasthan Royals
24      Kings XI Punjab              Kings XI Punjab
```

```
In [ ]: count = 0
        for i,j in large_data.iterrows():
            if j['city'] == 'Kolkata':
                print(j['id'],j['winner'])
                count += 1
                print()
        print ("Games held in Kolkata: ",count)
```

```
In [74]: small_data = pd.read_csv("ipl_data.csv",index_col='season')  # Loading a
```

```
In [75]: print (small_data.loc[[2008],['winner','player_of_match']])
```

```
                             winner   player_of_match
season                                               
2008          Kolkata Knight Riders       BB McCullum
2008            Chennai Super Kings        MEK Hussey
2008               Delhi Daredevils       MF Maharoof
2008    Royal Challengers Bangalore        MV Boucher
2008          Kolkata Knight Riders         DJ Hussey
2008               Rajasthan Royals         SR Watson
2008               Delhi Daredevils          V Sehwag
2008            Chennai Super Kings         ML Hayden
2008               Rajasthan Royals         YK Pathan
2008                Kings XI Punjab     KC Sangakkara
2008               Rajasthan Royals         SR Watson
2008            Chennai Super Kings          JDP Oram
2008                Deccan Chargers      AC Gilchrist
2008                Kings XI Punjab         SM Katich
2008            Chennai Super Kings          MS Dhoni
2008                 Mumbai Indians     ST Jayasuriya
2008               Delhi Daredevils        GD McGrath
2008                Kings XI Punjab          SE Marsh
2008               Rajasthan Royals       SA Asnodkar
2008               Delhi Daredevils          V Sehwag
2008    Royal Challengers Bangalore     R Vinay Kumar
2008                Kings XI Punjab         IK Pathan
2008                 Mumbai Indians        SM Pollock
2008               Rajasthan Royals     Sohail Tanvir
2008                Kings XI Punjab       S Sreesanth
2008                Deccan Chargers      AC Gilchrist
2008                 Mumbai Indians           A Nehra
2008            Chennai Super Kings          MS Dhoni
2008          Kolkata Knight Riders        SC Ganguly
2008               Rajasthan Royals         YK Pathan
2008                 Mumbai Indians      CRD Fernando
2008            Chennai Super Kings          L Balaji
2008          Kolkata Knight Riders        SC Ganguly
2008               Rajasthan Royals         SR Watson
2008                Kings XI Punjab          SE Marsh
2008          Kolkata Knight Riders     Shoaib Akhtar
2008                 Mumbai Indians     ST Jayasuriya
2008                Kings XI Punjab          SE Marsh
2008               Delhi Daredevils          A Mishra
2008                 Mumbai Indians        SM Pollock
2008                Kings XI Punjab  DPMD Jayawardene
2008               Rajasthan Royals          GC Smith
2008                 Mumbai Indians          DJ Bravo
2008            Chennai Super Kings           M Ntini
2008               Delhi Daredevils        SP Goswami
2008               Rajasthan Royals         YK Pathan
2008                Kings XI Punjab          SE Marsh
2008    Royal Challengers Bangalore          A Kumble
2008                Kings XI Punjab          SE Marsh
2008               Delhi Daredevils        KD Karthik
2008               Rajasthan Royals         JA Morkel
2008    Royal Challengers Bangalore           P Kumar
2008          Kolkata Knight Riders          Umar Gul
2008               Rajasthan Royals     Sohail Tanvir
2008            Chennai Super Kings          SK Raina
2008               Rajasthan Royals         SR Watson
2008            Chennai Super Kings           M Ntini
2008               Rajasthan Royals         YK Pathan
```

Tasks for today

Solve the following problems.

- Solve the differential equation:

  $$\frac{dx}{dt} = x(t) + 1$$

  $$\frac{dy}{dt} = -\frac{1}{5}(y(t) - x(t)),$$

  where, $x(0) = y(0) = 0$ and $t \in [0, 10]$.

  Create a dictionary: dict = {'time':[],'x_value':[], 'y_value':[]}, and store the output of the differential equation in the dictionary. Save it as both a pickle and numpy file.

- Load the pickle file that you have saved and, using matplotlib, plot x vs t and y vs t, on the same plot.

- Create a dictionary containing the name of your five of your friends, their city of birth, hometown, and a fictional passport number.

- Now write a function that can add a new friend's information. Add the Head as the new friend, {'HOD','Kolkata','Mumbai','XYZ789'}.

- Create a dictionary containing the name of your five of your friends, their city of birth, hometown, and a fictional passport number.

  Save the above dictionary a json file and create a function, that can upload the file and add a new friend's information, and then save the json file. Add the Head as the new friend, {'SDhar','Kolkata','Mumbai','XYZ789'}. Upload the file again and display the dictionary as a dataframe using pandas.

- Upload the file "ipl_data.csv" and count the number of matches won by "Chennai Super Kings" in each season.

  From the above file "ipl_data.csv" and count the number of matches where the umpire was 'DJ Harper'.

```
In [ ]:
```