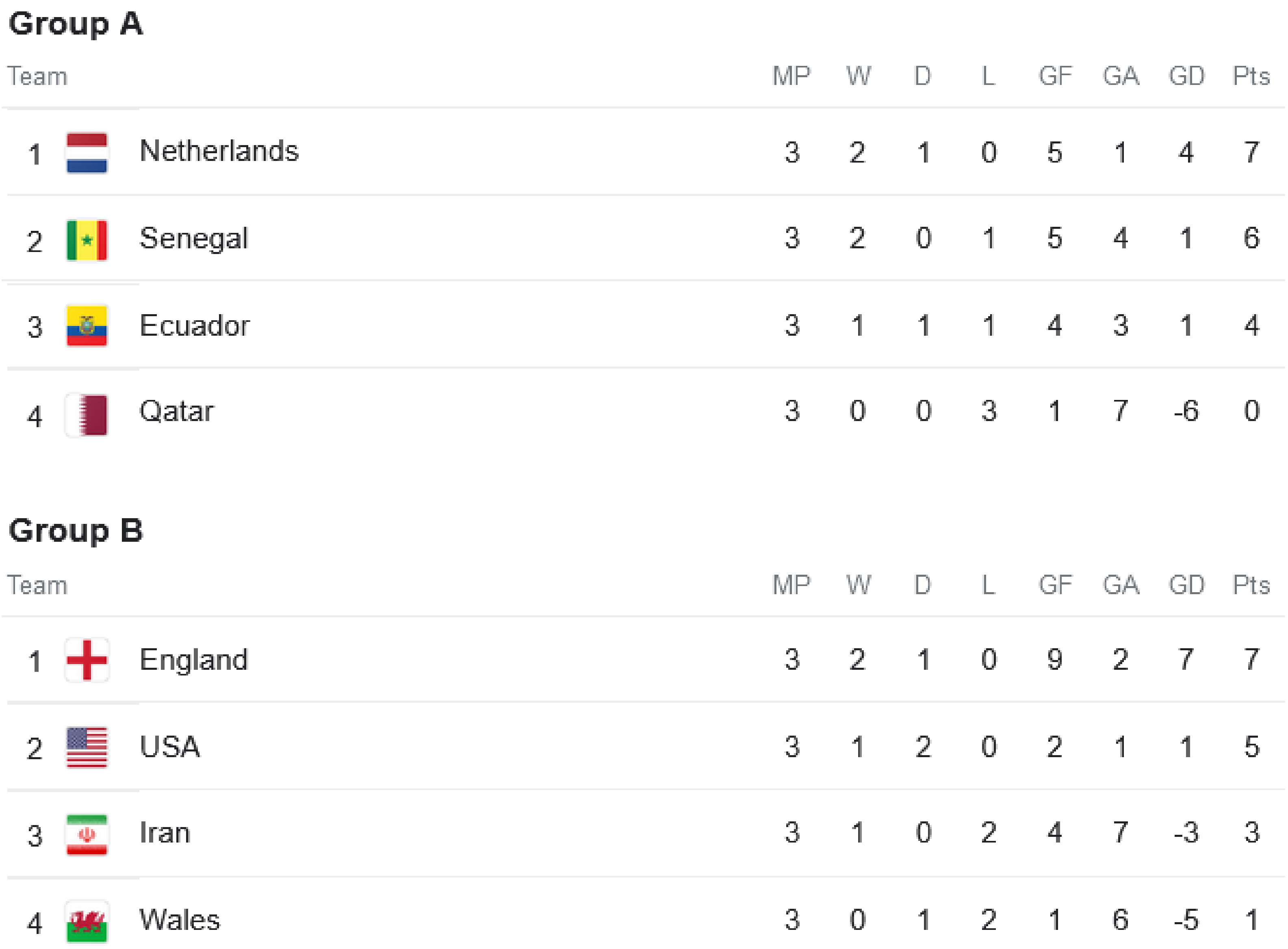
Lecture 29
Serialization - Pickle

OOP – Spring 2022 (Python)

Points Table



Group A

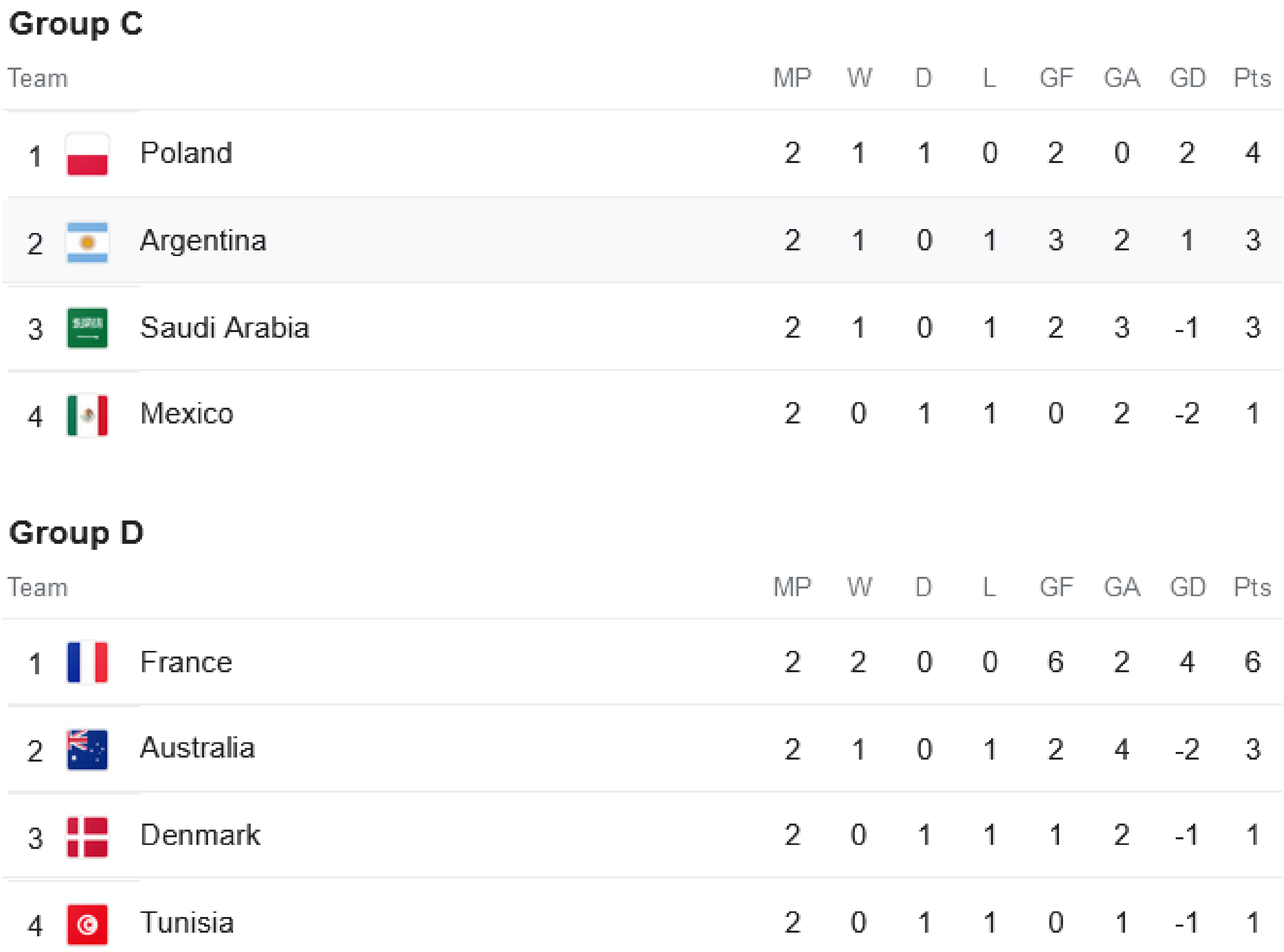
Team	MP	W	D	L	GF	GA	GD	Pts
1 Netherlands	3	2	1	0	5	1	4	7
2 Senegal	3	2	0	1	5	4	1	6
3 Ecuador	3	1	1	1	4	3	1	4
4 Qatar	3	0	0	3	1	7	-6	0

Group B

Team	MP	W	D	L	GF	GA	GD	Pts
1 England	3	2	1	0	9	2	7	7
2 USA	3	1	2	0	2	1	1	5
3 Iran	3	1	0	2	4	7	-3	3
4 Wales	3	0	1	2	1	6	-5	1

MP:	Match Played
W:	Won
D:	Draw
L:	Lose
GF:	Goal For
GA:	Goal Against
GD:	Goad Difference
Pts:	Points

Points Table



Group C

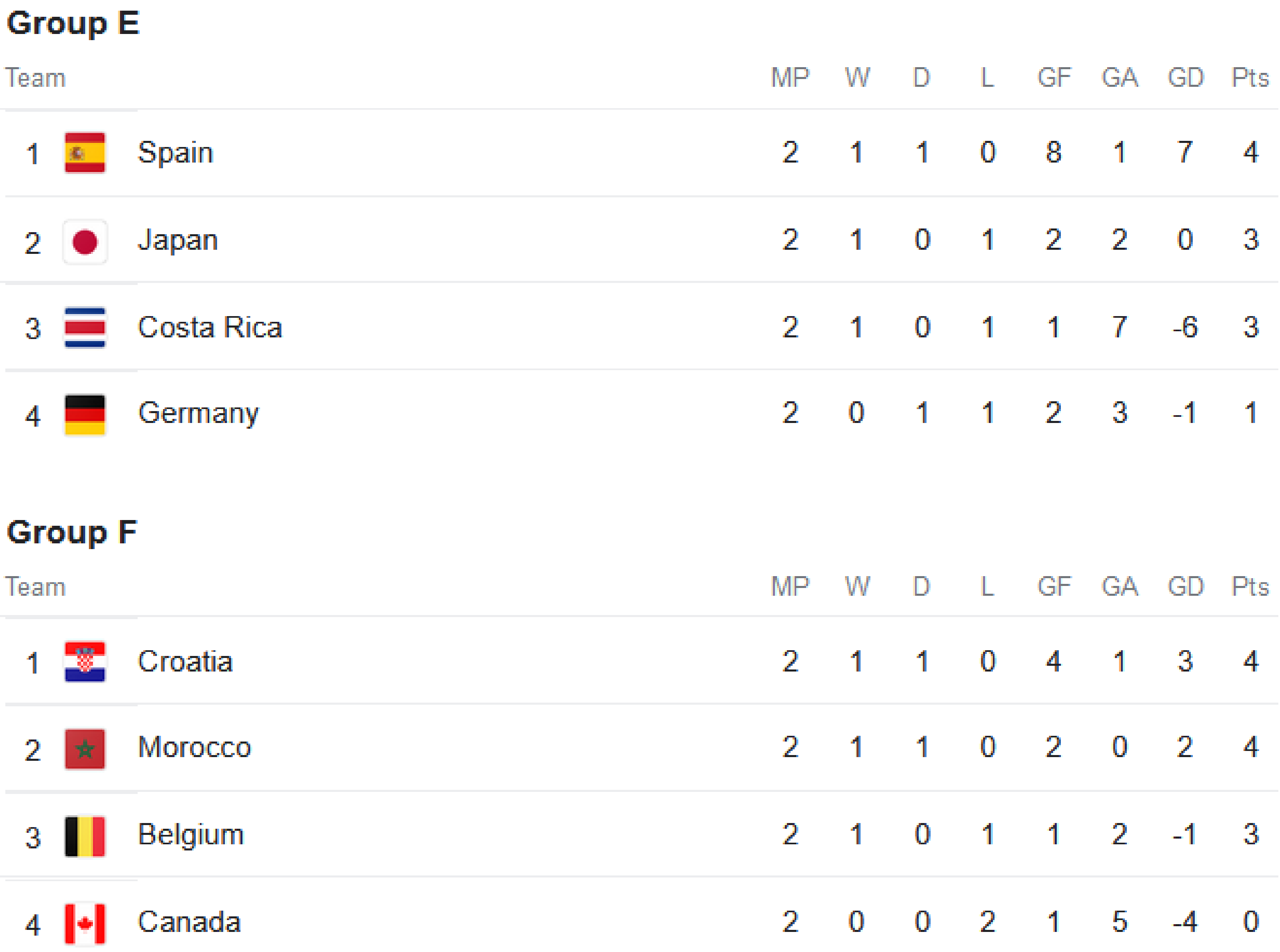
Team		MP	W	D	L	GF	GA	GD	Pts
1	Poland	2	1	1	0	2	0	2	4
2	Argentina	2	1	0	1	3	2	1	3
3	Saudi Arabia	2	1	0	1	2	3	-1	3
4	Mexico	2	0	1	1	0	2	-2	1

Group D

Team		MP	W	D	L	GF	GA	GD	Pts
1	France	2	2	0	0	6	2	4	6
2	Australia	2	1	0	1	2	4	-2	3
3	Denmark	2	0	1	1	1	2	-1	1
4	Tunisia	2	0	1	1	0	1	-1	1

MP:	Match Played
W:	Won
D:	Draw
L:	Lose
GF:	Goal For
GA:	Goal Against
GD:	Goad Difference
Pts:	Points

Points Table



Group E

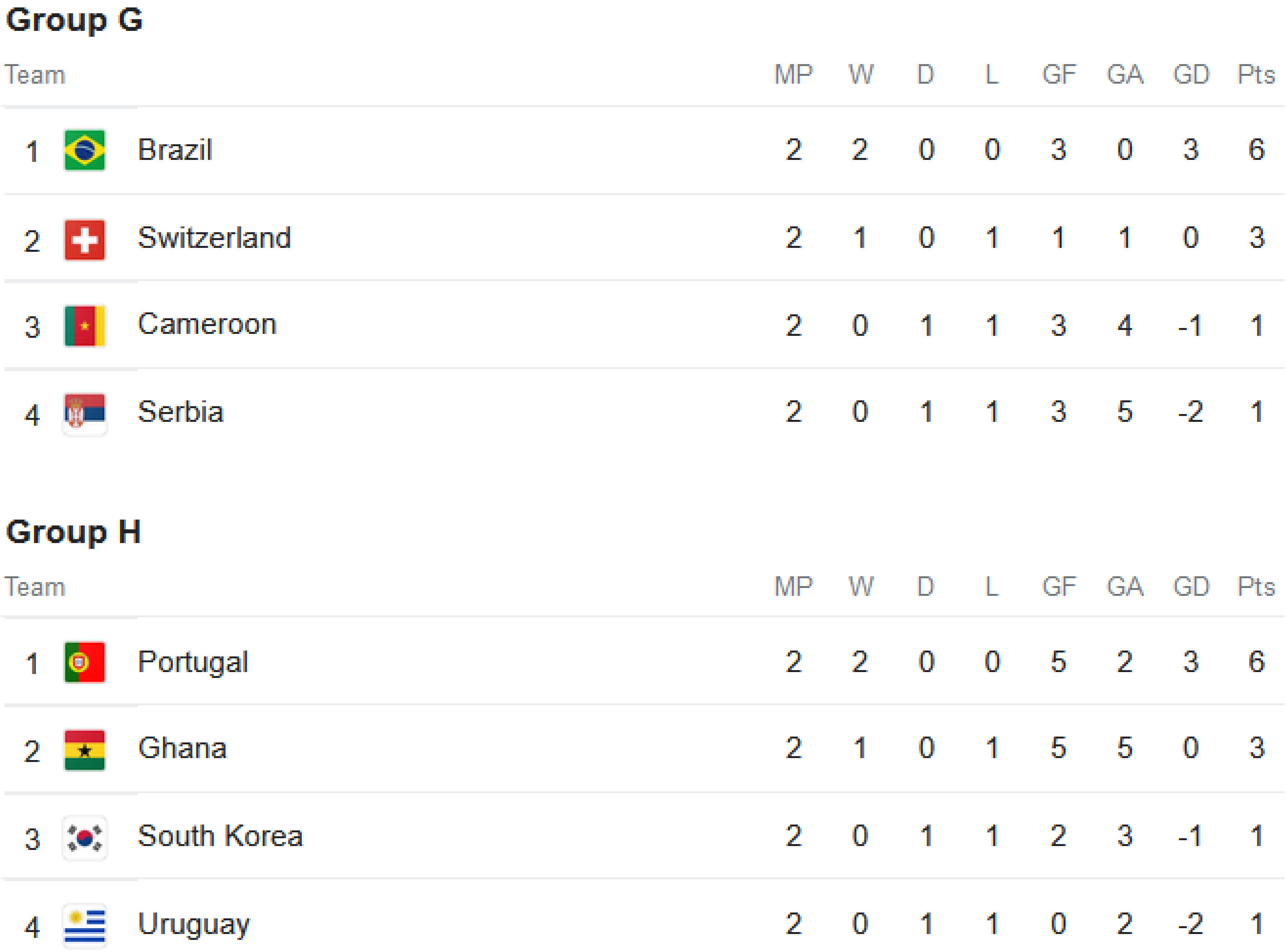
	Team	MP	W	D	L	GF	GA	GD	Pts
1	Spain	2	1	1	0	8	1	7	4
2	Japan	2	1	0	1	2	2	0	3
3	Costa Rica	2	1	0	1	1	7	-6	3
4	Germany	2	0	1	1	2	3	-1	1

Group F

	Team	MP	W	D	L	GF	GA	GD	Pts
1	Croatia	2	1	1	0	4	1	3	4
2	Morocco	2	1	1	0	2	0	2	4
3	Belgium	2	1	0	1	1	2	-1	3
4	Canada	2	0	0	2	1	5	-4	0

MP:	Match Played
W:	Won
D:	Draw
L:	Lose
GF:	Goal For
GA:	Goal Against
GD:	Goad Difference
Pts:	Points

Points Table



Group G

	Team	MP	W	D	L	GF	GA	GD	Pts
1	Brazil	2	2	0	0	3	0	3	6
2	Switzerland	2	1	0	1	1	1	0	3
3	Cameroon	2	0	1	1	3	4	-1	1
4	Serbia	2	0	1	1	3	5	-2	1

Group H

	Team	MP	W	D	L	GF	GA	GD	Pts
1	Portugal	2	2	0	0	5	2	3	6
2	Ghana	2	1	0	1	5	5	0	3
3	South Korea	2	0	1	1	2	3	-1	1
4	Uruguay	2	0	1	1	0	2	-2	1

MP: Match Played

W: Won

D: Draw

L: Lose

GF: Goal For

GA: Goal Against

GD: Goad Difference

Pts: Points

FIFA.CSV

```
Group,Team,Played,Won,Draw,Lost,GF,GA,GD,Points
A,Netherlands,3,2,1,0,5,1,4,7
A,Senegal,3,2,0,1,5,4,1,6
A,Ecuador,3,1,1,1,4,3,1,4
A,Qatar,3,0,0,3,1,7,-6,0
B,England,3,2,1,0,9,2,7,7
B,USA,3,1,2,0,2,1,1,5
B,Iran,3,1,0,2,4,7,-3,3
B,Wales,3,0,1,2,1,6,-5,1
C,Poland,2,1,1,0,2,0,2,4
```

CSV: Comma Separated Values. This file format is widely used format independent of programming languages and tools. Attributes are separated with comma, wok as long as comma is not part of any value

Example-1

```
file = open('fifa.csv', 'r')
heading = file.readline()
for i in range(32):
    string = file.readline().strip()
    print(string)
```

```
A,Netherlands,3,2,1,0,5,1,4,7
A,Senegal,3,2,0,1,5,4,1,6
A,Ecuador,3,1,1,1,4,3,1,4
A,Qatar,3,0,0,3,1,7,-6,0
B,England,3,2,1,0,9,2,7,7
B,USA,3,1,2,0,2,1,1,5
```

Example-2-I

```
class Team:
    def __init__(self, string):
        self.info = string.split(',')
        self.group = self.info[0]
        self.team = self.info[1]
        self.matches = int(self.info[2])
        self.won = int(self.info[3])
        ...
        self.goals_difference = int(self.info[9])

    def __str__(self):
        string = ''
        for s in self.info:
            string += s + '\t'
        return string
```

Example-2-II

```
file = open('fifa.csv', 'r')
heading = file.readline()
teams = []
for i in range(32):
    string = file.readline().strip()
    teams.append(Team(string))
for team in teams:
    print(team)
```

```
A Netherlands 3 2 1 0 5 1 4 7
A Senegal 3 2 0 1 5 4 1 6
A Ecuador 3 1 1 1 4 3 1 4
A Qatar 3 0 0 3 1 7 -6 0
B England 3 2 1 0 9 2 7 7
```

Example-3

```
file = open('fifa.csv', 'r')
heading = file.readline()
teams = []
for i in range(32):
    string = file.readline().strip()
    teams.append(Team(string))
most_difference = -99999999
most_difference_team_name = ''
for team in teams:
    if team.goals_difference > most_difference:
        most_difference_team_name = team.team_name
        most_difference = team.goals_difference
print(f'{most_difference_team_name} has most goal difference, which is: {most_difference}')
```

```
England has most goal difference,
which is: 7
```

Object Serialization

In serialization, an object is transformed into a format that can be stored, so as to be able to deserialize it later and recreate the original object from the serialized format.

Serialized data can be delivered to another data store (primary memory, secondary memory, some other application, or any other destination.

Pickle

Pickle is one of the simple option available in Python for serialization.

Pickling is the process whereby a Python object hierarchy is converted into a byte stream (usually not human readable) to be written to a file, this is also known as Serialization.

Unpickling is the reverse operation, whereby a byte stream is converted back into a working Python object hierarchy.

Pickle – Methods

The pickle interface provides four different methods:

dump() : The dump() method serializes to an open file (file-like object)
dumps(): Serializes to a string
load(): Deserializes from an open-like object
loads() : Deserializes from a string

Consider Point Class

```
class Point:
    def __init__(self, x, y):
        self.__x = x
        self.__y = y
    def __str__(self):
        return f'X: {self.__x}, Y: {self.__y}'

class Point3D(Point):
    def __init__(self, x, y, z):
        super().__init__(x, y)
        self.__z = z
    def __str__(self):
        return super().__str__() + f'\tZ: {self.__z}\n'
```

Pickle – Example 1-I

```
from pickle import *

p1 = Point3D(1, 2, 3)
p2 = Point3D(4, 5, 6)
file = open('point3d.bin', mode='wb')
dump(p1, file)
dump(p2, file)
file.close()
file = open('point3d.bin', mode='rb')
print(load(file))
print(load(file))
file.close()
```

```
X: 1, Y: 2 Z: 3

X: 4, Y: 5 Z: 6
```

Pickle – Example 1-II

The binary file created by pickle dump function.

```
€•L       Œ__main__”ŒPoint3D”“”)⍰}”(Œ _Point__x”KŒ
     _Point__y”KŒ
_Point3D__z”Kub.€•L       Œ__main__”ŒPoint3D”“”)⍰}”(Œ
     _Point__x”KŒ    _Point__y”KŒ
_Point3D__z”Kub.
```

Pickle – Example 2-I

```
p1 = Point2D(1, 2)
p2 = Point3D(4, 5, 6)
file = open('points.bin', mode='wb')
dump(p1, file)
dump(p2, file)
file.close()
print ('---------------')
file = open('points.bin', mode='rb')
print(load(file))
print(load(file))
```

```
X: 1, Y: 2
X: 4, Y: 5 Z: 6
```

Pickle – Example 2-II

The binary file created by pickle dump function.

```
€•@       Œ__main__”ŒPoint2D”“”)⍰}”(Œ
_Point2D__x”KŒ
_Point2D__y”Kub.€•P       Œ__main__”ŒPoint3D”“”)⍰}”(Œ
_Point2D__x”KŒ
_Point2D__y”KŒ
_Point3D__z”Kub.€•@       Œ__main__”ŒPoint2D”“”)⍰}”(Œ
_Point2D__x”KŒ
_Point2D__y”Kub.€•@       Œ__main__”ŒPoint2D”“”)⍰}”(Œ
_Point2D__x”KŒ
_Point2D__y”Kub.€•P       Œ__main__”ŒPoint3D”“”)⍰}”(Œ
_Point2D__x”KŒ
_Point2D__y”KŒ
_Point3D__z”Kub.
```

Pickle – Example 3

```
points.append(Point2D(1, 2))
points.append(Point3D(4, 5, 6))
points.append(Point2D(7, 8))
points.append(Point2D(6, 3))
points.append(Point3D(3, 5, 1))
file = open('points.bin', mode='wb')
for point in points:
    dump(point, file)
file.close()
print ('---------------')
file = open('points.bin', mode='rb')
try:
  while True:
     point = load(file)
     print(point)
except:
  pass
```

```
X: 1, Y: 2
X: 4, Y: 5 Z: 6
X: 7, Y: 8
X: 6, Y: 3
X: 3, Y: 5 Z: 1
```

Pickle – Example 3-II

The binary file created by pickle dump function.

```
€•@       Œ__main__”ŒPoint2D”“”)⍰}”(Œ
_Point2D__x”KŒ
_Point2D__y”Kub.€•P       Œ__main__”ŒPoint3D”“”)⍰}”(Œ
_Point2D__x”KŒ
_Point2D__y”KŒ
_Point3D__z”Kub.€•@       Œ__main__”ŒPoint2D”“”)⍰}”(Œ
_Point2D__x”KŒ
_Point2D__y”Kub.€•@       Œ__main__”ŒPoint2D”“”)⍰}”(Œ
_Point2D__x”KŒ
_Point2D__y”Kub.€•P       Œ__main__”ŒPoint3D”“”)⍰}”(Œ
_Point2D__x”KŒ
_Point2D__y”KŒ
_Point3D__z”Kub.
```

Pickle – Example 4-I

```
class Points: #aggregation
 def __init__(self, count):
  self.points = []
   for i in range(count):
    self.points.append(Point3D(randint(1,9), randint(1,9), randint(1,9)))
 def __str__(self):
 string = ''
        for point in self.points:
            string += str(point) + '\n'
        return string
```

Pickle – Example 4-II

```
p1 = Points(4)
file = open('points_points.bin', mode='wb')
dump(p1, file)
file.close()
print ('---------------')
file = open('points_points.bin', mode='rb')
p2 = load(file)
print(p2)
file.close()
```

```
X: 4, Y: 9 Z: 3
X: 3, Y: 1 Z: 5
X: 7, Y: 1 Z: 3
X: 6, Y: 6 Z: 2
```

Pickle – Example 3-II

The binary file created by pickle dump function.

```
€•³       Œ__main__”ŒPoints”“”)⍰}”Œpoints”]”(h ŒPoint3D”“”)⍰}”(Œ
_Point2D__x”KŒ
_Point2D__y”K   Œ
_Point3D__z”Kubh)⍰}”(h
KhKh
Kubh)⍰}”(h
KhKh
Kubh)⍰}”(h
KhKh
Kubesb.
```

Pickle – Example 5-I

```
class Points: #aggregation
 def __init__(self, count):
  self.points = []
   for i in range(count):
    self.points.append(Point3D(randint(1,9), randint(1,9), randint(1,9)))
 def __str__(self):
 string = ''
        for point in self.points:
            string += str(point) + '\n'
        return string
```

Pickle – Example 4-II

```
p1 = Points(4)
file = open('points_points.bin', mode='wb')
dump(p1, file)
file.close()
print ('---------------')
file = open('points_points.bin', mode='rb')
p2 = load(file)
print(p2)
file.close()
```

```
X: 4, Y: 9 Z: 3
X: 3, Y: 1 Z: 5
X: 7, Y: 1 Z: 3
X: 6, Y: 6 Z: 2
```

Pickle – String

We may dump our object into a string using:

dumps(): Serializes to a string

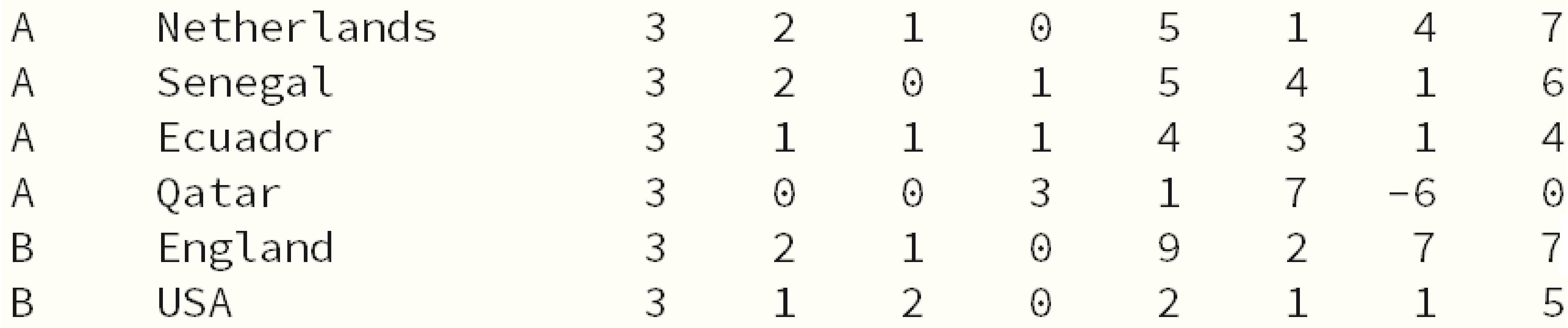
loads() : Deserializes from a string

Pickle – Example 4-I

```
from pickle import *
from tournament import *

file = open('fifa.csv', 'r')
tournament = Tournament(file)
file.close()
string = dumps(tournament) #transform object into string
print (string)
input('\nRead serialization string and press enter to see next
output')
print() #leave a blank line
tournament1 = loads(string) #transform string into object
print (tournament1)
```

Pickle – Example 4-II



```
A      Netherlands      3   2   1   0   5   1   4   7
A      Senegal          3   2   0   1   5   4   1   6
A      Ecuador          3   1   1   1   4   3   1   4
A      Qatar            3   0   0   3   1   7  -6   0
B      England          3   2   1   0   9   2   7   7
B      USA              3   1   2   0   2   1   1   5
```

Output

String

```
b'\x80\x04\x95\xa0\x0b\x00\x00\x00\x00\x00\x00\x8c\nto
urnament\x94\x8c\nTournament\x94\x93\x94)\x81\x94}\x94
\x8c\x05teams\x94]\x94(\x8c\x04team\x94\x8c\x04Team\x9
4\x93\x94)\x81\x94}\x94(\x8c\x04info\x94]\x94(\x8c\x01
A\x94\x8c\x0bNetherlands\x94\x8c\x013\x94\x8c\x012\x94
\x8c\x011\x94\x8c\x010\x94\x8c\x015\x94h\x12\x8c\x014\
x94\x8c\x017\x94e\x8c\x05group\x94h\x0e\x8c\tteam_name
\x94h\x0f\x8c\x07matches\x94K\x03\x8c\x03won\x94K\x02\
```