```
import json
import time
import msgpack
import statistics

class Owner(object):
    def __init__(self, id, name, birth, phone, address):
        self.id = id
        self.name = name
        self.birth = birth
        self.phone = phone
        self.address = address


class Pet(object):
    def __init__(self, id, name, species, gender, weight, birth, description, owner):
        self.id = id
        self.name = name
        self.species = species
        self.gender = gender
        self.weight = weight
        self.birth = birth
        self.description = description
        self.owner = owner


class Encoder(json.JSONEncoder):
    def default(self, o):
        return o.__dict__


def JSON(n, data):
    s = open("json_time_s.txt", "w")
    d = open("json_time_d.txt", "w")
    times_s = []
    times_d = []
    for i in range(n):
        with open('json_serialized.txt', 'w') as f:
            startS = time.perf_counter()
            json.dump(data, f, cls=Encoder)
            endS = time.perf_counter()


        with open('json_serialized.txt') as f:
            startD = time.perf_counter()
            json.load(f)
            endD = time.perf_counter()

        timeS = endS - startS
        timeD = endD - startD

        times_s.append(timeS)
        times_d.append(timeD)
```

```
54
55          s.write(str(timeS) + '\n')
56          d.write(str(timeD) + '\n')
57
58          #print(f"[JSON] - Serialization time: {timeS}")
59          #print(f"[JSON] - Deserialization time {timeD}")
60
61      s.write('\n' + str(statistics.mean(times_s)))
62      s.write('\n' + str(statistics.pstdev(times_s)))
63      d.write('\n' + str(statistics.mean(times_d)))
64      d.write('\n' + str(statistics.pstdev(times_d)))
65
66      s.close()
67      d.close()
68
69
70  def MSGPACK(n, data):
71      s = open("msgpack_time_s.txt", "w")
72      d = open("msgpack_time_d.txt", "w")
73      times_s = []
74      times_d = []
75      for i in range(n):
76          with open('msgpack_serialized.txt', 'wb') as f:
77              startS = time.perf_counter()
78              msgpack.pack(data, f, default=encoder_msgpack)
79              endS = time.perf_counter()
80
81          with open('msgpack_serialized.txt', 'rb') as f:
82              startD = time.perf_counter()
83              msgpack.unpack(f)
84              endD = time.perf_counter()
85
86          timeS = endS - startS
87          timeD = endD - startD
88
89          times_s.append(timeS)
90          times_d.append(timeD)
91
92          s.write(str(timeS) + '\n')
93          d.write(str(timeD) + '\n')
94
95          #print(f"[MSG PACK] - Serialization time: {timeS}")
96          #print(f"[MSG PACK] - Deserialization time: {timeD}")
97
98      s.write('\n' + str(statistics.mean(times_s)))
99      s.write('\n' + str(statistics.pstdev(times_s)))
100     d.write('\n' + str(statistics.mean(times_d)))
101     d.write('\n' + str(statistics.pstdev(times_d)))
102
103     s.close()
104     d.close()
105
106
107 def encoder_msgpack(o):
```

```
108     return o.__dict__
109 
110 
111 def gen_owners(n, owners):
112     for i in range(1, n + 1):
113         name = "Owner " + str(i)
114         owners.append(Owner(i, name, "19/10/2000", "912345678", "
    Coimbra"))
115 
116 
117 def gen_pets(n, pets):
118     for i in range(1, n + 1):
119         name = "Pet " + str(i)
120         species = "Species " + str(i)
121         description = name + " that belongs to " + species
122         if i % 2 == 0:
123             pets.append(Pet(i, name, species, "Male", 8, "10/04/2005"
    , description, i))
124         else:
125             pets.append(Pet(i, name, species, "Female", 4, "10/04/
    2005", description, i))
126 
127 
128 def gen_data(n):
129     data = []
130     gen_owners(n, data)
131     gen_pets(n, data)
132     return data
133 
134 
135 if __name__ == '__main__':
136     data = gen_data(100) #gerar dados
137     reps = 10
138     JSON(reps, data)
139     MSGPACK(reps, data)
```