| **Json** | **Pickle** |
| --- | --- |
| Json это текстовый формат обмена данными. Ключи в json могут быть только строками, заключенными в двойные кавычки | **Pickle** представляет бинарную сериализацию, то есть сериализацию в байты. |
| **Функция dumps()**  import json  data = {'name': 'Russia', 'phone\_code': 7, 'capital': 'Moscow', 'currency': 'RUB'}  json\_data = json.dumps(data) # сериализуем словарь data в json строку  print(type(json\_data)) print(json\_data)  <class 'str'>  {"name": "Russia", "phone\_code": 7, "capital": "Moscow", "currency": "RUB"} | **Функция dumps()**  **Функция dumps()** принимает сериализуемый Python объект, сериализует его в бинарный, Python-зависимый формат, используя протокол pickle и возвращает эти сериализованные данные.  import pickle  obj = {'Python': 1991, 'Java': 1995, 'C#': 2002}  binary\_obj = pickle.dumps(obj)  print(binary\_obj) print(type(binary\_obj))  b'\x80\x03}q\x00(X\x06\x00\x00\x00Pythonq\x01M\xc7\x07X\x04\x00\x00\x00Javaq\x02M\xcb\x07X\x02\x00\x00\x00C#q\x03M\xd2\x07u.'  <class 'bytes'>  **Тип данных** **bytes** — это неизменяемые последовательности отдельных байтов. Синтаксис для байтовых литералов в основном такой же, как и для строковых литералов, за исключением того, что добавляется префикс **b.** |
| **Функция dump()**  Функция dump() записывает переданный Python объект в файл.  Приведенный ниже код:  import json  data = {'name': 'Russia', 'phone\_code': 7, 'capital': 'Moscow', 'currency': 'RUB'}  with open('countries.json', 'w') as file:  json.dump(data, file)  **создает** файл **countries.json** и сохраняет в него информацию из словаря data в json формате. | **Функция dump()**  модуля pickle принимает сериализуемый Python объект, сериализует его в бинарный, Python-зависимый формат, используя протокол pickle, и  сохраняет его в открытый для записи бинарный файл.  import pickle  obj = {'Python': 1991, 'Java': 1995, 'C#': 2002}  with open('file.pkl', 'wb') as file:  pickle.dump(obj, file)  **создает** файл **file.pkl,** содержащий бинарное представление объекта obj на основе протокола pickle. |
| **Функция loads()** Для десериализации данных нужно использовать функцию **loads()**. Ее аргумент — это строка с данными в формате json.**Функция** **loads()** десериализует **json** строку в **словарь**.  import json  json\_data = '{"name": "Russia", "phone\_code": 7, "capital": "Moscow", "currency": "RUB"}'  data = json.loads(json\_data)  print(type(data))  print(data)  <class 'dict'>  {'name': 'Russia', 'phone\_code': 7, 'capital': 'Moscow', 'currency': 'RUB'}  В случае если строка для десериализации содержит данные с ошибкой, то модуль json не сможет правильно прочитать такую строку, и программа **завершится с ошибкой.** | **Функция loads()** Функция **loads()**выполняет десериализацию, она принимает объект типа **bytes**, содержащий сериализованные данные.  import pickle  obj = {'Python': 1991, 'Java': 1995, 'C#': 2002}  binary\_obj = pickle.dumps(obj)  new\_obj = pickle.loads(binary\_obj)  print(new\_obj)  {'Python': 1991, 'Java': 1995, 'C#': 2002}  Обратите внимание, что **объекты obj** и **new\_obj** равны, то есть имеют одинаковое содержимое, однако объекты не являются идентичными. Мы создали идеальную копию, но это всё же копия. |
| **Функция load()** Принимает файловый объект и возвращает его десериализованное содержимое.  Пусть файл data.json имеет следующее содержимое:  {  "name": "Russia",  "phone\_code": 7,  "capital": "Moscow",  "cities": ["Abakan", "Almetyevsk", "Anadyr", "Anapa", "Arkhangelsk", "Astrakhan"],  "currency": "RUB"  }  >>>>  import json  with open('data.json') as file:  data = json.load(file)  for key, value in data.items():  if type(value) == list:  print(f'{key}: {", ".join(value)}')  else:  print(f'{key}: {value}')  **Читает содержимое** data.json файла в словарь data и **выводит** его содержимое:  name: Russia  phone\_code: 7  capital: Moscow  cities: Abakan, Almetyevsk, Anadyr, Anapa, Arkhangelsk, Astrakhan  currency: RUB | **Функция load()** Принимает файловый объект, читает из него сериализованные данные, десериализует их в Python-объект и возвращает полученный Python-объект.  import pickle  with open('file.pkl', 'rb') as file:  obj = pickle.load(file)  print(obj)  print(type(obj))  {'Python': 1991, 'Java': 1995, 'C#': 2002}  <class 'dict'> |