1.What advantages do Excel spreadsheets have over CSV spreadsheets?

**Ans.** Excel spreadsheets have several advantages over CSV spreadsheets, including:

**Ability to handle complex data:** Excel spreadsheets can handle complex data more easily than CSV spreadsheets. For example, they can store data in multiple sheets within a single file, create charts and graphs, and use formulas and functions to analyze data.

**Formatting:** Excel allows you to format your data in a more visually appealing way than CSV. This includes setting font types and sizes, cell borders, shading, and color coding.

**Data validation:** Excel has built-in data validation features that allow you to set rules for data entry, such as specifying the range of acceptable values or requiring certain fields to be filled in.

**Security:** Excel allows you to password protect your spreadsheets and control who can view or edit them. CSV files, on the other hand, do not have any built-in security features.

**Macros:** Excel allows you to automate repetitive tasks with macros, which can save time and increase efficiency. CSV files do not have this capability.

2.What do you pass to csv.reader() and csv.writer() to create reader and writer objects?

**Ans.** To create a reader object using csv.reader(), you need to pass a readable file object, such as the result of opening a CSV file in read mode with open().

To create a writer object using csv.writer(), you need to pass a writable file object, such as the result of opening a CSV file in write mode with open().

3. What modes do File objects for reader and writer objects need to be opened in?

**Ans.** For a reader object, the File object needs to be opened in "read" mode, which is denoted by the letter **"r"**. This allows the reader object to read data from the file.

For a writer object, the File object needs to be opened in "write" mode, which is denoted by the letter **"w"**. This allows the writer object to write data to the file, overwriting any existing data.

4. What method takes a list argument and writes it to a CSV file?

**Ans.** The csv.writer() method is used to write a list argument to a CSV file in Python.

Here's an example code snippet that demonstrates this method:

**import csv**

**# create a list of data**

**data = [['Name', 'Age', 'Country'],**

**['John', '25', 'USA'],**

**['Sarah', '28', 'Canada'],**

**['David', '30', 'Australia']]**

**# open a CSV file in write mode**

**with open('output.csv', 'w', newline='') as file:**

**# create a CSV writer object**

**writer = csv.writer(file)**

**# write the data to the CSV file**

**writer.writerows(data)**

5. What do the keyword arguments delimiter and line terminator do?

**Ans.** The keyword argument delimiter is used to specify a character or string that separates values in a data file or a text string. For example, in a CSV (comma-separated values) file, the comma is the delimiter that separates each value in a row.

6. What function takes a string of JSON data and returns a Python data structure?

**Ans.** The function that takes a string of JSON data and returns a Python data structure is called "json.loads()". This function is provided by the "json" module in Python and is used to parse a JSON-formatted string and convert it into a Python object, such as a dictionary or a list. The "loads" in "json.loads()" stands for "load string", indicating that the function loads the string containing the JSON data and converts it into a Python object.

7. What function takes a Python data structure and returns a string of JSON data?

**Ans.** The function that takes a Python data structure and returns a string of JSON data is called "json.dumps()".

This function is part of Python's built-in "json" module, which provides functionality for encoding and decoding JSON (JavaScript Object Notation) data.

The "dumps" function takes a Python object as input and converts it to a JSON string representation, which can then be transmitted over the network or stored in a file.