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

ANS :- Excel spreadsheets have several advantages over CSV (Comma-Separated Values) spreadsheets:

1. Formatting: Excel supports rich formatting options, allowing you to customize the appearance of cells, apply styles, add colors, adjust column widths, and more. CSV files, on the other hand, contain only plain text data without any formatting.

2. Formulas and Functions: Excel provides a wide range of built-in formulas and functions that can be used to perform calculations, data analysis, and other operations on spreadsheet data. This makes it easier to perform complex calculations and automate tasks. CSV files do not support formulas or functions.

3. Multiple Sheets: Excel allows you to work with multiple sheets within a single workbook. This can be useful for organizing and managing data across different sheets or creating complex reports. CSV files represent a single sheet of data, and if you need multiple sheets, you would need separate CSV files.

4. Data Validation: Excel provides data validation features that allow you to define rules and restrictions on the input data. You can set data validation rules to ensure that only specific types of data or values within a certain range are entered. CSV files do not have built-in data validation capabilities.

5. Charts and Graphs: Excel supports creating various types of charts and graphs based on the data in the spreadsheet. This enables visual representation of data, making it easier to understand and analyze trends. CSV files do not support charting features.

6. Macros and Automation: Excel allows you to automate tasks and create custom macros using Visual Basic for Applications (VBA). This enables you to build sophisticated automated workflows and perform complex operations on the spreadsheet data. CSV files do not support macros or automation.

Overall, Excel provides a more comprehensive and feature-rich environment for working with spreadsheet data, offering advanced functionality, formatting options, data analysis capabilities, and customization possibilities that are not available in plain CSV files.

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

ANS :- To create reader and writer objects in the `csv` module, you pass a file object as an argument to the `csv.reader()` and `csv.writer()` functions. The file object represents the CSV file you want to read from or write to.

For example, to create a reader object for reading from a CSV file, you can pass an opened file object in read mode to `csv.reader()` like this:

```python

import csv

with open('data.csv', 'r') as file:

reader = csv.reader(file)

# Use the reader object to read data from the CSV file

# ...

```

Similarly, to create a writer object for writing to a CSV file, you can pass an opened file object in write mode to `csv.writer()` like this:

```python

import csv

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

writer = csv.writer(file)

# Use the writer object to write data to the CSV file

# ...

```

In both cases, the `open()` function is used to open the CSV file, and the resulting file object is passed to the respective `csv.reader()` or `csv.writer()` function to create the reader or writer object.

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

ANS :- For reader objects in the `csv` module, the File object needs to be opened in text mode with the `'r'` (read) mode. Therefore, you need to open the file with the `open()` function using the `'r'` mode before passing it to `csv.reader()`.

Example:

```python

import csv

with open('data.csv', 'r') as file:

reader = csv.reader(file)

# Use the reader object to read data from the CSV file

# ...

```

For writer objects in the `csv` module, the File object needs to be opened in text mode with the `'w'` (write) mode. Additionally, if you're writing to a CSV file on Windows, you should pass `newline=''` as well to ensure proper handling of line endings.

Example:

```python

import csv

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

writer = csv.writer(file)

# Use the writer object to write data to the CSV file

# ...

```

In both cases, the `open()` function is used with the appropriate mode to open the file, and the resulting File object is used in conjunction with `csv.reader()` or `csv.writer()` to perform the desired reading or writing operations on the CSV file.

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

ANS :- The `writerow()` method is used to write a list of values as a single row in a CSV file using the `csv.writer` object. This method takes a list as an argument, where each element of the list corresponds to a value in the row.

Here's an example that demonstrates how to use the `writerow()` method to write a list of values as a row in a CSV file:

```python

import csv

data = ['John', 'Doe', 'john.doe@example.com']

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

writer = csv.writer(file)

writer.writerow(data)

```

In this example, the `writerow()` method is called on the `writer` object, passing the `data` list as an argument. The values in the `data` list will be written as a single row in the CSV file.

Note that you can call `writerow()` multiple times to write multiple rows to the CSV file, with each call representing a new row of data.

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

ANS :- The `delimiter` keyword argument is used to specify the character or sequence of characters that separates fields (values) in a CSV file. By default, the delimiter is a comma (`,`), but you can customize it to use a different character or sequence.

The `line\_terminator` keyword argument is used to specify the character or sequence of characters that marks the end of a line in a CSV file. By default, the line terminator is the newline character (`\n`), but you can provide a different character or sequence if needed.

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

ANS :- The json.loads() function is used to take a string of JSON data and convert it into a Python data structure. The function name "loads" stands for "load from string." It parses the JSON-formatted string and constructs the corresponding Python objects, such as dictionaries, lists, strings, numbers, booleans, and None.

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

ANS :- The json.dumps() function is used to take a Python data structure and convert it into a string of JSON data. The function name "dumps" stands for "dump to string." It serializes the Python objects into a JSON-formatted string representation.