Q1. Describe the differences between text and binary files in a single paragraph.

Text files contain human-readable characters encoded using a specific character encoding, such as ASCII or Unicode. They typically store plain text, including alphabets, numbers, symbols, and control characters. Text files are structured with newline characters to represent line breaks and are commonly used for storing textual data that can be easily read and modified using text editors

Q2. What are some scenarios where using text files will be the better option? When would you like to use binary files instead of text files?

Using Text Files:

When storing or handling textual data, such as plain text documents, configuration files, or log files, text files are the better option. They preserve the human-readable format and can be easily opened and modified with text editors. Text files are also compatible with various programming languages, allowing easy parsing and manipulation of the text data.

Text files are suitable when you need interoperability between different systems and platforms. Since they use standardized character encodings (e.g., UTF-8), they can be reliably shared and processed across different operating systems and software

Q3. What are some of the issues with using binary operations to read and write a Python integer directly to disc?

Endianness

Data Type Size

Portability

Data Interpretation

Q4. Describe a benefit of using the with keyword instead of explicitly opening a file.

Using the with keyword in Python when working with files provides the benefit of automatic resource management, specifically for file handling. When you use the with statement, Python takes care of the necessary setup and teardown operations for the file, ensuring proper handling and cleanup, even in the event of exceptions or errors

Q5. Does Python have the trailing newline while reading a line of text? Does Python append a newline when you write a line of text?

Yes, Python does preserve the trailing newline when reading a line of text from a file. When you use the readline() method or iterate over a file object using a for loop, Python includes the newline character (\n) at the end of each line in the returned string

Q6. What file operations enable for random-access operation?

Random-access operations, which allow accessing data at any position within a file, can be achieved using the following file operations:

seek()

tell()

Q7. When do you think you'll use the struct package the most?

The struct package in Python is particularly useful when working with binary data or performing low-level data manipulations

Q8. When is pickling the best option?

Pickling in Python refers to the pro

zcess of serializing objects into a binary format for storage or transmission. Pickling offers several benefits and is a good option in the following scenarios:

Object Persistence

Data Caching

Interprocess Communication

Machine Learning Model Serialization

Q9. When will it be best to use the shelve package?

The shelve package in Python provides a simple and efficient way to store and retrieve Python objects as key-value pairs. It is built on top of the dbm module and offers a persistent dictionary-like interface.

Q10. What is a special restriction when using the shelve package, as opposed to using other data dictionaries?

When using the shelve package, there is a special restriction to be aware of compared to using other data dictionaries: the keys in a shelve shelf must be strings.

In shelve, the keys are internally stored as strings, and any attempt to use a key that is not a string will result in a TypeError. This restriction means that you cannot use keys of other data types, such as integers, floats, or tuples directly.