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

Ans: Binary file contains the data in the form of 0 and 1(series of binary values) and text files contains the data in the form of stream of characters.

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?

Ans: When file does need to be read by people or need to be ported to a different type of system text files should be preferred over binary files.

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

Ans: Python has tools for working with binary files. Binary files use strings of type bytes. This means when reading binary data from a file, an object of type bytes is returned. The binary file is opened using the open() function, whose mode parameter contains the character 'b'.

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

Ans: with statement in Python is used in exception handling to make the code cleaner and much more readable. It simplifies the management of common resources like file streams.

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?

Ans: Python provides three methods to read data from the input file. The readline method reads one line from the file and returns it as a string. The string returned by readline will contain the newline character at the end.

Yes python can append newline when we write a line of text.

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

Ans: This class is used for reading and writing to random access file. A random access file behaves like a large array of bytes. There is a cursor implied to the array called file pointer, by moving the cursor we do the read write operations

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

Q8. When is pickling the best option?

Ans: Pickle in Python is primarily used in serializing and deserializing a Python object structure. In other words, it's the process of converting a Python object into a byte stream to store it in a file/database, maintain program state across sessions, or transport data over the network.

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

Ans: The shelve module implements persistent storage for arbitrary Python objects which can be pickled, using a dictionary-like API.

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

Ans: The shelf dictionary has certain restrictions. Only string data type can be used as key in this special dictionary object, whereas any picklable Python object can be used as value.