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

|  |  |
| --- | --- |
| **Text File** | **Binary File** |
| Bits represent character. | Bits represent a custom data. |
| Less prone to get corrupt as changes reflect as soon as the file is opened and can easily be undone. | Can easily get corrupted, even a single bit change may corrupt the file. |
| Can store only plain text in a file. | Can store different types of data (image, audio, text) in a single file. |
| Widely used file format and can be opened using any simple text editor. | Developed especially for an application and may not be understood by other applications. |
| Mostly .txt and .rtf are used as extensions to text files. | Can have any application defined extension. |

**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?**

text files are less prone to get corrupted as any undesired change may just show up once the file is opened and then can easily be removed. Whereas Use binary files instead of text files for image data.

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

When we read or write a python integer using binary operations

1. Binary operations deal with raw data
2. One needs to identify how many bytes one would read or write.

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

When a file is opened using the **with** keyword, if some exceptions occur after opening a file, or at the end of the file it automatically does the closing of the file. There by not leaving an file in open mode and there would no need to explicitly close a file.

**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 have the trailing newline while reading a line of text. When we write a newline has to be provided in python excpicitly.

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

The file operations enable for random-access operation are **seek()** and **tell()**

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

he **struct** package is mostly used while converting a common python data types into **C** language types and vice versa

**Q8. When is pickling the best option?**

 Pickling is best option for creating a new binary file using python.

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

**Shelve** package is used to pickle data but treats the entire file as dictionary. pickle is for serializing some object (or objects) as a single bytestream in a file. shelve builds on top of pickle and implements a serialization dictionary where objects are pickled, but associated with a key (some string), so you can load your shelved data file and access your pickled objects via keys.

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

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.