1. Binary files typically contain a sequence of bytes, or ordered groupings of eight bits.When creating a custom

# file format for a program, these bytes are arranged into a format that stores the necessary information for

# the application. Binary file formats may include multiple types of data in the same file, such as image, video,

# and audio data. This data can be interpreted by supporting programs, but will show up as garbled text in a

# text editor.

# Text files are more restrictive than binary files since they can only contain textual data.However, unlike

# binary files, they are less likely to become corrupted. While a small error in a binary file may

# make it unreadable, a small error in a text file may simply show up once the file has been opened. Text files

# may be saved in either a plain text (.TXT) format and rich text (.RTF) format. A typical plain text file

# contains several lines of text that are each followed by an End-of-Line (EOL) character. An End-of-File (EOF)

# marker is placed after the final character, which signals the end of the file. Rich text files use a similar

# file structure, but may also include text styles, such as bold and italics, as well as page formatting

# information. Both plain text and rich text files include a (character encoding) scheme that determines how the

# characters are interpreted and what characters can be displayed. Since text files use a simple, standard

# format, many programs are capable of reading and editing text files.

2. # Ans : Text files include small size and versatility. Kilobytes or megabytes smaller than the same data stored in

# other formats, they can be rapidly and massively exchanged via email or disk. Most can be opened on computers

# running diverse operating systems, using very basic software. Binary files is that they are more efficient.

# In terms of memory, storing values using numeric formats, rather than as text characters, tends to use less

# memory. In addition, binary formats also offer advantages in terms of speed of access.

3. # Ans : When we read or write a python integer using binary operations

# a)Binary operations deal with raw data

# b)one needs to identify how many bytes one would read or write.

4. # Ans : 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.

5. # Ans : While reading a newline of text from atext file, python reads the newline also.

# While writing the python doesnt append a new line at end of line. It has to be handled explicitly.

6. # Ans : While reading a newline of text from atext file, python reads the newline also.

# While writing the python doesnt append a new line at end of line. It has to be handled explicitly.

7. # Ans : The struct package is mostly used while converting a common python types into 'C' language types.This is done

# by packing python variables into data fields of specific sizes.So, when we try read , right number of bytes are

# read. This useful when interacting with existing binary files.

# Reading and writing a single integer using struct package

from struct import pack, unpack, calcsize

def write\_file(fname,int\_n):

with open(fname,'wb') as f:

bss=pack('h',int\_n)

f.write(bss)

def read\_file(fname):

with open(fname,'rb') as rf:

bss=rf.read(calcsize('h'))

return unpack('h',bss)

write\_file('struct\_file1.dat',155)

read\_file('struct\_file1.dat')

(155,)

8. # Ans : Once a file is created it can be read by other python programs. The functions available in the pickle package

# take care how to represent the data is written in th file.

9. # Ans : Shelve package is used to pickle data but treats the treats the entire file as dictionary. The location of any

# object is looked up according to its key and is returned easily.

10. # Ans : A “shelf” is a persistent, dictionary-like object. The difference with “dbm” databases is that the values

# (not the keys!) in a shelf can be essentially arbitrary Python objects — anything that the pickle module can

# handle. This includes most class instances, recursive data types, and objects containing lots of shared

# sub-objects. The keys are ordinary strings.