**1. To what does a relative path refer?**

A relative path refers to the path of a file or directory relative to the current working directory. It does not start from the root directory but instead assumes the current location as the starting point.

**2. What does an absolute path start with your operating system?**

The starting point for an absolute path in most operating systems, including Windows, macOS, and Linux, is the root directory.

On Windows systems, an absolute path typically starts with the drive letter followed by a colon (C:, D:, etc.) and then continues with the directory hierarchy. For example, C:\Users\Username\Documents\file.txt represents an absolute path on Windows.

On macOS and Linux systems, an absolute path starts with a forward slash (/) and then continues with the directory hierarchy. For example, /Users/Username/Documents/file.txt represents an absolute path on macOS or Linux.

**3. What do the functions os.getcwd() and os.chdir() do?**

os.getcwd(): This function returns a string representing the current working directory. The current working directory is the directory from which the Python script is being executed.

os.chdir(path): This function changes the current working directory to the specified path. The path can be either an absolute path or a relative path.

**4. What are the . and .. folders?**

. (dot): The . (dot) directory represents the current directory. It serves as a reference to the directory where the current file or script is located. It is often used to specify the current directory in file paths or to reference files within the same directory.

.. (dot-dot): The .. (dot-dot) directory represents the parent directory. It serves as a reference to the directory immediately above the current directory in the file system hierarchy. It is commonly used to navigate up one level in the directory structure.

**5. In C:\bacon\eggs\spam.txt, which part is the dir name, and which part is the base name?**

Directory Name: C:\bacon\eggs

Base Name: spam.txt

**6. What are the three “mode” arguments that can be passed to the open() function?**

Read Mode ('r'): This is the default mode if no mode argument is specified. It opens the file for reading. If the file does not exist or cannot be opened, it raises a FileNotFoundError exception.

Write Mode ('w'): This mode opens the file for writing. If the file already exists, it truncates (empties) the file content. If the file does not exist, it creates a new file. If the operation is successful, the file pointer is positioned at the beginning of the file.

Append Mode ('a'): This mode opens the file for appending data. If the file exists, the new data is written at the end of the file. If the file does not exist, it creates a new file. If the operation is successful, the file pointer is positioned at the end of the file.

**7. What happens if an existing file is opened in write mode?**

Opening the existing file in write mode will truncate or clear the contents of the file, effectively erasing all existing data. The file will be ready to accept new data starting from the beginning of the file.

**8. How do you tell the difference between read() and readlines()?**

read(): The read() method reads the entire contents of a file as a single string.

readlines(): The readlines() method reads the contents of a file line by line and returns a list of strings.

**9. What data structure does a shelf value resemble?**

In Python, a shelf value, as obtained from the shelve module, resembles a dictionary-like data structure. The shelf behaves like a dictionary, allowing us to store, retrieve, and update data using keys.