File Packer Unpacker - Interview Questions and Answers

1. Explain the flow and working of your project.
The project is a file packer and unpacker built using Java and Swing for the graphical user interface (GUI). The process begins with a login screen where the user must enter the credentials "Marvellous" and "Marvellous". Upon successful login, the main menu offers two options: Pack and Unpack.

• Packing Process: The user specifies a directory name and a destination packed file name. The program then iterates through all files in the given directory. For each file, it creates a 100-byte header containing the file's name and size. This header is written to the packed file, followed by the content of the original file.
• Unpacking Process: The user provides the name of the packed file. The program reads the packed file in 100-byte chunks. Each chunk is a header, from which the program extracts the original file name and size. A new file with the original name is created, and the data (the exact number of bytes specified in the header) is read from the packed file and written to the new file.

2. What is mean by Packing And Unpacking?
Packing is the process of consolidating multiple files from a directory into a single, combined file. This single file contains both the content of the original files and metadata, like file names and sizes, in a 100-byte header. Unpacking is the reverse operation; it takes the packed file and recreates all the original files using the information stored in the headers.

3. What is the use of Encryption?
Encryption is a security measure that converts data into a coded format to prevent unauthorized access. In a file packing utility, encryption would add a layer of protection. The provided project, however, does not include any encryption.

4. How you can save memory by using packing?
The current project does not save memory. It combines files without using compression. The total size of the packed file is the sum of the sizes of all the original files, plus the 100-byte header for each file.

5. Why you select Java as a developmental language for this project?
Java was chosen because it is a platform-independent language, allowing the application to run on various operating systems with a Java Virtual Machine (JVM). Its libraries, like java.io for file handling and javax.swing for the GUI, are well-suited for this type of project.

6. Explain the concept of MD5 Checksum?
An MD5 checksum is a unique 128-bit digital signature for a file, used to verify data integrity. A slight change in the file content results in a completely different checksum. This project does not use MD5 checksums.

7. How you recreate all the files again in case of unpacking?
The MarvellousUnpacker class handles the recreation of files by reading the packed file sequentially. It reads a 100-byte header to get the original file's name and size. It then creates a new file with that name and writes the exact number of bytes (the size) from the packed file into the newly created file. This process is repeated for each file in the packed archive.

8. How you maintain a record of each file in case of packing?
The MarvellousPacker class maintains a record by creating a fixed-size, 100-byte header for each file. This header contains the file's name and its size, padded with spaces. The header is written to the packed file just before the content of the corresponding file.

9. Which User interface you provide for this project?
The project provides a graphical user interface (GUI) built using the Java Swing library. This GUI includes a login frame, a main menu, a packing frame, and an unpacking frame.

10. Which is the targeted audience for this project?
The targeted audience would be users who need a basic utility for combining and extracting files, such as students, developers, or individuals who require a simple file grouping solution without advanced features.

11. Are you generating any log of Packing and unpacking activity?
Yes, the project generates a log of its activities by printing messages to the console using System.out.println. It logs events like directory access, file creation, and a statistical report showing the number of files processed.

12. Which types of File Manipulations are used in this project?
The project uses core file I/O operations from the java.io package.
• File Reading: It uses FileInputStream to read data from source files and the packed file.
• File Writing: It uses FileOutputStream to write data to the packed file and to the newly created files during unpacking.

13. What happens if our folder contains duplicate files in case of packing?
If a folder contains duplicate file names, the packing process will include both files. However, during unpacking, the second file extracted with the same name will overwrite the first one, leaving only the last one with that name.

14. Which types of files get packed and unpacked using your project?
The project can pack and unpack any type of file, as the process operates at the byte level. It does not depend on file extensions or content and can handle text, binary, images, and other file types.

15. What are the resources that you refer during development of this project?
The primary resources for this project are the official Java documentation (Java API) for the java.io and javax.swing packages, along with online tutorials and forums.

16. Which difficulty you faced in this project?
A key challenge is ensuring the fixed-size 100-byte header is correctly handled. Issues with padding or parsing the header can lead to corrupted files during the unpacking process.

17. Is there any chance of improvement in your project?
Yes, there is significant potential for improvement. Possible enhancements include adding compression, encryption, a progress bar, a file browser, and more robust error handling.

18. Can we use this project on different platforms?
Yes, the project is platform-independent because it is written in Java. It can run on any operating system that has a Java Runtime Environment (JRE) installed.

19. Explain File header in case of packed file?
A file header is a 100-byte block of metadata written at the beginning of each file's data in the packed file. It stores the original file name and its size, which is critical for the unpacker to correctly recreate the file.

20. Which type of Encryption technique is used for this project?
This project does not use any encryption technique. The packed file content is not protected and can be viewed using a simple editor.