Name – Shivranjan Pathak

College-MITWPU

File Packer Unpacker

This project is used to perform process monitoring activity. - By using this project we can fetch data from all files and merge it into one file. As well as we can also extract all packed file whenever required.

Platform required-

Windows NT platform OR Linux

Architectural requirement-

Intel 32 bit processor

User Interface-
Command- Line Interface
or
Graphical User Interface

Technology used-

Java Programming

Features provided by File Packer-Unpacker-

This project is divided into two parts as Packing and Unpacking.

Packing Activity :

• In case of Packing activity we accept directory name and file name from user.

• We have to create new regular file as the name specified by the user.

• Now open the directory and traverse each file from that directory. In newly created file write Metadata as header and actual file data in sequence.

• While writing data perform encryption.

• Each name of file, its size and checksum should be written in log file which gets created in system directory.

• After packing display packing report.

UnPacking Activity :

• In case of UnPacking activity we accept packed file name from user.

• for authentication of packed file use any logic like Magic Number.

• Open the packed file in read mode and perform below activity as

• Read header

• From the name specified in header create new file.

• Write data into newly created file from packed file.

• Repeat all above steps till we reached at end of the file unpacked file.

• After unpacking display unpacking report.

Main purpose of this project is to merge large amount of files into one file by avoiding memory wastage.

We also provide data security by using the concept of Encryption. For next level data security we add MD5 Checksum check and Primary header check.

Expected interview Questions on File Packer Unpacker

Explain flow and Working of your project?

- The project allows users to either pack (combine) text files from a specified directory into a single packed file or unpack (extract) files from a packed file back into individual files. It operates through a command-line interface (CLI) or graphical user interface (GUI), where users input directory paths, file names, and choose actions like packing, unpacking, or displaying help.

What is meant by Packing and Unpacking?

- **Packing:** Combining multiple files into a single file. In this context, text files from a directory are packed into a single packed file.
- **Unpacking:** Extracting files from a packed file back into their original individual files.

What is the use of Encryption?

- Encryption is used to secure the contents of files during packing, making them unreadable without the proper decryption key. This ensures confidentiality and prevents unauthorized access to the data.

How can you save memory by using packing?

- Packing reduces memory usage by combining multiple files into one. Instead of keeping track of and accessing many individual files, only one file needs to be managed and processed.

Why did you select Java as a developmental language for this project?

- Java was chosen for its platform independence, robustness, extensive libraries (like Swing for GUI), and strong support for file handling and networking. It allows the project to run on any platform with a Java Virtual Machine (JVM).

Explain the concept of MD5 Checksum?

- MD5 (Message Digest Algorithm 5) Checksum is a hash function that produces a fixed-size (128-bit) hash value from input data. It is used to verify data integrity by comparing checksums of the original and received data. If the checksums match, the data is likely to be intact.

How do you recreate all the files again in case of unpacking?

- During unpacking, each file's metadata (like filename and size) is stored in the packed file's header. This metadata is used to recreate each file by reading the corresponding data blocks from the packed file and writing them into new individual files.

How do you maintain a record of each file in case of packing?

- Each file's metadata (filename and size) is appended to the header of the packed file. This allows the unpacking process to know how many files are in the packed file and how to reconstruct each file.

Which User interface do you provide for this project?

- Initially a command-line interface (CLI) was provided. Later, a graphical user interface (GUI) using Java Swing was implemented in the GUI version. Users interact with buttons and input fields to perform actions like packing, unpacking, displaying help, and exiting.

Which is the targeted audience for this project?

- The project targets users who need to efficiently archive and extract text files, especially in scenarios where bundling files for distribution or storage is useful.

Are you generating any log of Packing and unpacking activity?

- The current implementation does not explicitly generate logs. However, logging mechanisms could be added to record activities such as file operations, errors, and other relevant events for troubleshooting and auditing purposes.

Which types of File Manipulations are used in this project?

- File manipulations include reading and writing files using FileInputStream, FileOutputStream, and other file-related classes in Java. These are used for reading files to pack and writing files during unpacking.

What happens if our folder contains duplicate files in case of packing?

- The current implementation of the project does not handle duplicate files explicitly. It would typically overwrite existing files if files with the same name are packed into the same packed file.

Which types of files get packed and unpacked using your project?

- The project is designed to pack and unpack text files (files ending with .txt). Other file types are not handled by the current implementation.

What are the resources that you refer to during development of this project?

- Development resources include Java documentation, tutorials on file handling and Swing GUI, and possibly resources related to cryptographic techniques and algorithms for encryption.

What difficulties did you face in this project?

- Common challenges might include handling file I/O operations safely, ensuring data integrity during packing/unpacking, designing a user-friendly GUI, and implementing encryption securely.

Is there any chance of improvement in your project?

- Yes, potential improvements could include handling different file types, adding support for directories (recursively packing/unpacking), implementing more robust error handling and logging, and enhancing security features like using stronger encryption algorithms.

Can we use this project on different platforms?

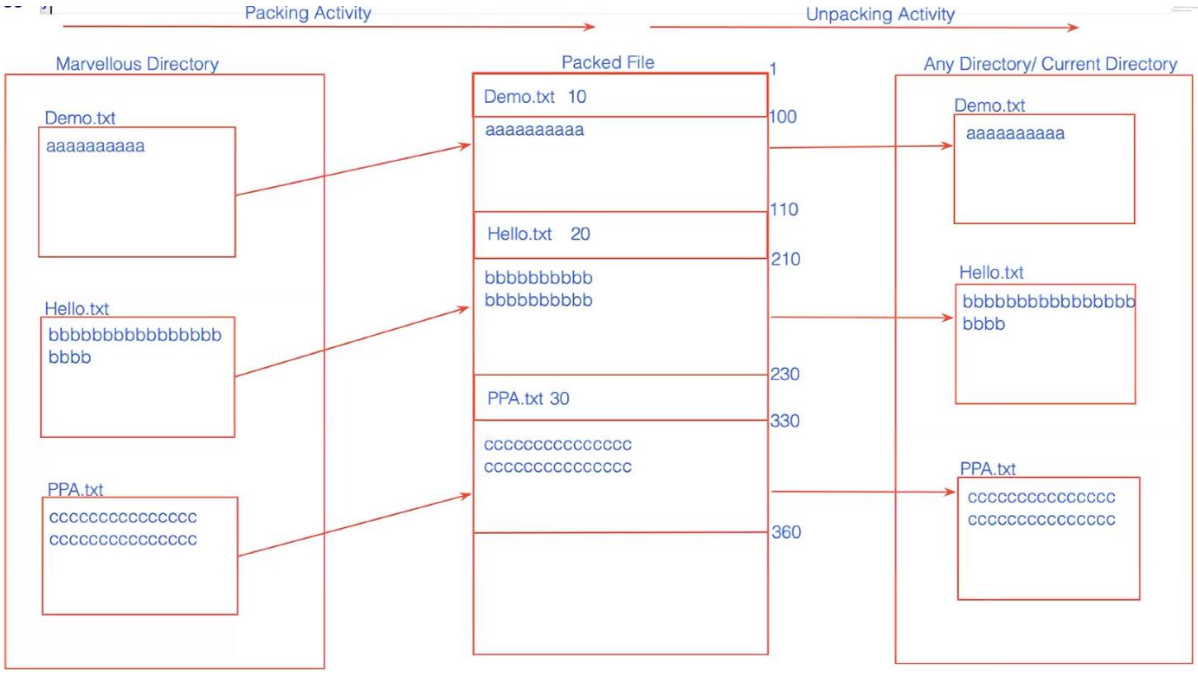
- Yes, Java's platform independence allows the project to run on different platforms (Windows, macOS, Linux) as long as a Java Runtime Environment (JRE) or Java Development Kit (JDK) is installed.

Explain File header in case of a packed file?

- The file header in the packed file contains metadata for each packed file, typically including the filename and size. This metadata is used during unpacking to recreate the original files.

Which type of Encryption technique is used for this project?

- The provided code does not implement encryption. However, common encryption techniques like AES (Advanced Encryption Standard) could be integrated for securing file contents during packing.


Packing Activity
Unpacking Activity
Marvellous Directory
Demo.txt
aaaaaaaaaa
Hello.txt
bbbbbbbbbbbbbbb
bbbb
PPA.txt
ccccccccccccc
ccccccccccccc
Packed File
1
Demo.txt 10
aaaaaaaaaa
100
110
Hello.txt 20
210
bbbbbbbbbb
bbbbbbbbbb
230
PPA.txt 30
330
ccccccccccccc
ccccccccccccc
360
Any Directory/ Current Directory
Demo.txt
aaaaaaaaaa
Hello.txt
bbbbbbbbbbbbbbb
bbbb
PPA.txt
ccccccccccccc
ccccccccccccc