***Directory structure maintenance using shell script in the operating system***

**GROUP MEMBER:**

AYUSHI KOUL (2021A1R009)

ANJANA SHARMA (2021A1R011)

TANISHA SOMA (2021A1R029)

**SUBMITTED TO:** MRS. VEENA TRIPATHI MAM

**ACKNOWLEDGEMENT**

We would like to express our special thanks of gratitude to our operating system lab teacher “Mrs. Veena Tripathi” for their able guidance and support in completing our project.

We came to know about so many new things that we were unaware of.

**“Directory structure maintenance using shell script in the operating system”**

➢ **ABSTRACT:**

In this report we are going to learn about The selection of directory-allocation and directory-management algorithms significantly affects the efficiency, performance, and reliability of the file system. In this section, we discuss the trade-offs involved in choosing one of these algorithms.

➢ **INTRODUCTION**

Directory implementation in the operating system can be done using Singly Linked List and Hash table. The efficiency, reliability, and performance of a file system are greatly affected by the selection of directory-allocation and directory-management algorithms. There are numerous ways in which the directories can be implemented. But we need to choose an appropriate directory implementation algorithm that enhances the performance of the system.

➢ **Types of implementation of directory**

Following are the two main types of **implementation of directory** :

1. Directory Implementation Using Singly Linked List: To create a new file the entire list has to be checked such that the new directory does not exist previously.

The new directory then can be added to the end of the list or at the beginning of the list.

In order to delete a file, we first search the directory with the name of the file to be deleted. After searching we can delete that file by releasing the space allocated to it.

To reuse the directory entry we can mark that entry as unused or we can append it to the list of free directories.

To delete a file linked list is the best choice as it takes less time.

2. Directory Implementation using Hash Table: An alternative data structure that can be used for directory implementation is a hash table. It overcomes the major drawbacks of directory implementation using a linked list. In this method, we use a hash table along with the linked list. Here the linked list stores the directory entries, but a hash data structure is used in combination with the linked list.

In the hash table for each pair in the directory key-value pair is generated. The hash function on the file name determines the key and this key points to the corresponding file stored in the directory. This method efficiently decreases the directory search time as the entire list will not be searched on every operation. Using the keys the hash table entries are checked and when the file is found it is fetched.

.

➢ **Characteristics of Directory Structure**

When a new file is created, then the entire list is checked whether the new file name is matching to a existing file name or not. In case, it doesn't exist, the file can be created at the beginning or at the end. Therefore, searching for a unique name is a big concern because traversing the whole list takes time.

The list needs to be traversed in case of every operation (creation, deletion, updating, etc) on the files therefore the systems become inefficient.

➢ **Operations On Directory**

a directory contains the entries of all the related files. For organizing the directory in the better way the user must be able to insert, delete, search, list the entries in the directory. Below we will be discussing the operation that can be performed on the directory.

1. Searching : A directory can be searched for a particular file or for another directory. It can also be searched to list all the files with the same name.

2. Creating : A new file can be created and inserted to the directory or new directory can be created keeping in mind that its name must be unique under that particular directory.

3. Deleting : If a file is no longer needed by the user, it can be deleted from the directory. The entire directory can also be deleted if it is not needed. An empty directory can also be deleted. When a directory is empty it is resembled by dot and dotdot.

4. List a directory : List of all the files in the directory can be retrieved and also the contents of the directory entry, for each file in a list. To read the list of all the files in the directory, it must be opened and after reading the directory must be closed to free up the internal tablespace.

5. Renaming : The name of the file or a directory represents the content it holds and its use. The file or directory can be renamed in case, the content inside or the use of file get changed. Renaming the file or directory also changes its position inside the directory.

6. Link : The file can be allowed to appear in more than one directory. Here, the system call creates a link between the file and the name specified by the path where the file is to appear.

7. Unlink : If the file is unlinked and is only present in one directory its directory entry is removed. If the file appears in multiple directories, only the link is removed.

➢ **Advantages of implementing directory in OS**

Following are the benefits of implementing Directory :

• Its implementation is very easy.

• If the files are smaller in size, searching will become faster.

• The operations like file creation, searching, deletion, updating are very easy in such a directory structure.

•Different users can have the same directory as well as the file name.

➢ **Disadvantages of implementing directory**

Here, are cons/drawbacks of implementing directory

•There may chance of name collision because two files can have the same name.

• Still, it not very scalable, two files of the same type cannot be grouped together in the same user.

• A user is not allowed to share files with other users.

• Searching will become time taking if the directory is large.

• This can not group the same type of files together.

➢ **Structure Of Directory**

When we were learning about priorities being 1. Single-level directory structure

Single level directory structure has only one directory which is called the root directory. The users are not allowed to create subdirectories under the root directory. All the files created by the several users are present in the root directory only.

There is one drawback of Single-level directory structure, a user cannot use the same file name used by another user in the system. Even if the file with the same name is created the old file will get destroyed first and replaced by the new file having the same name.

2. Two-level directory structure

In Two-level directory structure, the users create directory directly inside the root directory. But once a user creates such directory, further he cannot create any subdirectory inside that directory. Observe the figure below, 4 users have created their separate directory inside the root directory. But further, no subdirectory is created by the users.

This two-level structure allows each user to keep their files separately inside their own directory. This structure allows to use the same name for the files but under different user directories.

3. Hierarchical Directory Structure

In Hierarchical directory structure, the users can create directories under the root directory and can also create sub-directories under this structure. As the user is free to create many sub-directories, it can create different sub-directories for different file types.

Here, the files are accessed by their location using the path. There are two types of paths to locate the file in this directory structure

* Absolute Path

Here, the path for the desired file is described by considering the root directory as the base directory.

* Relative Path

Here, either the user’s directory is considered as the base directory or the desired file directory is considered as the base directory.

4. Tree Directory Structure

In a tree directory structure, except root directory, every directory or file has only one parent directory. So, there is a total separation between the users which provide complete naming freedom. Here, if a user wishes to access another users file, it has to go through two or more directories.

The tree directory structure provides an asymmetric way for the user to access the shared files of a different user. For example, a user can access a file of its own user directory with a shorter path than the other user.

5. Acyclic-Graph Directory Structure

This problem can be solved by the acyclic-graph directory structure. As this directory structure allows a directory or a file to have many parent directories. So, a shared file in a directory can be pointed by the other user directories who have access to that shared file using the links.

➢ **Reference:**

[**https://www.javatpoint.com/os-directory-implementation**](https://www.javatpoint.com/os-directory-implementation)

➢ **Links of GitHub accounts:**

1) https://github.com/ayushi-koul

2)<https://github.com/Tanishasoma29/Os-project>

3)