RV College of Engineering®
Mysore Road, RV Vidyaniketan Post,
Bengaluru - 560059, Karnataka, India

Disk Space Analyzer

OPERATING SYSTEM- CS235AI

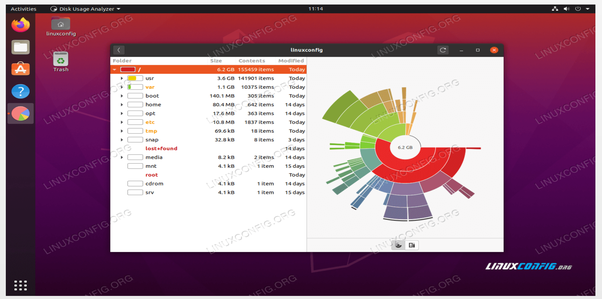
UNDER THE GUIDENCE OF
Dr.Jyoti Shetty (Assistant Professor)

VIDWATH H HOSUR - 1RV22CS231 , SHRINIWAS MAHESHWARI – 1RV22CS194

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

Abstract

A disk space analyzer is a crucial tool in operating systems designed to efficiently manage storage resources. It provides users with a visual representation of disk usage, identifying large files and directories that consume significant space. By presenting this information in a comprehensible format, disk space analyzers facilitate informed decision-making regarding file organization, cleanup, and storage optimization, ultimately enhancing system performance and resource utilization.



Introduction

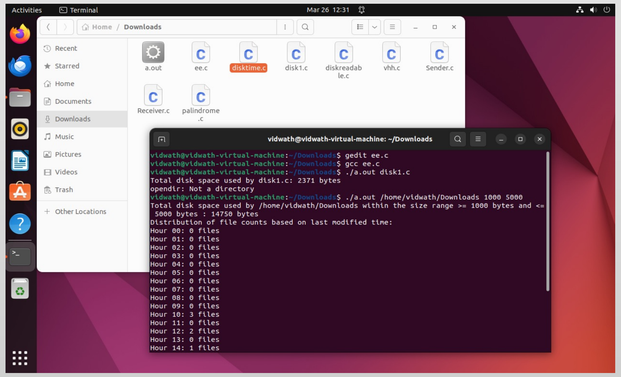
A Disk Space Analyzer is a utility tool that helps users understand how disk space is being utilized on their computer. It scans the hard drive (or a specific folder) and provides a detailed view of which folders and files are using space1. This allows users to make informed decisions about what to remove to quickly free up space. It aims to provide a comprehensive solution for analyzing and managing disk space usage on computer systems. The provided code is a C program designed to analyze disk space usage within a specified directory, optionally filtering files based on size constraints. It recursively traverses directories, calculating the total space consumed by files falling within the specified size range. Additionally, it analyzes the distribution of file counts based on their last modified time, providing insights into temporal patterns of file activity. This tool offers valuable functionality for understanding and managing storage resources efficiently within an operating system environment.

Working and Output

. It calculates the total disk space consumed by files within the specified directory, considering optional size constraints. Through the `calculateDiskSpace` function, it iterates through directories and files, summing up their sizes based on the provided criteria. Additionally, the program analyzes file counts based on their last modified time, providing insights into temporal patterns of file activity. This is achieved via the `analyzeLastModifiedTime` function, which counts files modified within each hour of the day. Overall, the program offers users a comprehensive understanding of disk usage and file modification trends, aiding in efficient storage management and system optimization.

Outputs of the code:

• It first prints a usage message if the number of command-line arguments is incorrect, explaining how to use the program.
•Then, it calculates and prints the total disk space used by files within the specified directory.
•it displays the size range filter used for the calculation.
•it analyzes and prints the distribution of file counts based on their last modified time, showing the number of files modified within each hour of the day.
•Finally, it terminates successfully, returning 0 as the exit status.



Implementation

1. **Directory Traversal and File System Monitoring:** Implement functions to traverse the directory structure recursively. Use system calls or platform-specific APIs to monitor file system events such as file creations, modifications, and deletions.
2. **Data Collection and Analysis:** Collect information about directories, subdirectories, and files, including their sizes and attributes. Aggregate disk space usage statistics for each directory and file. 3. User Interface: Develop a user-friendly interface to interact with the Disk Space Analyzer. Include options for users to start the analysis, view results, and configure settings. Design intuitive displays to present disk space usage information, such as directory hierarchies, file sizes, and trends
3. **Identification of Large Files and Directories:** Define thresholds to identify large files and directories based on size. Implement algorithms to flag files and directories that exceed specified thresholds.

```
shriniwas@shriniwas:~/Downloads$ gcc -o code0 code0.c
shriniwas@shriniwas:~/Downloads$ ./code0 /home/shriniwas/Downloads
Total disk space: 415.7 KB
Free disk space: 27.7 GB
Average file size: 17026.44 bytes
Number of files: 25
Number of directories: 1
Number of symbolic links: 0
Number of hard links: 3
Largest 5 files in /home/shriniwas/Downloads:
/home/shriniwas/Downloads/code0 71448 bytes
/home/shriniwas/Downloads/disk.c        3687 bytes
/home/shriniwas/Downloads/dsa.c 4766 bytes
/home/shriniwas/Downloads/cp.c  800 bytes
/home/shriniwas/Downloads/ls.c  1231 bytes
/home/shriniwas/Downloads/disk_space_analyzer.c 3336 bytes
/home/shriniwas/Downloads/2.c   2245 bytes
/home/shriniwas/Downloads/mv.c  552 bytes
/home/shriniwas/Downloads/a.out 71120 bytes
/home/shriniwas/Downloads/analyser.c    5782 bytes
/home/shriniwas/Downloads/a.c   271 bytes
/home/shriniwas/Downloads/2.txt 7447 bytes
/home/shriniwas/Downloads/code0.c       7733 bytes
/home/shriniwas/Downloads/code3.c       5622 bytes
/home/shriniwas/Downloads/binary.c      1054 bytes
/home/shriniwas/Downloads/shriniwas.c   3687 bytes
/home/shriniwas/Downloads/analyser      71336 bytes
/home/shriniwas/Downloads/rm.c  138 bytes
/home/shriniwas/Downloads/code1.c       3687 bytes
/home/shriniwas/Downloads/1.c   9589 bytes
/home/shriniwas/Downloads/file1.c       2332 bytes
/home/shriniwas/Downloads/analyer       71336 bytes
/home/shriniwas/Downloads/file1 71072 bytes
/home/shriniwas/Downloads/ls.cp 1231 bytes
/home/shriniwas/Downloads/lol.txt       63 bytes
shriniwas@shriniwas:~/Downloads$
```

Working & Future Scope

•**File Type Analysis**: Incorporating functionality to categorize files based on their types (e.g., documents, images, executables) and presenting statistics accordingly could provide users with more detailed insights into their storage usage patterns.
•**Integration with Cloud Storage**: Extending the program's functionality to analyze disk space usage across cloud storage platforms or network-attached storage (NAS) devices could cater to users with diverse storage environments.
•**Enhanced User Interface**: Implementing a more user-friendly interface, such as a graphical user interface (GUI) or interactive command-line interface (CLI), could improve usability and make the tool more accessible to users.
•**Visualization**: Introducing graphical representations, such as charts or graphs, to visualize disk space usage trends and file modification patterns could enhance the program's effectiveness in conveying information to users.
•**Performance Optimization**: Optimizing the program's algorithms and data structures to improve performance, especially for large directories or systems with numerous files, could make the analysis process more efficient.

Conclusion

In conclusion, the provided C program serves as a functional disk space analyzer, offering users the ability to assess disk space usage and file modification patterns within a specified directory. By calculating total disk space utilization and analyzing file counts based on last modified time, the program provides valuable insights into storage consumption and temporal file activity. While the program's current implementation fulfills its core objectives, performance optimization, and additional analysis metrics could further augment its utility and effectiveness. Overall, this disk space analyzer represents a valuable tool for users seeking to manage and optimize their storage resources efficiently.