USER'S MANUAL

OBJECT-ORIENTED PROGRAMMING
COMP 2021

Comp Virtual File System (CVFS)

Prepared by: Johannes Christian Koeleman, Chekanov Trofim, Tang Kin Wah

November, 2024

Revision Sheet

Release No.	Date	Revision Description
Rev. 0	18/11/24	User’s Manual Created
Rev. 1	20/11/24	Added advanced features (criteria and save/load)

USER'S MANUAL

TABLE OF CONTENTS

1.0 GENERAL INFORMATION

1.0 GENERAL INFORMATION

1.1 System Overview

The Comp Virtual File System (CVFS) is an in-memory virtual file system designed to provide functionality similar to real-world file systems. It allows users to:

- Manage virtual disks.
- Create and organize files and directories.
- Define and apply search criteria.
- Save and load virtual disks for persistence.

This document explains how to use CVFS commands effectively.

1.2 Acronyms and Abbreviations

CVFS - Comp Virtual File System.

CLI – Command-Line Interface.

JDK – Java Development Kit.

JRE – Java Runtime Environment.

2.0 SYSTEM SUMMARY

2.0 SYSTEM SUMMARY

2.1 System Configuration

Platform: Java 9 or later required.
Components:

- `Application.java`: Main entry point for the CLI.
- `model/*.java`: Core classes for managing the virtual file system.

3.0 GETTING STARTED

3.0 GETTING STARTED

3.1 Installation

Install Java JDK 9 or later.
USE command `git clone` https://github.com/regitxx/CVFSproject to locally install the project on your machine. The main file that combines everything is ‘**Applicaton.java’** if you are using **IDE** you may just run by clicking the run button for the **Application.java** file **OTHERWISE** run this inside your projects terminal**:**

Compile the project:
```
javac -d out Application.java model/*.java
```

Run the CLI:
```
java -cp out hk.edu.polyu.comp.comp2021.cvfs.Application
```

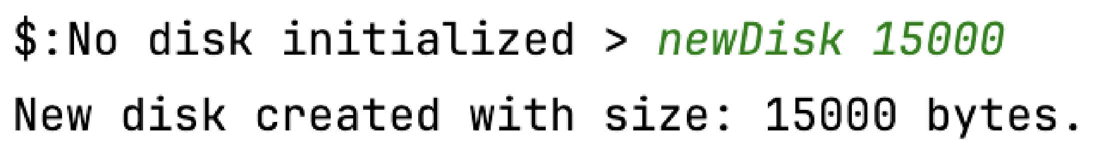
4.0 USING THE CLIENT APPLICATION

4.0 COMMAND LIST AND EXAMPLES

4.1 Disk Management

4.1.1 Create a New Disk

- **Command**: `newDisk <diskSize>`
- **Description**: Creates a virtual disk with the specified maximum size (in bytes).
- **Example**:



```
$:No disk initialized > newDisk 15000
New disk created with size: 15000 bytes.
```

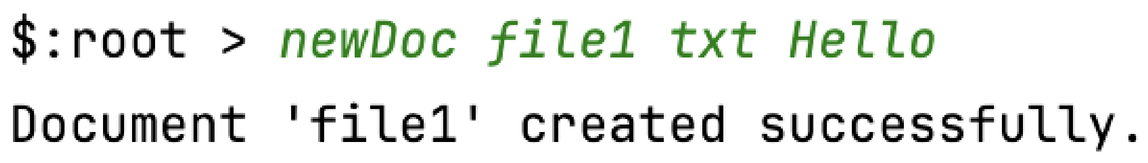
4.2 File and Directory Operations

4.2.1 Create a New Document

Command: `newDoc <docName> <docType> <docContent>`

Description: Creates a new document in the current directory. Supported types: `txt`, `java`, `html`, `css`.

Example:



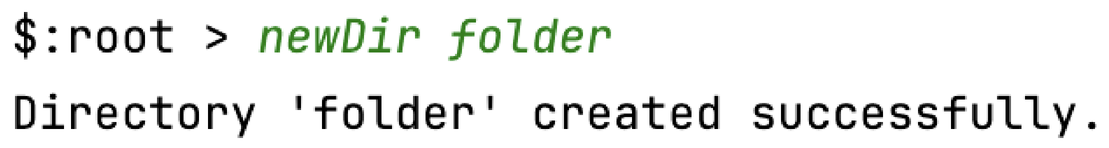
```
$:root > newDoc file1 txt Hello
Document 'file1' created successfully.
```

4.2.2 Create a New Directory

Command: `newDir <dirName>`

Description: Creates a new directory in the current directory.

Example:



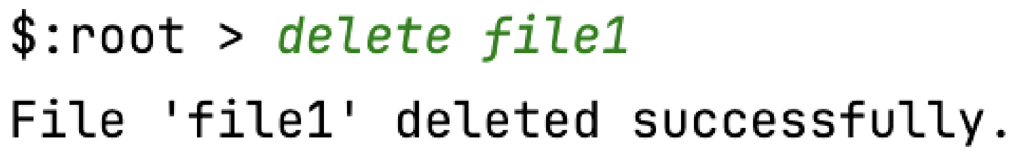
```
$:root > newDir folder
Directory 'folder' created successfully.
```

4.2.3 Delete a File or Directory

Command: `delete <fileName>`

Description: Deletes the specified file or directory from the current directory.

Example:



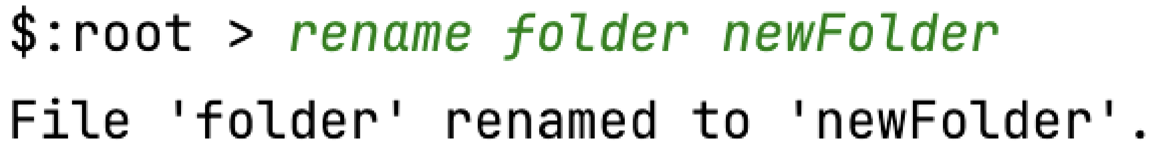
```
$:root > delete file1
File 'file1' deleted successfully.
```

4.2.4 Rename a File or Directory

Command: `rename <oldName> <newName>`

Description: Renames the specified file or directory.

Example:



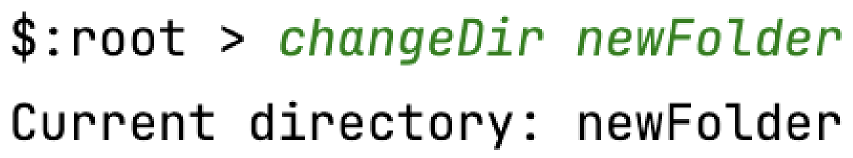
```
$:root > rename folder newFolder
File 'folder' renamed to 'newFolder'.
```

4.2.4 Change Directory

Command: `changeDir <dirName>`

Description: Changes the current directory. Use `..` to move to the parent directory.

Example:



```
$:root > changeDir newFolder
Current directory: newFolder
```

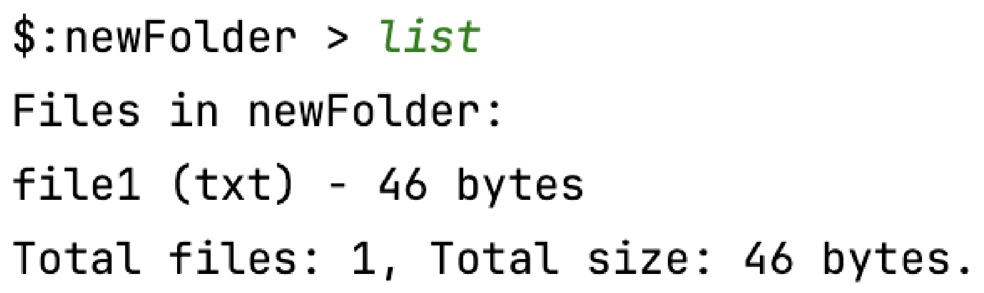
4.3 Listing Files

4.3.1 List Files in the Current Directory

Command: `list`

Description: Lists all files and directories in the current directory

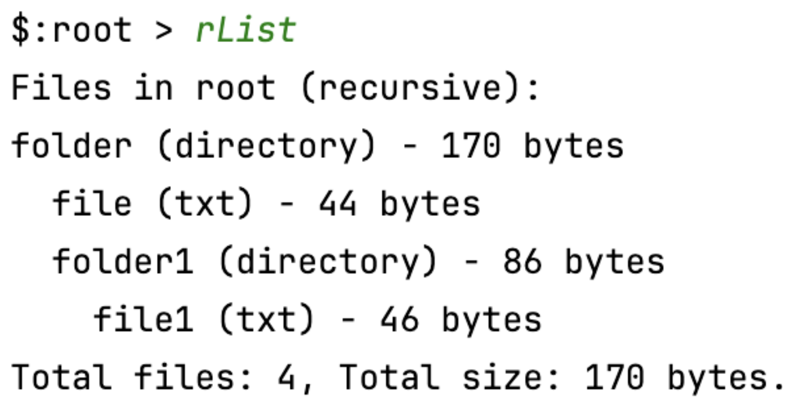
Example:



```
$:newFolder > list
Files in newFolder:
file1 (txt) - 46 bytes
Total files: 1, Total size: 46 bytes.
```

4.3.2 Recursively List Files

Command: `rList`
Description: Recursively lists all files and directories, showing their hierarchy.
Example:



```
$:root > rList
Files in root (recursive):
folder (directory) - 170 bytes
  file (txt) - 44 bytes
  folder1 (directory) - 86 bytes
    file1 (txt) - 46 bytes
Total files: 4, Total size: 170 bytes.
```

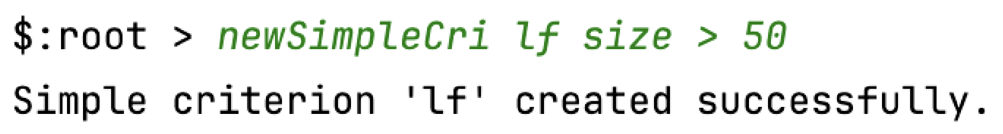
4.4 Criteria Management

4.4.1 Add a Simple Criterion

Command: `newSimpleCri <criName> <attrName> <op> <val>`
Description: Adds a search criterion based on file attributes.

- **Attributes:** `name`, `type`, `size`.
- **Operators:**
  - **For** `name`: `contains`.
  - **For** `type`: `equals`.
  - **For** `size`: `>`, `<`, `>=`, `<=`, `==`, `!=`.

Example:



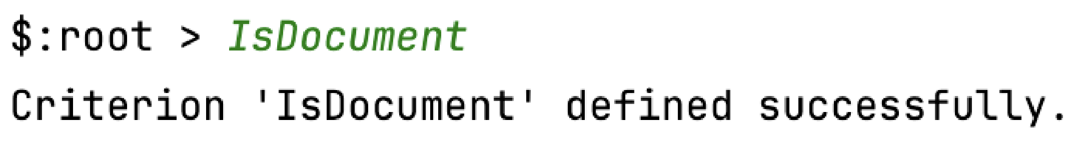
```
$:root > newSimpleCri lf size > 50
Simple criterion 'lf' created successfully.
```

4.4.2 IsDocument

Command: `IsDocument`

Description: Defines Criterion isDocument, can be used only once.

Example:



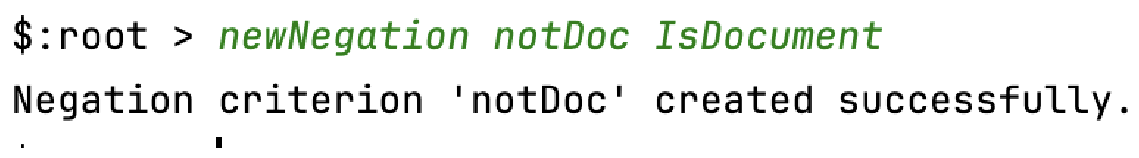
```
$:root > IsDocument
Criterion 'IsDocument' defined successfully.
```

4.4.3 Add a Negation Criterion

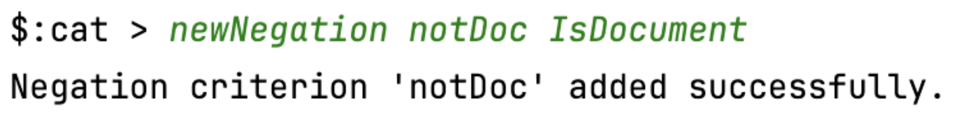
Command: `newNegation <criName1> <criName2>`

Description: Creates a negation of an existing criterion.

Example:



```
$:root > newNegation notDoc IsDocument
Negation criterion 'notDoc' created successfully.
```



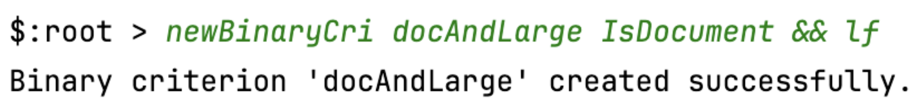
```
$:cat > newNegation notDoc IsDocument
Negation criterion 'notDoc' added successfully.
```

4.4.4 Add a Composite Criterion

Command: `newBinaryCri <criName1> <criName3> <logicOp> <criName4>`

Description: Combines two criteria using logical operators (`&&` or `||`).

Example:



```
$:root > newBinaryCri docAndLarge IsDocument && lf
Binary criterion 'docAndLarge' created successfully.
```

4.4.5 Search Files

Command: `search <criName>`

Description: Searches files in the current directory using a criterion.

Example:

```
$:root > search docAndLarge
Files in the working directory that satisfy 'docAndLarge':
Total files: 0, Total size: 0 bytes.
```

4.4.6 Recursively Search Files

Command: `rSearch <criName>`

Description: Recursively searches files using a criterion.

Example:

```
$:root > rSearch docAndLarge
Files in the working directory (recursive) that satisfy 'docAndLarge':
Total files: 0, Total size: 0 bytes.
```

4.4.7 Print All Criteria

Command: `printAllCriteria`

Description: Lists all defined criteria.

Example:

```
$:root > printAllCriteria
Defined criteria:
notDoc: NOT (IsDocument)
IsDocument: IsDocument
lf: lf (size > 50)
docAndLarge: (IsDocument && lf (size > 50))
```

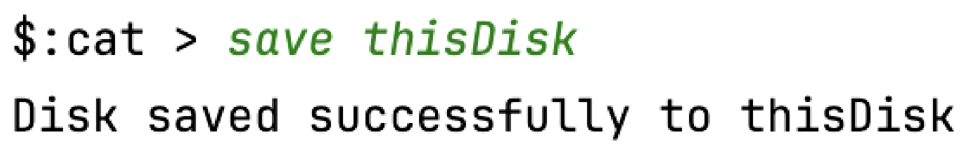
4.5 Save and Load Disk / QUIT

4.5.1 Save Disk

Command: `save <path>`

Description: Saves the virtual disk to the specified path.

Example:



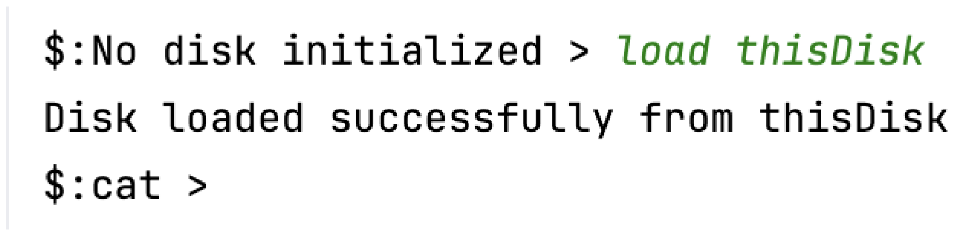
```
$:cat > save thisDisk
Disk saved successfully to thisDisk
```

4.5.2 Load Disk

Command: `load <path>`

Description: Loads the virtual disk from the specified path.

Example:



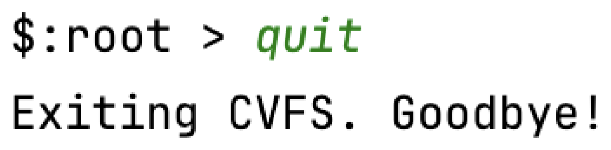
```
$:No disk initialized > load thisDisk
Disk loaded successfully from thisDisk
$:cat >
```

4.5.3

Command: `quit`

Description: Quits the program

Example:



```
$:root > quit
Exiting CVFS. Goodbye!
```

5.0 TROUBLESHOOTING

5.0 TROUBLESHOOTING

"Cannot find file or directory": Verify the name using `list`.

"Not enough space on the disk": Delete unused files or increase disk size.

"Error loading disk": Ensure the file exists and is a valid disk file.