VCS - A Simple Version Control System

A lightweight version control system implemented in C that handles file versioning, metadata tracking, and rollback operations.

Features

- **Repository Initialization**: Create new repositories with proper directory structure
- **File Check-in**: Version files with optional comments and automatic metadata generation
- **File Check-out**: Retrieve specific versions of files
- **Version Listing**: Display all versions of a file with timestamps and metadata
- **Rollback Functionality**: Restore files to previous versions
- **Hash-based Integrity**: File integrity verification using custom hash function
- **Metadata Persistence**: Comprehensive version tracking with file-based storage

Components

1. **Repository Management** (`repo.c`)
   - Initialize new repositories
   - Load existing repository metadata
   - Manage repository lifecycle
2. **File Operations** (`fileops.c`)
   - File copying and version storage
   - Hash generation for integrity checking
   - Version file creation and restoration
3. **Version Management** (`version.c`)
   - Check-in/check-out operations
   - Version listing and comparison
   - Rollback functionality
4. **Metadata Operations** (`metadata.c`)
   - Persistent storage of version information
   - File version tracking and retrieval
   - Repository state management
5. **Utility Functions** (`utils.c`)
   - Helper functions for file system operations
   - User interface and help system

Directory Structure

```
project_directory/
├── .vcs/                 # VCS metadata directory
│   ├── versions/           # Versioned file storage
│   │   └── filename/       # Per-file version directories
│   │       ├── v1          # Version 1 of file
│   │       ├── v2          # Version 2 of file
│   │       └── ...
│   ├── temp/               # Temporary operations
│   ├── versions.meta       # Metadata file
│   └── current.info        # Current state information
└── other_files...          # Working directory
```

Requirements

- GCC compiler
- Linux environment
- Make utility

Build

```
make clean && make all
```

Usage

Initialize a Repository

```
./vcs init
```

Check in a File

```
# Basic check-in
./vcs checkin myfile.txt

# Check-in with comment
./vcs checkin myfile.txt "Initial version of my file"
```

Check out a File

```
# Check out latest version
./vcs checkout myfile.txt
```

```
# Check out specific version
./vcs checkout myfile.txt 2
```

List File Versions

```
./vcs list myfile.txt
```

Rollback to Previous Version

```
./vcs rollback myfile.txt 1
```

Example Workflow

```
# Initialize repository
./vcs init

# Create a test file
echo "Hello World" > test.txt

# Check in initial version
./vcs checkin test.txt "Initial version"

# Modify the file
echo "Hello World - Modified" > test.txt

# Check in modified version
./vcs checkin test.txt "Added modification"

# List all versions
./vcs list test.txt

# Rollback to version 1
./vcs rollback test.txt 1

# Verify rollback
cat test.txt
```

Metadata File Format

```
# VCS Metadata File
TOTAL_VERSIONS=3

# File Versions
FILE=test.txt|VERSION=3|TIMESTAMP=1749809022|SIZE=12|HASH=51f2700345be361790
```

```
22|COMMENT=Rollback to version 1
FILE=test.txt|VERSION=1|TIMESTAMP=1749808953|SIZE=12|HASH=51f2700345be361789
53|COMMENT=Initial version
FILE=test.txt|VERSION=2|TIMESTAMP=1749808985|SIZE=23|HASH=6dff0ec486be5ff089
85|COMMENT=Added modification
```

Limitations

- Uses djb2 hash algorithm instead of cryptographic hashes (like SHA-256)
- No network functionality
- No differential storage (each version is complete copy)
- No encryption or advanced security features
- Limited to single-user scenarios