Report

* The program lists files in a specified directory using the C++ filesystem library.
* It handles multiple directories as command-line arguments.
* Error handling is implemented for invalid directories or permission issues.

Functionality

The program defines a function list\_files that takes a directory path as input and iterates through its contents. It prints each filename found in the directory.

void list\_files(const string& directory) { try { for (const fs::directory\_entry& entry : fs::directory\_iterator(directory)) { cout << entry.path().filename().string() << endl; } } catch (const fs::filesystem\_error& exception\_error) { cerr << "Error: " << exception\_error.what() << endl; } }

Error Handling

* If no directory is provided, the program outputs an error message.
* If a directory does not exist or cannot be accessed, an exception is caught and an error is displayed.

int main(int files\_count, char\* files[]) { if (files\_count < 2) { cerr << "Error: No directory provided. Please enter a valid directory." << endl; return 1; } for (int i = 1; i < files\_count; i++) { cout << files[i] << ":" << endl; list\_files(files[i]); cout << endl; } return 0; }

Execution

* Compile the program using a C++ compiler with filesystem support (C++17 or later).
* Run the executable, providing directory paths as arguments.
* Example usage: ./program\_name /path/to/directory1 /path/to/directory2

The program will print the filenames of each directory provided as input.

Source Code

#include <iostream>

#include <filesystem>

using namespace std;

namespace fs = filesystem;

void list\_files(const string& directory) {

try {

for (const fs::directory\_entry& entry : fs::directory\_iterator(directory)) {

cout << entry.path().filename().string() << endl;

}

}

catch (const fs::filesystem\_error& exception\_error) {

cerr << "Error: " << exception\_error.what() << endl;

}

}

int main(int files\_count, char\* files[]) {

if (files\_count < 2) {

cerr << "Error: No directory provided. Please enter a valid directory." << endl;

return 1;

}

for (int i = 1; i < files\_count; i++) {

cout << files[i] << ":" << endl;

list\_files(files[i]);

cout << endl;

}

return 0;

}