Mansor M. Malik Platform technologies Lab\_Exe 1 Directory Management System

#include <iostream>

#include <string>

#include <vector>

#include <algorithm>

#ifdef \_unix\_

#include <dirent.h>

#include <sys/stat.h>

#include <unistd.h>

#include <limits.h>

#elif \_WIN32

#include <windows.h>

#include <string>

#endif

void listFiles(const std::string& directoryPath, int listOption) {

std::vector<std::string> files;

#ifdef \_unix\_

DIR\* dir = opendir(directoryPath.c\_str());

if (dir == nullptr) {

std::cerr << "Error opening directory: " << directoryPath << std::endl;

return;

}

struct dirent\* entry;

while ((entry = readdir(dir)) != nullptr) {

if (entry->d\_name[0] != '.') {

files.push\_back(entry->d\_name);

}

}

closedir(dir);

#elif \_WIN32

std::string searchPath = directoryPath + "\\\*";

WIN32\_FIND\_DATA findFileData;

HANDLE hFind = FindFirstFile(searchPath.c\_str(), &findFileData);

if (hFind == INVALID\_HANDLE\_VALUE) {

std::cerr << "Error opening directory: " << directoryPath << std::endl;

return;

}

do {

if (findFileData.cFileName[0] != '.') {

files.push\_back(findFileData.cFileName);

}

} while (FindNextFile(hFind, &findFileData) != 0);

FindClose(hFind);

#endif

if (listOption == 1) {

for (const auto& file : files) {

std::cout << file << std::endl;

}

} else if (listOption == 2) {

std::string extension;

std::cout << "Enter file extension (e.g., txt): ";

std::cin >> extension;

for (const auto& file : files) {

if (file.substr(file.find\_last\_of('.') + 1) == extension) {

std::cout << file << std::endl;

}

}

} else if (listOption == 3) {

std::string pattern;

std::cout << "Enter file name pattern (e.g., moha\*.\*): ";

std::cin >> pattern;

for (const auto& file : files) {

if (file.find(pattern) != std::string::npos) {

std::cout << file << std::endl;

}

}

} else {

std::cerr << "Invalid option!" << std::endl;

}

}

void createDirectory(const std::string& directoryPath) {

#ifdef \_unix\_

if (mkdir(directoryPath.c\_str(), 0755) != 0) {

std::cerr << "Error creating directory: " << directoryPath << std::endl;

} else {

std::cout << "Directory created: " << directoryPath << std::endl;

}

#elif \_WIN32

if (CreateDirectory(directoryPath.c\_str(), nullptr) || GetLastError() == ERROR\_ALREADY\_EXISTS) {

std::cout << "Directory created: " << directoryPath << std::endl;

} else {

std::cerr << "Error creating directory: " << directoryPath << std::endl;

}

#endif

}

void changeDirectory(const std::string& directoryPath) {

#ifdef \_unix\_

if (chdir(directoryPath.c\_str()) != 0) {

std::cerr << "Error changing directory to: " << directoryPath << std::endl;

} else {

std::cout << "Changed working directory to: " << directoryPath << std::endl;

}

#elif \_WIN32

if (SetCurrentDirectory(directoryPath.c\_str())) {

std::cout << "Changed working directory to: " << directoryPath << std::endl;

} else {

std::cerr << "Error changing directory to: " << directoryPath << std::endl;

}

#endif

}