Koncepcja zadanie 6

Mateusz Ostaszewski

# Struktura Projektu

1. Plik główny aplikacji (main.cpp):

* Punkt wejścia programu.
* Obsługa argumentów linii poleceń.
* Wywołanie odpowiednich funkcji w zależności od wprowadzonych poleceń.

1. Moduł zarządzania dyskiem wirtualnym (VirtualDiskManager.cpp, VirtualDiskManager.h):

* Funkcje do tworzenia, usuwania, modyfikacji plików i katalogów na wirtualnym dysku.
* Zarządzanie strukturą katalogów.
* Obsługa dowiązań twardych.

1. Moduł operacji na plikach (FileManager.cpp, FileManager.h):

* Kopiowanie plików między dyskiem systemowym a wirtualnym.
* Dodawanie i usuwanie bajtów z plików.
* Wyświetlanie informacji o plikach i katalogach.

# Koncepcja realizacji kolejnych funkcji

1. Tworzenie Wirtualnego Dysku

* Moduł: VirtualDiskManager
* Funkcja: createVirtualDisk(const std::string& diskName, size\_t size)
* Opis: Tworzy plik binarny reprezentujący wirtualny dysk, z inicjalizacją jego struktury i alokacją określonej przestrzeni.

1. Kopiowanie Pliku z Dysku Systemu na Dysk Wirtualny

* Moduł: FileManager
* Funkcja: copyFileToVirtualDisk(const std::string& source, const std::string& destination)
* Opis: Odczytaj plik z systemowego systemu plików używając standardowych operacji wejścia/wyjścia, a następnie zapisz jego zawartość do wirtualnego dysku. Zarezerwuje odpowiednią ilość miejsca na wirtualnym dysku, zapisując metadane pliku (takie jak nazwa, rozmiar, lokalizacja)

1. Tworzenie Katalogu na Dysku Wirtualnym

* Moduł: VirtualDiskManager
* Funkcja: createDirectory(const std::string& path)
* Opis: Analizuje ścieżkę katalogu i rekurencyjnie tworzy każdy wymagany poziom katalogów w strukturze wirtualnego dysku. Każdy katalog będzie miał wpis w systemie plików wirtualnego dysku z metadane.

1. Usuwanie Katalogu z Dysku Wirtualnego

* Moduł: VirtualDiskManager
* Funkcja: removeDirectory(const std::string& path)
* Opis: Rekurencyjnie usuwa katalog i wszystkie jego zawartości. Przeszukuje wszystkie pliki i podkatalogi wewnątrz usuwanego katalogu, usuwając każdy z nich i zwalniając przypisaną im przestrzeń na wirtualnym dysku.

1. Kopiowanie Pliku z Dysku Wirtualnego na Dysk Systemu

* Moduł: FileManager
* Funkcja: copyFileToSystem(const std::string& source, const std::string& destination)
* Opis: lokalizuje plik na wirtualnym dysku, odczytuje jego dane, a następnie zapisuje te dane jako plik w systemowym systemie plików.

1. Wyświetlanie Katalogu Dysku Wirtualnego

* Moduł: FileManager
* Funkcja: displayDirectoryInfo(const std::string& path)
* Opis: Wyświetla informacje o katalogu, w tym rozmiar plików w katalogu, w podkatalogach oraz ilość wolnej pamięci na dysku.

1. Tworzenie Twardego Dowiązania

* Moduł: VirtualDiskManager
* Funkcja: createHardLink(const std::string& source, const std::string& linkName)
* Opis: Tworzy nowy wpis w systemie plików wirtualnego dysku, który wskazuje na tę samą lokalizację co oryginalny plik lub katalog.

1. Usuwanie Pliku lub Dowiązania z Wirtualnego Dysku

* Moduł: VirtualDiskManager
* Funkcja: removeFile(const std::string& fileName)
* Opis: Usuwa plik lub dowiązanie z wirtualnego dysku.

1. Dodanie do Pliku o Zadanej Nazwie n Bajtów

* Moduł: FileManager
* Funkcja: addBytesToFile(const std::string& filename, size\_t bytes)
* Opis: Dodaje określoną liczbę bajtów do pliku na wirtualnym dysku.

1. Skrócenie Pliku o Zadanej Nazwie o n Bajtów

* Moduł: FileManager
* Funkcja: removeBytesFromFile(const std::string& filename, size\_t bytes)
* Opis: Usuwa określoną liczbę bajtów z końca pliku na wirtualnym dysku.

1. Wyświetlenie Informacji o Zajętości Dysku

* Moduł: VirtualDiskManager
* Funkcja: displayDiskUsage()
* Opis: Wyświetla ogólne informacje o wykorzystaniu dysku, w tym zajętości i wolnej przestrzeni.

# Implementacja

Pliki nagłówkowe do VirtualDiskManager i FileManager zostały zaimplementowane.  
  
Koncepcja funkcji VirtualDiskManager::createVirtualDisk  
  
#include <iostream>

#include <fstream>

#include <cstring>

struct DiskHeader {

char signature[4];

uint32\_t blockSize;

uint32\_t totalBlocks;

};

void createVirtualDisc(const std::string& diskName, uint32\_t blockSize, uint32\_t totalBlocks) {

std::ofstream diskFile(diskName, std::ios::binary | std::ios::out);

if (!diskFile) {

std::cerr << "File can not be opened!" << std::endl;

return;

}

DiskHeader header;

std::strcpy(header.signature, "VDISK");

header.blockSize = blockSize;

header.totalBlocks = totalBlocks;

diskFile.write(reinterpret\_cast<const char\*>(&header), sizeof(header));

char\* emptyBlock = new char[blockSize];

std::memset(emptyBlock, 0, blockSize);

for (uint32\_t i = 0; i < totalBlocks; ++i) {

diskFile.write(emptyBlock, blockSize);

}

delete[] emptyBlock;

diskFile.close();

std::cout << "Virtual Disc have been created." << std::endl;

}