

# System Plików

Zuzanna Santorowska

May 2020

## Treść Zadania

Należy napisać w środowisku systemu Minix program w języku C (oraz skrypt demonstrujący wykorzystanie tego programu) realizujący podstawowe funkcje systemu plików.

System plików należy zorganizować w dużym pliku o zadanej wielkości, który będzie "wirtualnym dyskiem". Program powinien tworzyć dysk wirtualny, oraz dokonywać zapisów i odczytów w celu zrealizowania podstawowych operacji na dysku, związanych z zarządzaniem katalogiem, alokacją plików oraz utrzymywaniem unikalności nazw.

W pliku na dysku należy zorganizować system plików z jednopoziomym katalogiem. Elementem katalogu jest opis pliku, zawierający co najmniej nazwę, wielkość i sposób rozmieszczenia pliku na wirtualnym dysku. Należy zaimplementować następujące operacje, dostępne dla użytkownika programu:

- wyświetlenie zestawienia z aktualną mapą zajętości wirtualnego dysku - czyli listy kolejnych obszarów wirtualnego dysku z opisem: adres, typ obszaru, rozmiar, stan (np. dla bloków danych: wolny/zajęty).
- usuwanie wirtualnego dysku
- usuwanie pliku z wirtualnego dysku
- wyświetlanie katalogu dysku wirtualnego
- kopiowanie pliku z dysku wirtualnego na dysk systemu Minix
- kopiowanie pliku z dysku systemu Minix na dysk wirtualny
- tworzenie wirtualnego dysku

Program ma kontrolować wielkość dostępnego miejsca na wirtualnym dysku i pojemność katalogu, reagować na próby przekroczenia tych wielkości.

Nie trzeba realizować funkcji otwierania pliku ani czytania/pisania fragmentów pliku.

Nie trzeba realizować funkcji związanych z współbieżnym dostępem. Zakłada się dostęp sekwencyjny i wyłączny do wirtualnego dysku.

Należy przygotować demonstrację (zgrupowanie serii poleceń w postaci skryptu interpretera sh) prezentująca słabe i silne strony przyjętego rozwiązania w kontekście ewentualnych zewnętrznej i wewnętrznej fragmentacji.

Uwaga: Do zdobycia 2 punkty za wejściówkę oraz 6 za projekt.

## Koncepcja

Jest to bardzo prosty system plików który posiada następujące elementy:

1. Super Block  
Przechowuje informacje na temat systemu plików takie jak: rozmiar, ilość bloków, rozmiar przestrzeni użytkownika, jaka część przestrzeni użytkownika jest obecnie w użyciu oraz gdzie znajduje się pierwszy INode.
2. Mapa INodów  
Zawarte są w niej informacje na temat tego które INode są obecnie zajęte.
3. Mapa Bloków  
Zawarte są w niej informacje na temat tego które Bloki są obecnie zajęte.
4. Sekcja INode  
Dane które przechowują kolene INode
5. Sekcja Bloków  
Dane które przechowują kolene Bloki

Bloki zawierają dane oraz wskaźniki na kolejny blok przechowujący dane pliku. Dzięki temu bloki danych pliku nie muszą być przechowywane po kolei a co za tym idzie fragmentacja zewnętrzna nie występuje. Fragmentacja wewnętrzna da się zmniejszyć tylko zmniejszając rozmiar bloków.

## Uruchamianie

Plik kompiluje się jak zwykły plik c

```
cc FileSystem.c
```

Uruchomienie skompilowanego programu wymaga użycia odpowiednich argumentów.

- Tworzenie Systemu plików  
Literka c oznacza create. Argument size oznacza rozmiar systemu plików  

```
c size
```
- Zapisywanie do systemu plików  
Literki st oznaczają save to. Argument filename oznacza nazwę pliku który chcemy zapisać w systemie plików

`st filename`

- Ekstrachowanie z systemu plików  
Literki sf oznaczają save from. Argument filename oznacza nazwę pliku który chcemy wyekstrahować z systemu plików

`sf filename`

- Usówanie pliku  
Literka r oznacza remove. Argument filename oznacza nazwę pliku który chcemy usunąć

`r filename`

- Wylistuj wszystkie pliki które się znajdują w systemie plików  
Literka l oznacza list

`l`

- Wypisz informacje na temat stanu pamięci  
Literka m oznacza memory

`m`

- Usuń system plików  
Literka d oznacza delete

`d`

Przykładowe wywołanie

`./a.out c 100000`