Etap	1	2	3	4	Suma
Punkty	4	4	5	3	16
Wynik					

L1: Dikaiopolis

Dikaiopolis jest obywatelem Aten żyjącym w V wieku p.n.e. Nie mieszka jednak w Atenach, tylko za miastem—jest ziemianinem. Ponieważ w tamtych czasach nie było internetu, a transport do miast i ich sklepów był bardzo utrudniony, Dikaiopolis nie może łatwo znaleźć dobrego programu do zarządzania plikami. Potrzebuje jednak takiego programu na swoim komputerze. Dikaiopolis jednak nie ma czasu, by napisać go samemu, gdyż pracuje często na swoim polu. Napisz dla niego aplikację według jego wymagań.

Do zadania dostarczony jest wstępny plik ze szkieletem programu do napisania. Kod poszczególnych etapów umieść w odpowiednich funkcjach. Dozwolone jest tworzenie funkcji pomocniczych i wywoływanie ich z `*_stageX`, a także wywoływanie `*_stageX` z `*_stageY`. **Niedozwolone** jest natomiast modyfikowanie nagłówków tych funkcji i modyfikowanie kodu w funkcji `main`.

W każdym etapie na bieżąco zwalniaj wszystkie niepotrzebne zasoby, czyli między innymi zamykaj nieużywane już pliki. Jeśli dana funkcja systemowa może zwrócić błąd, to należy sprawdzić, czy jej wykonanie się powiodło (jeśli nie, to wystarczy zamknąć program i wyświetlić informacje, że wystąpił błąd).

Interfejs CLI

Wyświetl listę następujących operacji:

1. `show`
2. `write`
3. `walk`
4. `exit`

a następnie oczekuj na wejście od użytkownika. Użytkownik powinien móc wpisać numer jednego z powyższych poleceń. Jeśli użytkownik wpisze coś innego, wyświetl informacje o błędnym poleceniu i zwróć 1 z `interface_stage1` .

Jeśli polecenie będzie poprawne, wczytaj z `stdin` ścieżkę, a następnie sprawdź, czy prowadzi ona do istniejącego pliku (użyj funkcji `stat`). Jeśli nie, wyświetl błąd i zwróć 1 z `interface_stage1`. Jeśli odczytana ścieżka prowadzi do pliku, to zależnie od wybranego polecenia wywołaj odpowiadającą mu funkcję (`*_stageX`) z odpowiednimi argumentami, a zaraz po wywołaniu zwróć 1 z `interface_stage1`. Wyjątek stanowi polecenie `exit`. W jego przypadku natychmiast zwróć 0 z `interface_stage1`.

Zawartość katalogów i plików

Funkcja `show_stage2` działa zarówno dla katalogów, jak i dla plików. Jeśli jej argumenty wskazują na plik, to wyświetl **rozmiar pliku w bajtach** oraz **zawartość pliku**. Jeśli argumenty funkcji wskazują na katalog, to wypisz **nazwy plików** w katalogu. Jeśli plik ma inny typ niż katalog lub zwykły plik, wyświetl informację, że jest to nieznany typ pliku.

Zapisywanie do plików

Funkcja `write_stage3` dopisuje na koniec podanego pliku nową zawartość. W przypadku tej funkcji do wyświetlania i zapisu pliku użyj funkcji niskopoziomowych, t.j. `open`, `write`, `read` i `close`. Przydatne mogą okazać się zdefiniowane (i opisane w tutorialu) funkcje `bulk_read` i `bulk_write`. Po wejściu do funkcji wyświetl obecną zawartość pliku, a następnie oczekuj na wejście od użytkownika. Każdą nową linijkę dopisuj na **koniec** otwartego pliku (wskazówka: `O_APPEND` w `man 3p open`). Jeśli program odczyta pustą linię, zwróć z `write_stage3`. Odczyt z `stdin` nie musi być realizowany funkcjami niskopoziomowymi.

Wykonywanie przejścia po katalogach

Funkcja `walk_stage4` wyświetla rekurencyjnie zawartość podanego katalogu, przy każdym pliku dodając informację, czy jest to plik, katalog czy inny (nieznany) typ pliku. Użyj funkcji `ftw` lub `ntfw`, nie implementuj zejść rekurencyjnych samodzielnie.

Etapy:

1. [4 p.] Funkcja `interface_stage1`
2. [4 p.] Funkcja `show_stage2`
3. [5 p.] Funkcja `write_stage3`
4. [3 p.] Funkcja `walk_stage4`