

Run L^AT_EX again to produce the table

L1: Plan B

Kolokwium z Matematyki Dyskretnej poszło w tym roku gorzej, niż się spodziewano. Studenci są zasmuceni i zaczynają szukać wszystkich dostępnych rozwiązań, aby zdać ten przedmiot. Na swoje (nie)szczęście, jednemu ze studentów udało się do wewnętrznego, wydającego systemu oceniania zadaa. W głosach studentów zrodziła się pewna myśl...

Wam się odkryło, że przyznawane punkty przechowywane są w pewnych plikach. Pliki te są umieszczane nie jednak w losowych katalogach, na różnych poziomach drzewa katalogów, bez ustalonej reguły. Student odkrył tylko, że w katalogu głównym systemu oceniania znajdują się pliki nazwane `grupaN` (gdzie N jest pewną liczbą całkowitą dodatnią), zawierające jedną liczbę całkowitą – liczbę studentów w grupie, a pliki z ocenami nazywają się `zMgN` (M, N – liczby całkowite dodatnie), zawierające tyle liczb całkowitych (oznaczających liczby punktów za zadanie), ile jest studentów w danej grupie. Liczba w pliku z grupami zapisana jest tekstowo, a z ocenami binarnie (4 bajty na jedną ocenę). Minimalny numer grupy to 1, a maksymalny to 20. Moesza wywołał, że dane są poprawne, w szczególności że dla każdego pliku `zMgN` będzie istniał plik `grupaN`.

Prowadzcy, gdy znajdą ostatni promyk słońca, wstawią punkty do USOS-a, więc nie ma za wiele czasu – potrzebna jest automatyzacja przetwarzania ocen.

System oceniania, jak przystało na porządne systemy, jest odpowiednio zabezpieczony – oprócz zwykłych plików istnieją także linki, które nazywają się według opisanych wcześniej reguł, ale otwarcie plików, na które wskazują, prowadzi do automatycznego niezaliczenia przedmiotu przez wszystkich studentów! Należy więc odpowiednio obsłużyć takie przypadki.

Uwagi

Początkowy kod (`sop-planb.c`) zawiera funkcje, które należy uzupełnić w tym i kolejnych etapach. Nie zmieniaj ich nazw. Moesz za to dodawać nowe funkcje oraz modyfikować argumenty istniejących. Zdefiniowana jest także zmienna globalna `groups_count`. Nie definiuj innych zmiennych globalnych oraz nie korzystaj jawnie ze zmiennej globalnej w funkcjach `get_groups_count`, `process_file` i `batch_process`.

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 informację, że wystąpił błąd).

W celu sprawdzenia Twojego rozwiązania, moesz skorzystać z przypadku testowego dostarczonego wraz z rozwiązaniem (katalog `data`). Przykładowe wyjście z użyciem tego przypadku znajduje się na końcu trzeciego zadania. Linki-powiązki prowadzą do plików z nieprawidłowymi danymi.

Etapy:

- 6 p. Uzupełnij funkcję `main`. Program powinien być uruchamiany z dokładnie jednym parametrem pozycyjnym, określającym ścieżkę do analizowanego pliku/folderu. Po uruchomieniu programu, program czeka na podanie na standardowym wejściu jednego z trzech słów:
 - `groups` – jeśli argument jest katalogiem, to wywołuje funkcję `get_groups_count`; w przeciwnym przypadku zgasza błąd i kończy działanie,
 - `process` – jeśli argument jest plikiem, to prosi o podanie liczby grup w pliku, a następnie wywołuje funkcję `process_file`; w przeciwnym przypadku zgasza błąd i kończy działanie,
 - `batch` – jeśli argument jest katalogiem, to wywołuje funkcję `batch_process`; w przeciwnym przypadku zgasza błąd i kończy działanie,

Podanie innego słowa powoduje wypisanie stosownego komunikatu i wyjście z programu.

- 6 p. Uzupełnij funkcję `get_groups_count`. Z podanego katalogu, wczytaj wszystkie pliki dotyczące grup. Dla pliku `grupaN`, gdzie N jest pewną liczbą, wypisz informację o numerze tej grupy i jej liczebności, np. jeśli zawartość pliku `grupa12` to 11, wypisz Grupa 12 zawiera 11 studentów. Następnie, do tablicy przekazanej jako argument funkcji, pod indeksem oznaczającym numer grupy, zapisz jej liczebność. Pamiętaj o niewczytywaniu linków – jeśli zostanie napotkany, wypisz stosowną informację.

3. 7 p. Uzupelnij funkcję `process`file`. Wczytaj oceny za zadanie z pliku funkcją `readv`. Po wczytaniu, w każdym buforze powinna znaleźć się jedna ocena. Wypisz kolejno te oceny na konsolę. Następnie zmodyfikuj te oceny, przyznając każdemu ze studentów dodatkowy punkt, a następnie zapisz wszystko do pliku funkcją `writev`. Nazwa nowego pliku powinna być taka, jak pierwotnego, z dopisanym przyrostkiem ``new`, np. jeśli otwieramy plik `z10g2`, plik wyjściowy powinien nazywać się `z10g2`new`. Format danych w pliku wyjściowym powinien być taki sam jak w wejściowym.
4. 5 p. Uzupelnij funkcję `batch`process`. Przejdź po całym drzewie katalogów i przetwórz wszystkie (i tylko te) pliki nazwane `zMgN`, gdzie M, N – liczby naturalne, w sposób opisany w powyższym etapie. Nie implementuj zeń rekurencyjnych samodzielnie (użyj funkcji `nftw`). Pamiętaj o niewczytywaniu linków – jeśli zostanie napotkany, wypisz stosowną informację. Możesz użyć funkcji uzupełnionych w dwóch poprzednich etapach. Dla każdego przetwarzanego pliku z ocenami, wypisz numer grupy i zadania oraz wczytane oceny.

Przykładowe wyjście

Polecenie `groups`

Grupa 13 ma 7 członków
grupa5: Link encountered
Grupa 1 ma 5 członków
Grupa 20 ma 6 członków
Grupa 2 ma 4 członków

Polecenie `process`

Student 0: 4pkt
Student 1: 3pkt
Student 2: 0pkt
Student 3: 1pkt
Student 4: 2pkt

Polecenie `batch` (fragmenty)

...
z4g2: Link encountered
...
Grupa 2, zadanie 1:
 Student 0: 7pkt
 Student 1: 12pkt
 Student 2: 4pkt
 Student 3: 0pkt
Grupa 20, zadanie 256:
 Student 0: 1pkt
 Student 1: 2pkt
 Student 2: 4pkt
 Student 3: 8pkt
 Student 4: 16pkt
 Student 5: 32pkt
...
d2/z55g2: Link encountered