

Wstęp do programowania, potok imperatywny (Info, I rok) 18/19, laboratorium

Kokpit ► Moje kursy ► WPI.LAB.INFO.I.18/19 ► Laboratorium 8 ► Przekształcenie

Przekształcenie

Wprowadzenie

Prostokątna plansza do gry SameGame dzieli się na pola, uporządkowane w wiersze i kolumny.

Każde pole albo jest puste albo jest na nim klocek określonego rodzaju.

Gracz usuwa z planszy grupy sąsiadujących klocków tego samego rodzaju.

Klocki opadają na puste pola w wierszach poniżej.

Kolumny, w których są jakieś klocki, są przesuwane w lewo, na miejsce kolumn pustych.

Polecenie

Napisz program przekształcający tekst w sposób wzorowany na grze SameGame.

Dane programu określają rozmiar planszy i jej początkowy stan.

Liczba wierszy planszy jest równa liczbie wierszy danych a liczba kolumn planszy jest równa maksymalnej długości wiersza danych.

Znaki różne od spacji, dalej nazywane znakami widocznymi, odpowiadają klockom w SameGame a spacje i miejsca za końcem wiersza to pola puste.

Zbiór widocznych znaków G w tekście nazywamy grupą, jeśli spełnione są trzy warunki:

1. wszystkie znaki w zbiorze G są takie same,
2. od każdego znaku w zbiorze G można przejść do każdego z pozostałych znaków w tym zbiorze, wykonując tylko kroki do bezpośrednich sąsiadów w wierszu lub w kolumnie i poruszając się przy tym wyłącznie po znakach ze zbioru G ,
3. zbioru G nie można rozszerzyć, nie łamiąc zasad 1 i 2.

Inaczej, niż w grze SameGame, nie wymagamy, by grupa była co najmniej dwuelementowa.

Można zauważyć, że każdy widoczny znak w tekście jednoznacznie określa grupę, do której należy.

Program wykonuje następujący algorytm:

1. czyta i porządkuje dane,
2. wymazuje jedną grupę znaków widocznych, o ile taka grupa istnieje,
3. porządkuje i pisze wynik.

Porządkując tekst, program:

- dopóki jest choć jeden widoczny znak, bezpośrednio poniżej którego, w tej samej kolumnie, jest pole puste, przesuwa ten znak o jeden wiersz w dół,
- usuwa wszystkie kolumny, w których nie ma widocznego znaku.

Wymazanie grupy widocznych znaków polega na zastąpieniu ich w tekście spacjami.

Do wymazania wybierana jest grupa widocznych znaków, do której należy znak w pierwszej kolumnie ostatniego wiersza tekstu.

Postać danych

Dane programu to dowolny tekst.

Wolno założyć, że każdy wiersz danych, także ostatni, będzie zakończony reprezentacją końca wiersza `\n`.

Wolno założyć, że dane zmieszczą się w pamięci a liczba wierszy i liczba kolumn będzie w zakresie typu `int`.

Jeśli rozwiązanie nałoży dodatkowe ograniczenia na rozmiar danych, to jego ocena jakości zostanie obniżona o 1 punkt.

Podczas walidacji i testów dane będą miały nie więcej niż 20 wierszy i nie więcej niż 80 kolumn.

Za program, który ogranicza liczbę wierszy do 20 i liczbę kolumn do 80, można więc zdobyć maksymalnie 19 punktów na 20 możliwych.

Postać wyniku

Wynikiem programu są niepuste wiersze przekształconych danych.

Na końcach wierszy nie są wypisywane spacje.

Każdy wiersz, także ostatni, jest zakończony końcem wiersza `\n`.

Przykłady

Do treści zadania dołączone są pliki `.in` z przykładowymi danymi i pliki `.out` z wynikami wzorcowymi.

- Dla danych przyklad1.in poprawny wynik to przyklad1.out.
- Dla danych przyklad2.in poprawny wynik to przyklad2.out.
- Dla danych przyklad3.in poprawny wynik to przyklad3.out.

Walidacja i testy

- Rozwiązania zostaną poddane walidacji, wstępnie sprawdzającej zgodność ze specyfikacją. Pomyślne przejście walidacji jest warunkiem dopuszczenia programu do testów poprawności.

- Walidacja i testy zostaną przeprowadzone na komputerze `students`.
- Programy będą kompilowane poleceniem:

```
gcc -std=c11 -pedantic -Wall -Wextra -Werror -fstack-protector-strong -g nazwa.c  
-o nazwa
```

gdzie `nazwa.c` to nazwa pliku z kodem źródłowym.

Wymagane są wszystkie wymienione opcje kompilatora. Nie będą do nich dodawane żadne inne.

Zwracamy uwagę, że poszczególne wersje kompilatora `gcc` mogą się różnić sposobem obsługi tych samych opcji. Przed wysłaniem rozwiązania warto więc skompilować je i przetestować na `students`, w sposób opisany powyżej.

- Podczas walidacji i testów, program `nazwa` z rozwiązaniem będzie uruchamiany pod kontrolą programu Valgrind poleceniem:

```
valgrind -q ./nazwa
```


Jeśli Valgrind wykryje błąd, to nawet, gdyby wynik był prawidłowy uznamy, że program testu nie przeszedł.

- Przyjmujemy, że wynik funkcji `main()` inny niż `0` informuje o błędzie wykonania programu.
- Poprawność wyniku sprawdzamy, przekierowując na wejście programu zawartość pliku `.in` i porównując rezultat, za pomocą programu `diff`, z plikiem `.out`, np.:

```
< przyklad.in ./nazwa | diff - przyklad.out
```

Ocena poprawności wyniku jest binarna. Uznajemy go za poprawny, jeżeli program `diff` nie wypisze żadnej różnicy między wynikiem programu a wynikiem wzorcowym.

Status przesłanego zadania

Status przesłanego zadania	Przesłane do oceny
Stan oceniania	Nie ocenione
Termin oddania	Tuesday, 4 December 2018, 18:00 PM
Pozostały czas	Zadanie zostało złożone 47 min. 31 sek. przed terminem
Ostatnio modyfikowane	Tuesday, 4 December 2018, 17:12 PM
Przesyłane pliki	 same.c
Komentarz do przesłanego zadania	► Komentarze (0)