Karol Cidyło

Zadanie 6 z listy4.

Mamy dwa podciągi x oraz y. Zadaniem jest znalezienie najdłuższego wspólnego podciągu nie zawierającego słowa 'egzamin'. Rozwiązanie wykorzystuje programowanie dynamiczne. Wykorzystuje również tablice dwuwymiarowe gdzie T[i][j] to długość najdłuższego wspólnego podciągu złożonego z prefiksów x[1...i], y[1...j]. Aby otrzymać rozwiązanie dla podciągu nie zawierającego słowa 'egzamin' dla danych x i y oraz i,j oznaczających indeksy znaków w danym ciągu będziemy obliczać tablice dla rozwiązań pośrednich. To znaczy:
T_1 jest tablicą gdzie trzymane są wyniki najdłuższego wspólnego podciągu nie zawierającego podsłowa 'egzamin' i nie kończy się ten podciąg na 'e'
T_2 jest tablicą gdzie trzymane są wyniki najdłuższego wspólnego podciągu nie zawierającego podsłowa 'egzamin' i nie kończy się ten podciąg na 'eg'
I tak dalej aż do: T_7 czyli wyniki najdłuższego wspólnego podciągu nie zawierającego podsłowa 'egzamin'
Przyjmjmy kiedy mówimy o indeksach i-tych mamy na myśli i-ty znak ciągu x, a kiedy o j-tych to j-ty znak ciągu y.

Funkcja obliczT1(i, j) będzie obliczać pole T_1[i][j].

```
obliczT1(i,j):
    jesli i == 0 || j == 0:
        T1[i][j] = 0;

    jesli x[i] != y[j]:                    (1)
        T1[i][j] = max(T1[i-1][j], T1[i][j-1])

    jesli x[i] == y[j] && x[i] == 'e':              (2)
        T1[i][j] = T1[i-1][j-1]

    jesli x[i] == y[j] && x[i] == 'n':              (3)
        T1[i][j] = max(T6[i-1][j-1] + 1, T1[i-1][j-1])

    jesli x[i] == y[j] && x[i] != 'e' &&  x[i] != 'n':  (4)
        T1[i][j] = T7[i-1][j-1] + 1
```

(1) - znaki na miejsach i oraz j są różne, więc naturalnie bierzemy pod uwagę maksymalne wyniki dla innych indeksów obu ciągów.
(2) - T_1 zawiera takie wyniki, że najdłuższy wspólny podciąg nie kończy się na 'e', więc nie możemy dodać wystąpienia litery w obu ciągach do wyniku.

(3) - znaki na miejsach i oraz j są równe 'n' czyli możemy wybrać maksymalny wynik z **I.** najdłuższych ciągów takich, które nie kończą się na 'egzami', ponieważ dodając n, mielibyśmy całe słowo 'egzamin'(co nie może być brane pod uwagę przy obliczaniu wyniku. Do tego dodajemy 1, ponieważ możemy dodać n i nie będziemy mieć słowa 'egzamin' oraz litery 'e' na końcu) oraz **II.** ciągiem nie kończącym się na 'e' ale dla obu indeksów mniejszych o 1.

(4) - jeśli wiemy, że litera która jest wspólna dla obu ciągów nie jest 'e' oraz 'n' bierzemy wynik z T7[i-1][j-1] i dodajemy do niego 1, czyli zwiększamy wynik, ponieważ nie psuje nam to założeń zadania oraz T1.

Natomiast funkcja oblicz T2(i,j) wygląda bardzo podobnie:

```
obliczT2(i,j):
    jesli i == 0 || j == 0:
        T2[i][j] = 0;

    jesli x[i] != y[j]:
        T2[i][j] = max(T2[i-1][j], T2[i][j-1])

    jesli x[i] == y[j] && x[i] == 'g':
        T2[i][j] = max(T1[i-1][j-1] + 1, T2[i-1][j-1])

    jesli x[i] == y[j] && x[i] == 'n':
        T2[i][j] = max(T1[i-1][j-1] + 1, T2[i-1][j-1])

    jesli x[i] == y[j] && x[i] != 'g' &&  x[i] != 'n':
        T2[i][j] = T7[i-1][j-1] + 1
```

Funkcje obliczT3(i,j) do obliczT6(i, j) są analogiczne tylko zamieniamy literę do sprawdzenia na koleją z ciągu 'egzamin'.

```
obliczT7(i,j):
    jesli i == 0 || j == 0:
        T7[i][j] = 0;

    jesli x[i] != y[j]:                    (1)
        T7[i][j] = max(T7[i-1][j], T7[i][j-1])

    jesli x[i] == y[j] && x[i] == 'n':          (2)
        T7[i][j] = max(T6[i-1][j-1] + 1, T7[i-1][j-1])
    jesli x[i] == y[j]  && x[i] != 'n':         (3)
        T7[i][j] = T7[i-1][j-1] + 1
```

(1) - znaki na miejsach i oraz j są różne, więc naturalnie bierzemy pod uwagę maksymalne wyniki dla innych indeksów obu ciągów.
(2) - znaki na miejsach i oraz j są równe 'n' czyli możemy wybrać maksymalny wynik z **I.** T6 czyli najdłuższego ciągu nie kończacego się na 'egzami' (tak, żeby po dodaniu 'n' otrzymać wciąż prawidłowe rozwiązanie) oraz **II.** T7 nie biorącego pod uwagę znaków na i oraz j. **(3)** - znaki na miejsach i oraz j są równe i różne 'n', więc wybieramy maksymalne rozwiązanie dla ciągu bez znaków na indeksach i oraz j następnie inkrementujemy wynik końcowy.

Każdą z tych funckji wykonujemy pokolei według indeksów i oraz j. To znaczy:

```
lst-egzamin(n,m):
    dla kazdego i = 0 do n - 1:
        dla kazdego j = 0 do m -1:
        obliczT1(i,j):
        obliczT2(i,j):
        obliczT3(i,j):
        obliczT4(i,j):
        obliczT5(i,j):
        obliczT6(i,j):
        obliczT7(i,j):
```

Wynik,a dokładnie długość najdłuższego wspólnego podciągu nie zawierającego słowa 'egzamin' jest w T_7[n-1][m-1] gdzie odowiednio
n - liczba znaków słowa x
m - liczba znaków słowa y.
Do otrzymania konkretnego ciągu potrzebujemy przejść od tyłu (to jest od $T_7[n-1][m-1]$) i wybierać największe wartośći według kryteriów tak jak w funkcjach T1-T7. Czyli tak naprawdę jak w większości przypadków odtwarzania rozwiązania algorytmów dynamicznych, tylko tutaj mamy więcej niż jedną tablicę do przejrzenia wyników.

Złożoność czasowa.
Każda funkcja wykonuje się w czasie $O(n * m)$ dla długości ciągów x i y odpowiednio n i m. Sprawdzanie warunków w fukcji oraz przypisanie wartości mamy w czasie stałym. Mamy 7 funkcji, co też jest stałą wartością. Czyli ogólniej $O(n^2)$.
Złożoność pamięciowa.
Siedem tablic $O(n * m)$ czyli również $O(n^2)$.