

## **XML i nowoczesne technologie zarządzania treścią 2009/10 – Zadanie 3**

Należy napisać program w Javie (lub po ustaleniu ze sprawdzającym w innym języku programowania) dokonujący zamiany kolejności kolumn w arkuszu Open Document.

### **Parametry wywołania**

Program będzie wywoływany z następującymi parametrami:

- ścieżka do pliku wejściowego,
- ścieżka do pliku wynikowego,
- nazwa arkusza, który ma być zmodyfikowany,
- N parametrów będących identyfikatorami kolumn – opisują one permutację N początkowych kolumn; każdy identyfikator jest (1-3)-literowym identyfikatorem kolumny jak to w arkuszach kalkulacyjnych (np. „A”, „CZ”, ostatnia kolumna w Open Office to „AMJ” i można uznać to za ograniczenie).

### **Oczekiwane działanie**

Plik wejściowy i wyjściowy to pliki w formacie Open Document Spreadsheet (.ods), do tworzenia i weryfikacji danych testowych będzie używany Open Office 3.1.

Program powinien czytać plik wejściowy nie modyfikując go, a wynik zapisywać do pliku wyjściowego. Przetwarzanie powinno odbywać się w sposób „strumieniowy”, bez wczytywania całych plików do pamięci i w miarę możliwości bez tworzenia plików pomocniczych. Pliki .ods są archiwami ZIP i należy „strumieniowo” dekompresować źródłowy i tworzyć wynikowy, a wewnętrznym plikiem, którego zawartość może ulec zmianie, jest `content.xml`.

Przetwarzanie dokumentów XML ma odbywać się za pomocą standardu SAX lub StAX, tj. w sposób nie wymagający wczytywania całego dokumentu do pamięci. W przypadku SAX trzeba, a w przypadku StAX można skorzystać z `Transformer` lub `TransformerHandler` do zapisywania wyniku przekształcenia. Należy obsługiwać przestrzenie nazw.

W pliku wynikowym („skoroszycie” zgodnie z terminologią Excela) w co najwyżej jednym arkuszu może zostać zmieniona kolejność kolumn. Permutacja podana w parametrach programu opisuje nową kolejność kolumn, używając starych identyfikatorów kolumn. Jeśli skoroszyt nie zawiera arkusza o podanej nazwie, nie jest modyfikowany.

Należy zaktualizować w formułach referencje do komórek, których adresy ulegają zmianie. Także w formułach na innych arkuszach.

Nie będą oceniane natomiast referencje m.in. w wykresach, filtrach itd.

Można założyć, że w arkuszu wejściowym nie będzie scalonych komórek. Nie przejmujemy się zbyt zachowaniem formatu (np. obramowań).

Nie oczekuję dogłębnej analizy specyfikacji Open Document i obsługi wszelkich przypadków, a jedynie analizy przykładów i obsługi arkuszy, w których występują:

- dane tekstowe, liczbowe, data i czas,
- formuły z referencjami do komórek z tego samego i z innego arkusza w tym samym skoroszycie; z adresami bezwzględными („z dolarami”), względnymi i mieszany.

## Informacje organizacyjne

W razie wątpliwości można pytać mailowo autora zadania: [czarnik@mimuw.edu.pl](mailto:czarnik@mimuw.edu.pl)

Rozwiązania (w postaci archiwum ZIP o nazwie równej loginowi studenckiemu) należy wysłać do **31 stycznia 2010** włącznie na adres: [czarnik@mimuw.edu.pl](mailto:czarnik@mimuw.edu.pl) z konta mailowego na students.

## Uzupełnienia i odpowiedzi

1. Do pamięci można wczytać dane całego wiersza zmienianej tabeli.
2. Zarówno względne, jak i bezwzględne adresy po zamianie mają odnosić się do tych samych logicznie komórek, do których odnosiły się wcześniej (oba rodzaje adresów traktujemy tak samo).
3. Problem z parserem SAX (i pośrednio Transformerem) wykonującym niepotrzebnie `close()` na otwartym `ZipInputStream` można rozwiązać tworząc własną podklasę `ZipInputStream`, która wykonuje prawdziwego `close()` tylko w odpowiednim stanie (można na przykład zliczać `open()` i `close()`).
4. Należy obsługiwać atrybuty `number-columns-repeated` a także napisy w formułach, m.in. napisy zawierające ciągi znaków podobne do adresów. [przyklad.ods](#) zawiera niektóre konstrukcje.
5. Można założyć, że liczba kolumn zadeklarowana dla arkusza zgadza się z liczbą komórek w wierszach. I w ogóle, że plik wejściowy został utworzony w OpenOffice 3.1, a nie jest spreparowany.
6. Można założyć, że permutacja podana w parametrach dotyczy N początkowych kolumn, gdzie  $N \leq$  liczby kolumn w arkuszu.
7. Program ma prawo dać niepoprawny rezultat jeśli w arkuszu wejściowym w formułach znajdują się zakresy dotyczące wielu kolumn. W takim przypadku mogłaby istnieć potrzeba rozbicia zakresu na wiele części, co z kolei nie w każdym kontekście jest dopuszczalne – nie musimy tego analizować i można zastosować jakieś naiwne rozwiązanie. Mają działać zakresy „pionowe” dotyczące jednej kolumny.