Praca domowa 11 –teryt

Termin zwrotu : 21 stycznia godz. 23.00
Zadanie uznaje się za zaliczone, gdy praca oceniona zostanie na co najmniej 6 pkt.

Na stronach Głównego Urzędu Statystycznego pod podanym niżej adresem

http://eteryt.stat.gov.pl/eTeryt/rejestr_teryt/udostepnianie_danych/baza_teryt/uzytkownicy_indywidualni/pobieranie/pliki_pelne.aspx

dostępne są publicznie informacje dotyczące między innymi aktualnego podziału terytorialnego kraju (baza TERC podstawowa) oraz urzędowy wykaz ulic (baza ULIC podstawowa). Wykazy mogą być pobierane w jednej z dostępnych wersji : CSV lub XML. Dla potrzeb realizacji niniejszego zadania korzystać należy wyłącznie z danych w formacie XML (rozwiązanie w oparciu o CSV nie będzie podlegało ocenie).

Należy stworzyć usługę sieciową pracującą w standardzie WebService's (JAX-RS). Usługa winna zwracać informację o ilości wystąpień określonej nazwy ulicy na poszczególnych stopniach hierarchii podziału administracyjnego kraju, a więc np. w województwach, na które podzielono terytorium Polski, powiatach w ramach wskazanego województwa, gminach w określonym województwie i powiecie.

Program winien być zapisany w pliku `GUService.java` oraz ewentualnie dalszych – wykonanych dla potrzeb zadania. Poszczególne elementy rozwiązania nie mogą korzystać z bibliotek zewnętrznych innych niż niezbędne moduły serwera (jak np. javaee.jar itp.).

Proces kompilacji musi być możliwy z użyciem komendy

```
javac –cp <app-server-modules> –Xlint GUService.java *.java
```

Zawartość pliku web.xml, który używany będzie w trakcie uruchamiania i testowania usługi podano niżej :

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns="http://java.sun.com/xml/ns/javaee" xmlns:web="http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
       xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
       id="WebApp_ID" version="3.0">
  <servlet>
    <servlet-name>javax.ws.rs.core.Application</servlet-name>
  </servlet>
  <servlet-mapping>
    <servlet-name>javax.ws.rs.core.Application</servlet-name>
    <url-pattern>/*</url-pattern>
  </servlet-mapping>
</web-app>
```

Proces wykonywania musi być możliwy z użyciem niżej wskazanych poleceń zwracających ilość wystąpień ulicy o wskazanej w treści żądania nazwie :

```
http://<host:port>/<app-name>/pl/<ulica>          - wybiera obszar kraju
http://<host:port>/<app-name>/pl/XX/<ulica>       - wybiera wskazane województwo
http://<host:port>/<app-name>/pl/XX/YY/<ulica>    - wybiera wskazane województwo i powiat
http://<host:port>/<app-name>/pl/XXYY/<ulica>     - jak wyżej
http://<host:port>/<app-name>/pl/NNN/<ulica>      - wybiera wskazane województwo
http://<host:port>/<app-name>/pl/NNN/YY/<ulica>   - wybiera wskazane województwo i powiat
http://<host:port>/<app-name>/pl/NNNYY/<ulica>    - jak wyżej
```

gdzie *XX* wskazuje numeryczny kod poszukiwanego województwa a *YY* numeryczny kod powiatu w obrębie województwa.

W miejsce numerycznego kodu województwa *XX* usługa winna akceptować alternatywnie trzyliterowy *NNN* kod województwa utworzony z trzech pierwszych liter jego nazwy (case insensitive), wyłącznie z użyciem liter alfabetu angielskiego (czyli np. dla 'Małopolski' oznaczenie 'mal') . `<ulica>` wskazuje nazwę szukanej ulicy np. 'Tadeusza Kościuszki'. Ze względu na konieczność użycia nazw wieloczłonowych oraz polskich liter ta część URL'a kodowana będzie u użyciem standardu URL encode, czyli żądanie postaci

```
http://<host:port>/<app-name>/pl/mal/Tadeusza%20Ko%C5%9Bciuszki
```

winno w odpowiedzi zwrócić ilość wystąpień tekstu 'Tadeusza Kościuszki' we wszystkich jednostkach administracyjnych wchodzących w skład województwa małopolskiego. Przeszukiwaniu podlega wyłącznie pole <NAZWA1> w urzędowym wykazie nazw ulic (baza ULIC). Zapisy 'Tadeusza Kościuszki', 'Rondo Tadeusza Kościuszki', 'Plac Tadeusza Kościuszki' czy wreszcie 'Rondo Tadeusza Kościuszki i Powstańców …' traktowane są jako spełniające warunki zadania, a więc wystarcza zgodność fragmentu nazwy z zadanym wzorcem. Porównywanie nazw winno uwzględniać wielkość liter (case sensitive).

W odpowiedzi usługa zawsze zwraca liczbę wskazującą ilość wystąpień badanej nazwy w przeszukiwanym obszarze, np. :

```
Ilość wystąpień : 7
```

Wymagania :

- Klasa implementująca aplikację winna zostać zdefiniowane w pliku `GUService.java`.
- W pliku README.pdf winien być zawarty opis mechanizm operowania danymi oraz algorytm wyznaczania wyniku.
- Proces obliczenia rozwiązania winien się kończyć w czasie nie przekraczającym 1 min (orientacyjnie dla typowego notebooka). Po przekroczeniu limitu czasu zadanie będzie przerywane, i traktowane podobnie jak w sytuacji błędów wykonania (czyli nie podlega dalszej ocenie).

Sposób oceny **:**

- 1 pkt – **Weryfikacja** : czy program jest skompletowany i spakowany zgodnie z ogólnymi zasadami przesyłania zadań.
- 1 pkt – **Kompilacja** : każdy z plików winien być kompilowany bez jakichkolwiek błędów lub ostrzeżeń (w sposób omówiony wyżej)
- 1 pkt – **Wykonanie** : program powinien wykonywać się bez jakichkolwiek błędów i ostrzeżeń (dla pliku danych wejściowych zgodnych z wyżej zamieszczoną specyfikacją) z wykorzystaniem omówionych wyżej parametrów linii komend
- 2 pkt – **README** : plik README.pdf dokumentuje w sposób kompletny i właściwy sposób zestawiania połaczenia
- 1 pkt – **Styl kodowania** : czy funkcji i zmienne posiadają samo-wyjaśniające nazwy ? Czy podział na funkcje ułatwia czytelność i zrozumiałość kodu ? Czy funkcje eliminują (redukują) powtarzające się bloki kodu ? Czy wcięcia, odstępy, wykorzystanie nawiasów itp. (formatowanie kodu) są spójne i sensowne ?
- 4 pkt – **Poprawność algorytmu** : czy algorytm został zaimplementowany poprawnie a wynik odpowiada prawidłowej (określonej zbiorem danych testowej) wartości.