Praca domowa 10 – control

Termin zwrotu : 12 stycznia godz. 23.00
Zadanie uznaje się za zaliczone, gdy praca oceniona zostanie na co najmniej 6 pkt.

Na serwerze aplikacyjnym Glassfish 4 w kontenerze *ejb* zainstalowany jest pod nazwą *mdb-project* (deployment descriptor) komponent (session bean) o nazwie *MdbManager* wraz z interfejsem *IMdbManager*, który zdefiniowany jest następująco :

```
package pl.jrj.mdb;
import javax.ejb.Remote;

@Remote
public interface IMdbManager {
        public String sessionId(String album);   // album – numer albumu studenta
}
```

Metoda *sessionId* dokonuje rejestracji użytkownika w systemie zwracając znakową postać numerycznego identyfikatora sesji, jeżeli proces rejestracji zakończył się poprawnie. Jeżeli rejestracja zakończyła się niepowodzeniem, metoda *sessionId* zwraca wartość **null.**

Należy zaprojektować i wykonać usługi sieciowe realizowane przez komponent o nazwie *RESTcontrol*, które wykonywane będą z wykorzystaniem odwołań :

```
/control/start                  /control/stop
/control/res                    /control/err
/control/clr
/control/icr                    /control/icr/n
/control/dcr                    /control/dcr/n
```

Komponent może znajdować się w jednym z dwóch stanów : *zliczania* lub *wstrzymania*. Początkowo komponent znajduje się w stanie wstrzymania a licznik komponentu jest wyzerowany. Przejście ze stanu wstrzymania do stanu zliczania jest możliwe wyłącznie po zakończonej sukcesem rejestracji i następuje w efekcie wywołania polecenia *start*. Przejście ze stanu zliczania do stanu wstrzymania następuje w efekcie wywołania polecenia *stop*. Wykonanie polecenia *icr* powoduje zmianę wartości licznika o 1, polecenia *icr/n* powoduje zmianę wartości licznika o wartość *n,* gdzie *n* jest liczbą naturalną, zapisaną w jednym z formatów stałych w Javie : dwójkowym, ósemkowym, szesnastkowym lub dziesiętnym. Wykonanie polecenia zmieniającego stan licznika dopuszczalne jest wyłącznie w stanie zliczania. Wykonanie metody *icr/n* lub *dcr/n* w stanie wstrzymania nie zmienia wartości licznika, powoduje natomiast zarejestrowanie faktu niepoprawnego żądania poprzez zwiększenie licznika błędów (error). Podobnie wywołanie metody *start* w stanie zliczania lub metody *stop* w stanie wstrzymania jest operacją

niepoprawną, skutkującą wyłącznie zwiększeniem licznika błędów (error). Metoda *clr* zeruje liczniki. Metody *res* oraz *err* zwracają wartości będące wynikiem obliczenia reszty z dzielenia *sessionId* modulo *N*, gdzie *N* jest wartością licznika (metoda *res*) lub licznika błędów (metoda *err*).

A więc w przykładowym ciągu żądań :	Odpowiedź usługi:	Uwagi :
`http://......./control/icr/1`	`pusta strona`	`żądanie błędne`
`http://......./control/start`	`pusta strona`	
`http://......./control/icr/2`	`pusta strona`	`licznik += 2`
`http://......./control/icr`	`pusta strona`	`licznik += 1`
`http://......./control/res`	`3`	
`http://......./control/start`	`pusta strona`	`żądanie błędne`
`http://......./control/icr`	`pusta strona`	`licznik += 1`
`http://......./control/stop`	`pusta strona`	
`http://......./control/icr/1`	`pusta strona`	`żądanie błędne`
`http://......./control/stop`	`pusta strona`	`żądanie błędne`
`http://......./control/err`	`sessionId % 4`	
`http://......./control/res`	`sessionId % 4`	

Program ma być zapisany w dwóch plikach : `IMdbManager.java` zawierającym definicję interfejsu komponentu `MdbManager`, oraz kod komponentu `RESTcontrol.java`. Poszczególne elementy rozwiązania nie mogą korzystać z bibliotek zewnętrznych innych niż niezbędne moduły serwera (jak np. gf-client.jar, javaee.jar itp.).

Proces kompilacji musi być możliwy z użyciem komendy

```
javac –cp <app-server-modules> –Xlint RESTcontrol.java IDbManager.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>
```

Uwaga :

Implementacja usługi winna zapewniać w pełni prawidłowe działanie w sytuacji obsługi strumienia żądań pochodzących od wielu współbieżnie korzystających z usługi klientów.

Wymagania :

- Klasa implementująca komponent winna zostać zdefiniowane w pliku `RESTcontrol.java`.
- Interfejs umożliwiający poprawną rejestrację zadania winien zostać zdefiniowany w pliku `IMdbManager.java`.
- W pliku README.pdf winien być zawarty opis architektury proponowanego rozwiązania.
- 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 połączenia
- 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.