Praca domowa 04 – block

Termin zwrotu : 18 listopada godz. 23.00
Zadanie uznaje się za zaliczone, gdy praca oceniona zostanie na co najmniej 6 pkt.

Wartości pewnej funkcji dwóch zmiennych z = f(x, y), gdzie x, y, z $\in \Re$, przechowywane są w SQL-owym repozytorium danych (w bazie danych) w pewnej tabeli o nieznanej początkowo nazwie <*name*>. Struktura tabeli utworzona została z wykorzystaniem instrukcji

```
CREATE TABLE <name> (
   id int NOT NULL,
   x float NOT NULL,
   y float NOT NULL,
   z float NOT NULL
)
CONSTRAINT [PK_Table] PRIMARY KEY
( id )
```

Połączenie do SQL-owej bazy danych (dostęp do bazy) realizowane jest z wykorzystaniem zdefiniowanego dla potrzeb serwera Glassfish 4 źródła danych (javax.sql.Datasource) dostępnego z wykorzystaniem usług JNDI. Na serwerze aplikacyjnym Glassfish 4 w kontenerze *ejb* zainstalowany jest pod nazwą *ejb-project* (deployment descriptor) komponent (stateful session bean) o nazwie *DSManager* wraz z interfejsem *IDSManagerRemote*, który zdefiniowany jest następująco :

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

@Remote
public interface IDSManagerRemote {
      public String getDS();
}
```

Metoda *getDS()* zwraca informację o nazwie źródła danych pod którą źródło zarejestrowane zostało w usłudze JNDI.

Należy obliczyć (wyznaczyć) objętość graniastosłupa prostego o podstawie P oraz wysokości h. Podstawę P graniastosłupa stanowi wypukła otoczka rzutu punktów (x,y) na płaszczyznę XY. Wysokość h graniastosłupa wyznaczona jest jako średnia wartość funkcji z = f(x,y) określonej nad zadanym zbiorem punktów (x,y).

Algorytm obliczania objętości bryły należy zaimplementować w postaci komponentu EJB o nazwie *Block* wraz z niezbędnym interfejsem o nazwie *IBlockRemote* udostępniającym metodę realizującą proces obliczeń i zwracającą wyznaczoną przez komponent wartość objętości bryły z dokładnością do 5 miejsc dziesiętnych.

Sterowanie procesu obliczeń winno być zaimplementowane w postaci servletu *Solver*. Servlet otrzymuje przekazywaną w żądaniu (url) jako parametr *t* informację o nazwie <*name*> tabeli przechowującej dane. Korzystając z wykonanej i udostępnionej (interfejs *IBlockRemote*) metody komponentu *Block (*realizującej algorytm obliczania objętości bryły) servlet ustala i wyprowadza prawidłowo obliczoną poszukiwaną wartość objętości.

Program ma być zapisany w czterech plikach : `IDSManagerRemote.java` zawierającym definicję interfejsu komponentu `DSManager`, `IBlockRemote.java` zawierającym definicję interfejsu komponentu zdefiniowanego w pliku `Block.java` , oraz kod servletu `Solver.java`. Poszczególne elementy rozwiązania nie mogą korzystać z bibliotek zewnętrznych innych niż niezbędne moduły serwera (jak np. gf-client.jar, javaee.jar itp.). Działanie rozwiązania nie może być zależne od jakiegokolwiek dialektu SQL.

Proces kompilacji musi być możliwy z użyciem komendy

```
javac –cp <app-server-modules> –Xlint Solver.java IBlockRemote.java Block.java IDSManagerRemote.java
```

Rozwiązanie testowane będzie w środowisku serwera aplikacyjnego GlassFish 4. Zawartość pliku web.xml, który używany będzie trakcie uruchamiania i testowania komponentu 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>servletNNNNN</servlet-name>
    <servlet-class>Solver</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>servletNNNNN</servlet-name>
    <url-pattern>/*</url-pattern>
  </servlet-mapping>
</web-app>
```

gdzie NNNNN oznacza numer albumu studenta, którym sygnowana jest praca.

Wymagania :

- Klasa implementująca komponent realizujący algorytm winna zostać zdefiniowane w pliku `Block.java`.
- Interfejs udostępniający metodę zwracającą poszukiwaną przez komponent `Block.java` wartość pola winien zostać zdefiniowany w pliku `IBlockRemote.java`.

- Servlet nadzorujący proces obliczeń zapisać należy w pliku `Solver.java`
- W pliku README.pdf winien być zawarty opis mechanizmu wyszukiwania (lookup) i zestawiania połączenia.
- 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 algorytm poszukiwania rozwiązania.
- 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.