Praca domowa 5 – ejb

Szlachetka Miłosz
Nr albumu. 114014

1.Cel zadania

Celem zadania była implementacja programu, który wyznacza promień bezwładności układu punktów względem prostej (prosta dana równaniami: **3ax + 2by = 0** i **3by + 2cz = 0**) w przestrzeni 3D. Dane wejściowe, niezbędne do utworzenia prostej oraz punktów, pobierane są z komponentu ejb.

2.Struktura programu

Program zawarty jest w dwóch plikach **EbClient.java** ora **IDataMonitor.java**.

IDataMonitor.java zawiera:
@Remote
public interface IDataMonitor - interfejs, który jest implementowany przez komponent ejb (statefull session bean). Interfejs zawiera dwie deklaracje metod:
public boolean hasNext() - zwraca false, jeśli wartość dla metody next() jest nieokreślona
public double next() - zwraca kolejną wartość liczbową

EbClient.java zawiera:
-public class EbClient - klasa główna, która implementuje rozwiązanie zadania
Zmienne składowe klasy **EbCLient**:
-private static IDataMonitor dataMonitor - przechowuje obiekt EJB, z którego są pobierane dane do programu
-private static double a - przechowuje współczynnik a prostej
-private static double b - przechowuje współczynnik b prostej
-private static double c - przechowuje współczynnik c prostej

Klasy wewnętrzne:
private static class Point3D - klasa, która reprezentuje punkt w przestrzeni 3D. Zawiera współrzędne x,y,z oraz masę punktu. Ponadto klasa zawiera gettery,settery, konstruktor bezparametrowy oraz konstruktor z parametrami x,y,z,masa.

private static class Vector3D - klasa, która reprezentuje wektor w przestrzeni 3D. Zawiera współrzędne x,y,z oraz gettery,settery,konstruktor bezparametrowy i konstruktor z parametrami x,y,z. Oprócz tego zawiera metody:
-public double calculateLength() - metoda obliczająca i zwracająca długość wektora, dla którego metoda została wywołana.
public static Vector3D getDirectionVector() - metoda zwracająca obekt wektora, który jest wektorem kierunkowym dla prostej, która została opisana w poleceniu pracy domowej.
-public static Vector3D getVectorP1P0(Point3D P1) - metoda zwracająca wektor P1P0 otrzymany z punktów P1 i P0, gdzie P1 jest punktem podanym jako parametr metody, natomiast P0=(0,0,0) jest to punkt, który leży na prostej, która została opisana w poleceniu pracy domowej.
-public static Vector3D calculateVectorProduct(Vector3D v1, Vector3D v2) - metoda wykonująca mnożenie wektorowe dwóch wektorów w 3D. Zwraca nowy wektor, który został otrzymany w wyniku mnożenia wektorowego wektorów v1 oraz v2 (**v1**x**v2)**.

Metody klasy **EbClient:**
-static void loadDataMonitor() throws NamingException - metoda ta tworzy kontekst aplikacji (InitialContext), a następnie korzystając z interfejsu JNDI oraz konketstu wyszukuje i pobiera komponent EJB do zmiennej składowej klasy EbClient.
-private static void loadABCCoeffiecients() - z komponentu EJB pobiera współczynniki a,b,c prostej, która została opisana w poleceniu pracy domowej.
-private static Point3D retrieveOnePoint() - z komponentu EJB pobiera współrzędne x,y,z oraz masę punktu i tworzy obiekt reprezentujący punkt w 3D, który zawiera uprzednio pobrane wartości.
-private static List<Point3D> retrieveAllPoints() - Tworzy listę punktów zawierających współrzędne x,y,z oraz masę. Metoda wywołuje w.w. metodę retrieveOnePoint() dopóki dopóki zwraca ona obiekty reprezentujące punkty. Zwracane punkty dodawane są do listy punktów. Metoda zwraca listę utworzonych punktów.
-private static double calculateDistanceFromPointToLine(Point3D point3D) - oblicza i zwraca odległość między podanym w parametrze punktem a prostą, która została opisana w poleceniu pracy domowej. Odległość punktu od prostej liczona jest w taki sposób, że z klasy Vector3D pobierany jest wektor kierunkowy prostej, która została opisana w poleceniu pracy domowej. Następnie pobierany wektor P1P0 (również przy użyciu metody z klasy Vector3D), gdzie P1 to punkt, który jest podawany jako argument metody. P0=(0,0,0) to punkt leżący na w.w. prostej. Wyliczany jest iloczyn wektorowy wektora P1P0 z wektorem kierunkowym

prostej. Później wyznaczana jest długość otrzymanego wektora, która jest dzielona przez długość wektora kierunkowego. Tak otrzymany wynik jest zwracany.

-private static double calculateMomentOfInertia(double distance, double mass) - oblicza i zwraca moment bezwładności dla zadanej masy i odległości punktu od osi wyznaczonej przez prostą, która została opisana w poleceniu pracy domowej. Moment bezwładności jest wyznaczany zgodnie ze wzorem: $I = mR^2$

-private static double calculateRadiusOfInertia(List<Point3D> point3DList) - oblicza i zwraca promień bezwładności dla podanej listy punktów w 3D. Dla każdego punktu na liście wyznaczana jest odległość między tym punktem, a prostą, która została opisana w poleceniu pracy domowej. Następnie przy użyciu otrzymanej odległości punktu od prostej i masy punktu wyliczany jest moment bezwładności dla danego punktu. Momenty bezwładności poszczególnych punktów są ze sobą sumowane, tak samo jak ich masy (masy punktów). Na końcu obliczany jest pierwiastek kwadratowy z ilorazu sumy momentów bezwładności i sumy mas punktów, co wyraża się wzorem:

$$r_{gyration} = \sqrt{\frac{\sum r^2 \cdot m}{\sum m}}$$

-public static void main(String[] args) - w tej metodzie pobierany jest komponent EJB , który dostarcza danych do programu. Następnie z komponentu EJB pobierane są współczynniki a,b,c Później pobierane są dane na temat punktów w przestrzeni 3D i na ich podstawie tworzona jest lista obiektów reprezentujących owe punkty. Następnie dla wcześniej utworzonej listy punktów obliczany jest promień bezwładności, a otrzymany wynik jest wypisywany na ekran z dokładnością do 5 miejsc dziesiętnych.

3.Mechanizm wyszukiwania (lookup) i zestawiania połączenia.

Wyszukiwanie komponentu EJB zostało zrealizowane dzięki interfejsowi JNDI, który daje możliwość wyszukiwania i uzyskiwania dostępu do serwisów i katalogów przy użyciu przenośnych nazw. Zgodnie z dokumentacją Oracle nazwa, która określa dany zasób, powinna być zgodna ze wzorem:

```
java:global[/application name]/module name/enterprise bean name[/interface name]
```

gdzie **java:global** określa przestrzeń nazw, dzięki której możemy wyszukiwać zdalne EJB. Aby uzyskać dostęp do komponentu EJB o nazwie DataMonitor i deskryptorze ejb-project użyto następującej nazwy - **java:global/ejb-project/DataMonitor.**

Połączenie z komponentem EJB było zrealizowane poprzez utworzenie kontekstu (InitialContext), a następnie wywołanie na nim metody lookup, z parametrem będącym adresem komponentu.

4.Mechanizmu poszukiwania rozwiązania

Czynności wykonane przed implementacją:

Prostą wyznaczoną układem równań **3ax + 2by = 0** i **3by + 2cz = 0** Następnie wyznaczyłem punkt P0 =(0,0,0), który leży na tej prostej .Później wyznaczyłem wektory normalne dla poszczególnych równań: n1=(3a,2b,0) oraz n2=(0,3b,2c). Kolejnym krokiem było znalezienie wektora kierunkowego prostej (4bc,-6ac,9ab). W tym celu wykonałem mnożenie wektorowe wektorów n1 oraz n2 (n1xn2). Do celów obliczenia odległości punktu od prostej przekształciłem układ równań opisujący prostą do postaci: **x=4bct, y=-6act, z=9abt**

Program:
Pierwszą czynnością jaką wykonuje program jest wyszukanie oraz nawiązanie połączenia z komponentem EJB znajdującym się na serwerze. W tym celu tworzony jest kontekst, a następnie dla nazwy **java:global/ejb-project/DataMonitor** wyszukiwany jest EJB, który jej odpowiada. Jeśli bean został znaleziony następuje nawiązania z nim połączenia. Następnie z komponentu EJB pobierane są współczynniki a,b,c prostej opisanej w.w. równaniami. W następnej kolejności pobierane są czwórki liczb (współrzędne x,y,z oraz masa), z których tworzone są obiekty reprezentujące punkty w przestrzeni 3D. Dane pobierane są do momentu gdy metoda hasNext() z komponentu DataMonitor zwraca true. Punkty są tworzone tylko wtedy kiedy każda z czwórki liczb nie jest wartością null. Tworzone punkty gromadzone są w liście. Następnie dla każdego punktu na liście wyznaczana jest jego odległość od prostej **x=4bct, y=-6act, z=9abt**, zgodnie ze wzorem:

$$\frac{\left|\vec{PP_0} \times \vec{u}\right|}{|\vec{u}|}$$

gdzie punkt P to aktualnie badany punkt, P0 =(0,0,0) to punkt leżący na prostej **3ax + 2by = 0** i **3by + 2cz = 0,** wektor u = (4bc,-6ac,9ab) to wektor kierunkowy prostej. Na wektorach PP0 oraz u wykonywane jest mnożenie wektorowe. Następnie wyznaczana jest długość otrzymanego wektora i dzielona jest przez długość wektora kierunkowego prostej.

Przy użyciu otrzymanej odległości punktu od prostej i masy punktu wyliczany jest moment bezwładności dla danego punktu, zgodnie ze wzorem:

$I = mR^2$, gdzie R to odległość punktu od prostej, m to masa punktu.

Momenty bezwładności poszczególnych punktów są ze sobą sumowane, tak samo jak ich masy (masy punktów). Na końcu obliczany jest pierwiastek kwadratowy z ilorazu sumy momentów bezwładności i sumy mas punktów, co wyraża się wzorem:

$$r_{gyration} = \sqrt{\frac{\sum r^2 \cdot m}{\sum m}}$$

Otrzymany wynik jest następnie wypisywany z dokładnością do 5 miejsc dziesiętnych, przy użyciu metody printf i formatowania "%.5f".