Java Advanced Assignment:

1. **SAX Parser :**

* **MyHandler.java :**

package com.accolite.xml.sax;

import java.util.ArrayList;

import java.util.List;

import org.xml.sax.Attributes;

import org.xml.sax.SAXException;

import org.xml.sax.helpers.DefaultHandler;

public class MyHandler extends DefaultHandler {

private List<Employee> empList = null;

private Employee emp = null;

private StringBuilder data = null;

public List<Employee> getEmpList() {

return empList;

}

boolean bfName = false;

boolean blName = false;

boolean bAge = false;

boolean bGender = false;

boolean blocation = false;

@Override

public void startElement(String uri, String localName, String qName, Attributes attributes) throws SAXException {

if (qName.equalsIgnoreCase("Employee")) {

String id = attributes.getValue("id");

emp = new Employee();

emp.setId(Integer.parseInt(id));

if (empList == null)

empList = new ArrayList<>();

} else if (qName.equalsIgnoreCase("firstName")) { variables

bfName = true;

} else if (qName.equalsIgnoreCase("lastName")) {variables

blName = true;

} else if (qName.equalsIgnoreCase("age")) {

bAge = true;

} else if (qName.equalsIgnoreCase("gender")) {

bGender = true;

} else if (qName.equalsIgnoreCase("location")) {

blocation = true;

}

data = new StringBuilder();

}

@Override

public void endElement(String uri, String localName, String qName) throws SAXException {

if (bAge) {

emp.setAge(Integer.parseInt(data.toString()));

bAge = false;

} else if (bfName) {

emp.setfirstName(data.toString());

bfName = false;

} else if (blName) {

emp.setlastName(data.toString());

blName = false;

} else if (blocation) {

emp.setLocation(data.toString());

blocation = false;

} else if (bGender) {

emp.setGender(data.toString());

bGender = false;

}

if (qName.equalsIgnoreCase("Employee")) {

empList.add(emp);

}

}

@Override

public void characters(char ch[], int start, int length) throws SAXException {

data.append(new String(ch, start, length));

}

}

* **XMLParserSAX.java :**

package com.accolite.xml.sax;

import com.accolite.annotations.\*;

import java.io.File;

import java.io.IOException;

import java.util.List;

import javax.xml.parsers.ParserConfigurationException;

import javax.xml.parsers.SAXParser;

import javax.xml.parsers.SAXParserFactory;

import org.xml.sax.SAXException;

public class XMLParserSAX {

public static void main(String[] args) {

SAXParserFactory saxParserFactory = SAXParserFactory.newInstance();

try {

SAXParser saxParser = saxParserFactory.newSAXParser();

MyHandler handler = new MyHandler();

saxParser.parse(new File("Demo.xml"), handler);

List<Employee> empList = handler.getEmpList();

for(Employee emp : empList)

System.out.println(emp);

} catch (ParserConfigurationException | SAXException | IOException e) {

e.printStackTrace();

}

}

}

* **Employee.java :**

package com.accolite.xml.sax;

public class Employee {

private int id;

private String firstName;

private String lastName;

private int age;

private String gender;

private String location;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getfirstName() {

return firstName;

}

public void setlastName(String lastName) {

this.lastName = lastName;

}

public String getlastName() {

return lastName;

}

public void setfirstName(String firstName) {

this.firstName = firstName;

}

public String getGender() {

return gender;

}

public void setGender(String gender) {

this.gender = gender;

}

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

public String getLocation() {

return location;

}

public void setLocation(String location) {

this.location = location;

}

@Override

public String toString() {

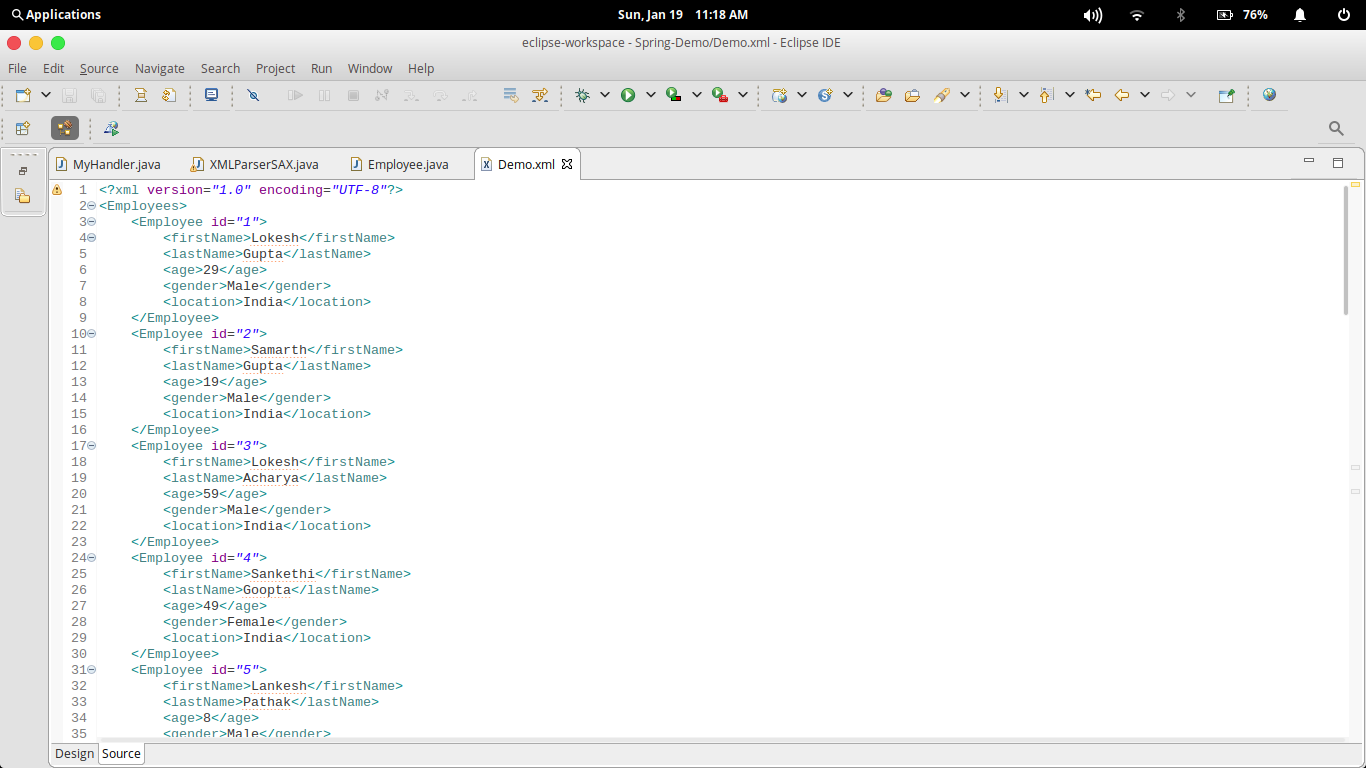
return "Employee:: ID=" + this.id + " First Name=" + this.firstName +" Last Name=" + this.lastName + " Age=" + this.age + " Gender=" + this.gender

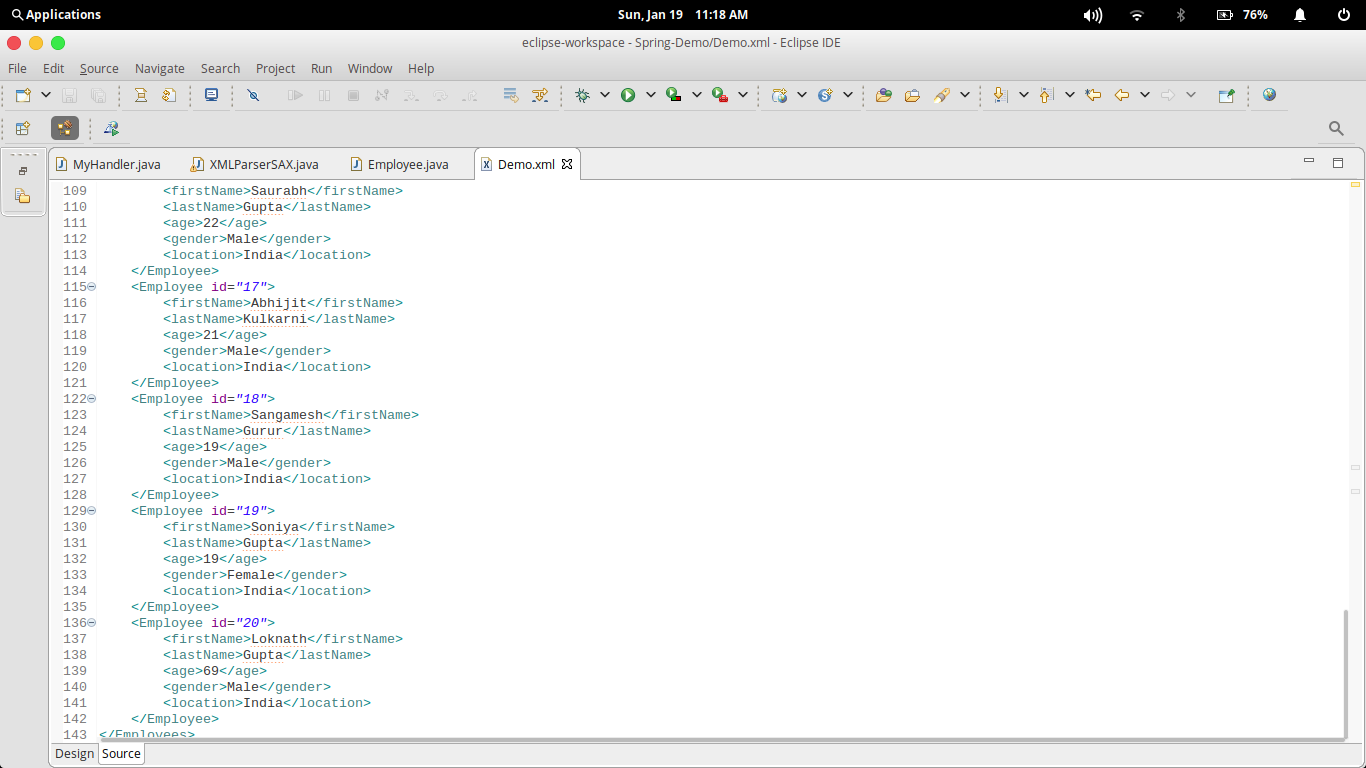
+ " Location =" + this.location;

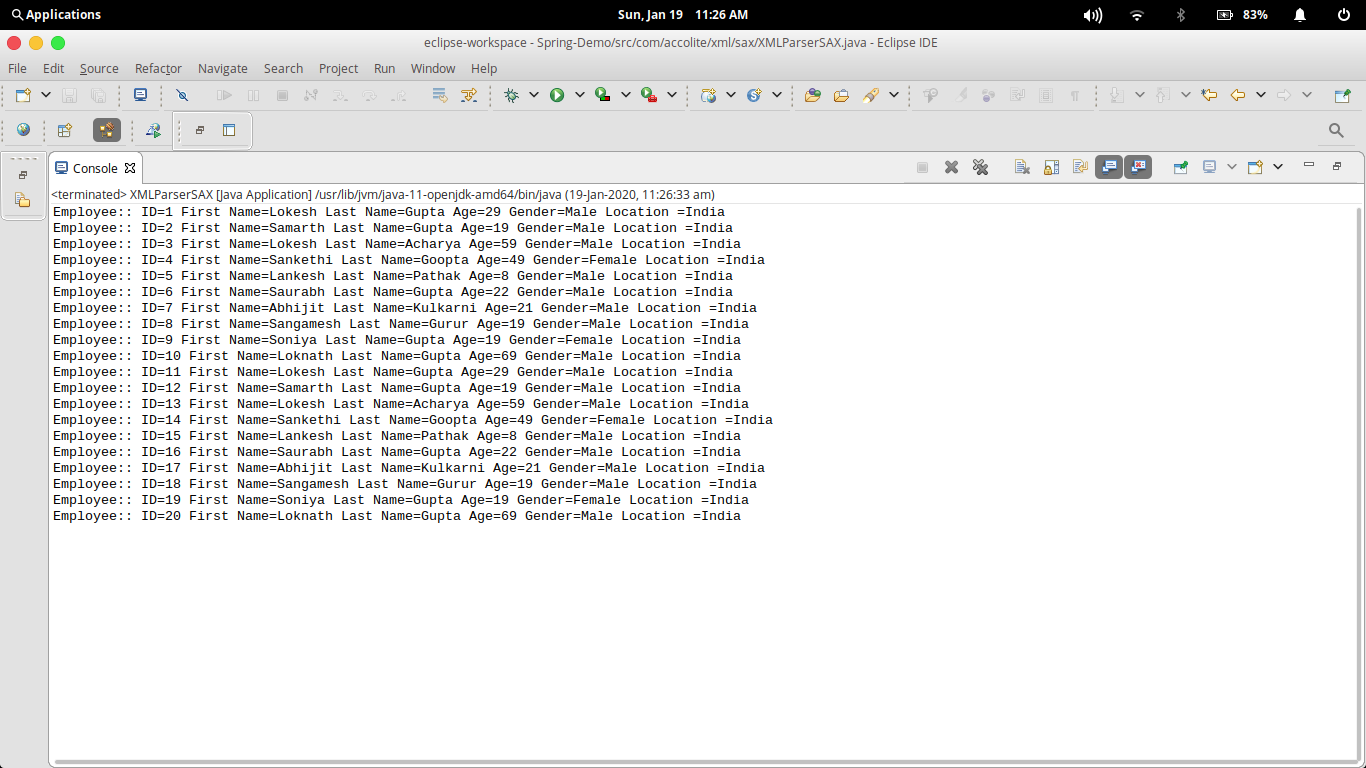
}

}

* **Input File :**

****

****

* **Output :**

**2. Create some meaningful FULL Annotation for Method and Class target types.**

* **TestCustomAnnotation1.java :**

package com.accolite.annotations;

import java.lang.reflect.Method;

public class TestCustomAnnotation1 {

public static void main(String args[])throws Exception{

Hello h = new Hello();

Method m = h.getClass().getMethod("sayHello");

MyAnnotation manno = m.getAnnotation(MyAnnotation.class);

System.out.println("Value is: "+manno.age());

}

}

* **MyAnnotation.java :**

package com.accolite.annotations;

import java.lang.annotation.ElementType;

import java.lang.annotation.Retention;

import java.lang.annotation.RetentionPolicy;

import java.lang.annotation.Target;

@Retention(RetentionPolicy.RUNTIME)

@Target(ElementType.METHOD)

public @interface MyAnnotation {

int age() default 0;

}

class Hello{

@MyAnnotation(age=10)

public void sayHello(){System.out.println("Hello from annotation");}

}

**3. Design your own TriConsumer lambda.**

@FunctionalInterface

interface Triconsumer<T1,T2,T3>{

void accept(T1 t1 ,T2 t2,T3 t3);

}

public class triconsumerlambda {

public static void main(String args[]) {

Triconsumer<Integer, Integer, Integer> t = (a,b,c) -> System.out.println(a+b+c);

t.accept(15, 10, 5);

}

}

**4. Demonstrate Exception handling in Lambda's using wrapper lambdas.**

import java.util.Scanner;

public class Exceptionlambdas {

public static void main(String[] args) {

// Lambda Function

float numerator = 0, denominator = 0;

try (Scanner s = new Scanner(System.in)) {

System.out.println("Enter the numerator :");

numerator = s.nextFloat();

System.out.println("Enter the denominator :");

denominator = s.nextFloat();

}

catch(Exception e) {

System.out.println("Enter proper input. Initialising numerator and denominator to zero.");

}

process(numerator, denominator, divideWrapper((num, den) -> System.out.println(num/den)));

}

public static interface DivideLambda{

public void divide(float numerator, float denominator);

}

private static void process(float numerator, float denominator, DivideLambda divider) {

divider.divide(numerator, denominator);

}

private static DivideLambda divideWrapper(DivideLambda dividelambda) {

return (numerator,denominator) -> {

try {

if(denominator == 0 )

throw new ArithmeticException();

System.out.println(numerator/denominator);

}

catch(ArithmeticException e) {

System.out.println("Can't Divide by Zero.");

}

};

}

}

**5. Create 2 tables in db having some kind of relationship, create a stored procedure which joins the two tables and returns columns in any db and call it using callable statements and map it to a model object.**

import java.sql.\*;

public class databaseJDBC {

public static void main(String[] args) {

Connection connection = null;

try {

connection = DriverManager.getConnection("jdbc:mysql://localhost/demo?" + "user=root&password=swarev123");

Statement statement = connection.createStatement();

ResultSet resultSet = statement.executeQuery("{call DemoProcedure()}");

while(resultSet.next())

System.out.println(resultSet.getInteger(1)+" "+resultSet.getString(2)+" "+rs.getInteger(3)+" "+rs.getInteger(4)+" "+rs.getString(5));

connection.close();

} catch (Exception exception) {

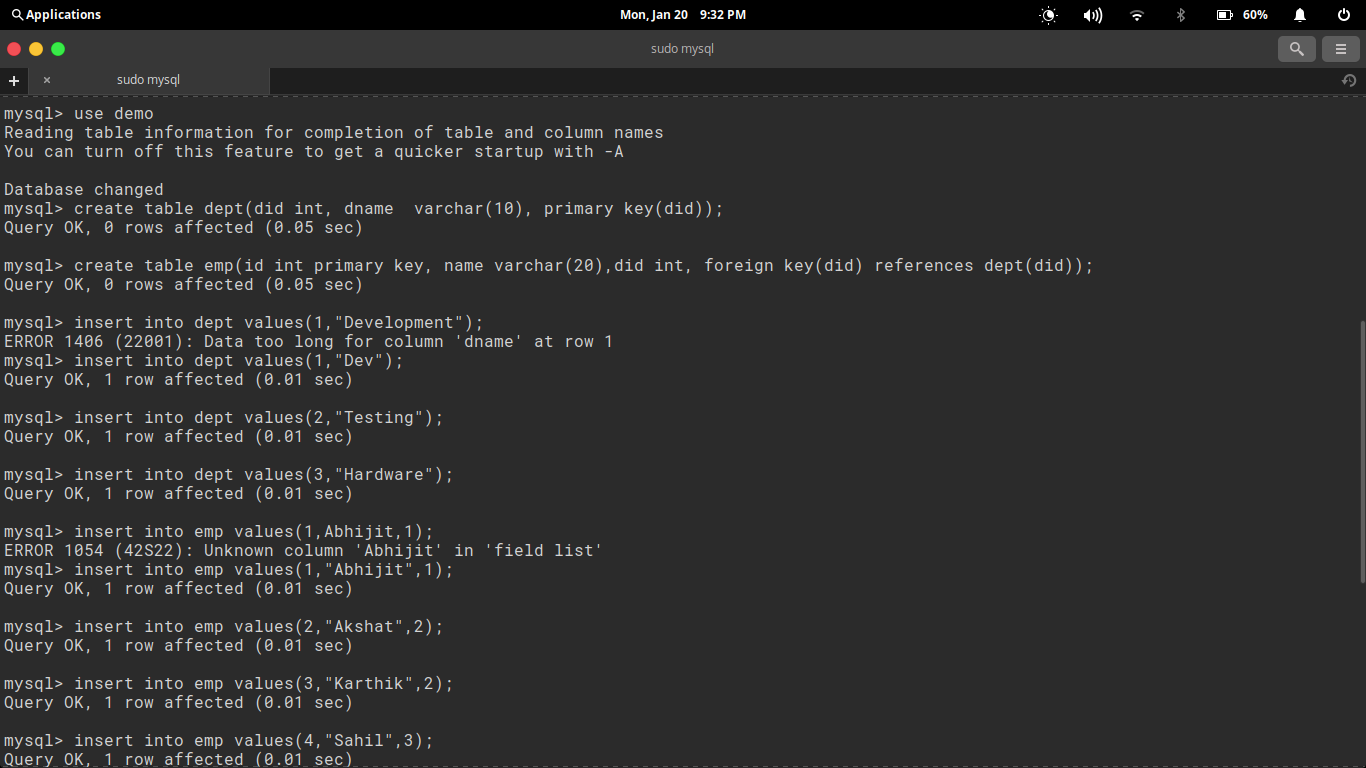
System.out.println("Operation can not be completed. Please try again.");

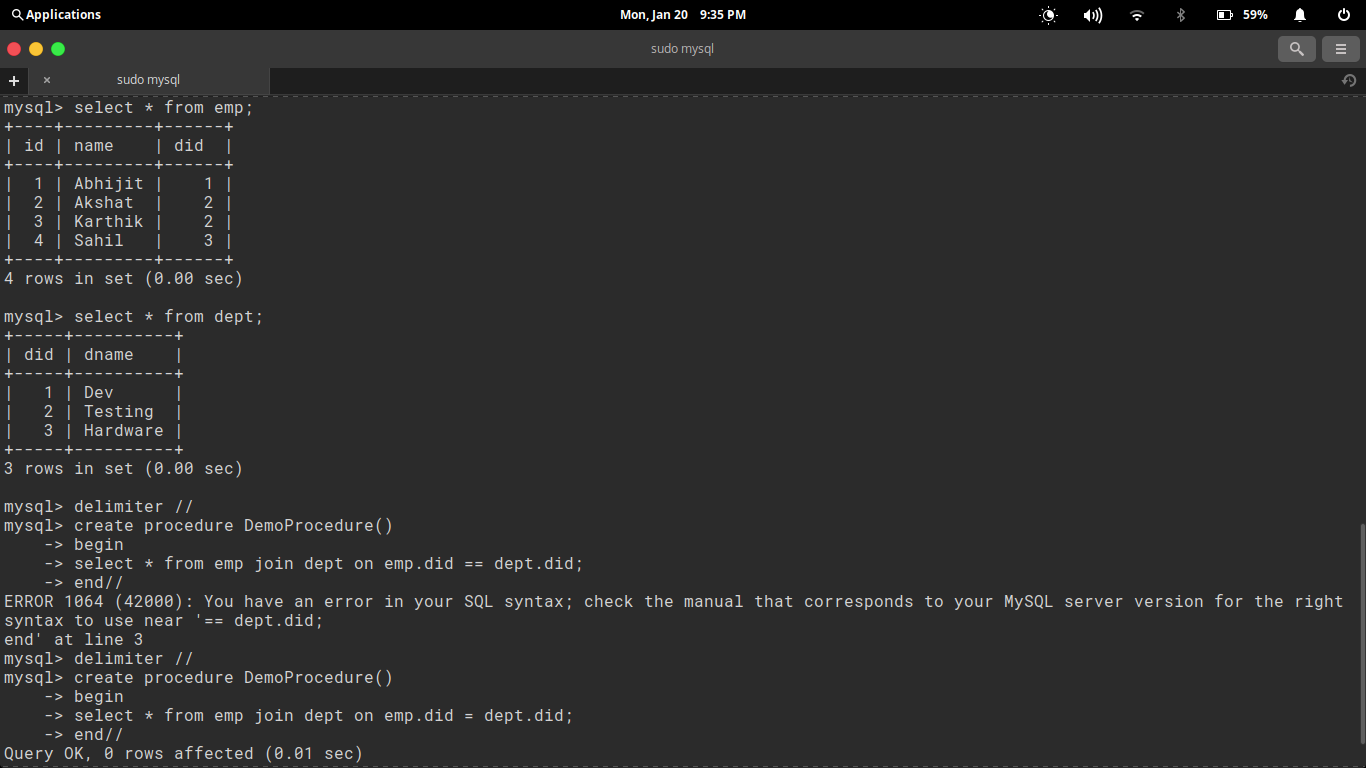
}

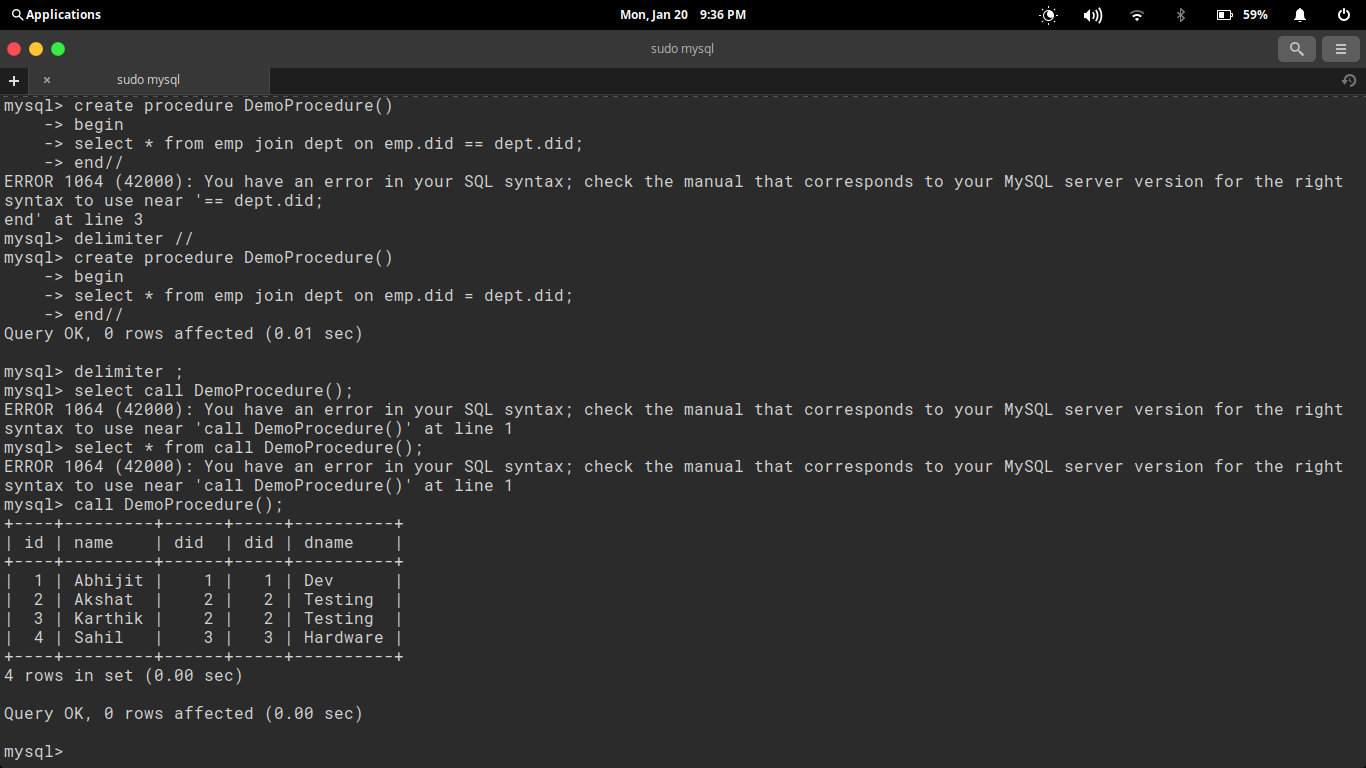
}

}

**Output :**

****



****

**6. Create a method which takes two strings as inputs (Your dob and your parent/sibling date of birth) and returns the difference in terms of :**

**1. The number of days of difference between two in nano seconds :**

import java.util.Scanner;

import java.util.concurrent.TimeUnit;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;

public class DobNanoseconds {

public static long getDateDiff(Date date1, Date date2, TimeUnit timeUnit) {

long diffInMillies = date2.getTime() - date1.getTime();

return timeUnit.convert(diffInMillies,TimeUnit.NANOSECONDS);

}

public static void main(String[] args) {

String myDOB="";

String parentDOB="";

Date dateMyDOB = null;

Date dateParentDOB = null;

try {

@SuppressWarnings("resource")

Scanner s = new Scanner(System.in);

System.out.println("Enter your DOB :");

myDOB = s.nextLine();

System.out.println("Enter your Parent DOB :");

parentDOB = s.nextLine();

} catch(Exception e) {

System.out.println("Can't initiate Scanner.");

}

try {

dateMyDOB = new SimpleDateFormat("dd/MM/yyyy").parse(myDOB);

dateParentDOB = new SimpleDateFormat("dd/MM/yyyy").parse(parentDOB);;

} catch (ParseException e) {

// TODO Auto-generated catch block

e.printStackTrace();

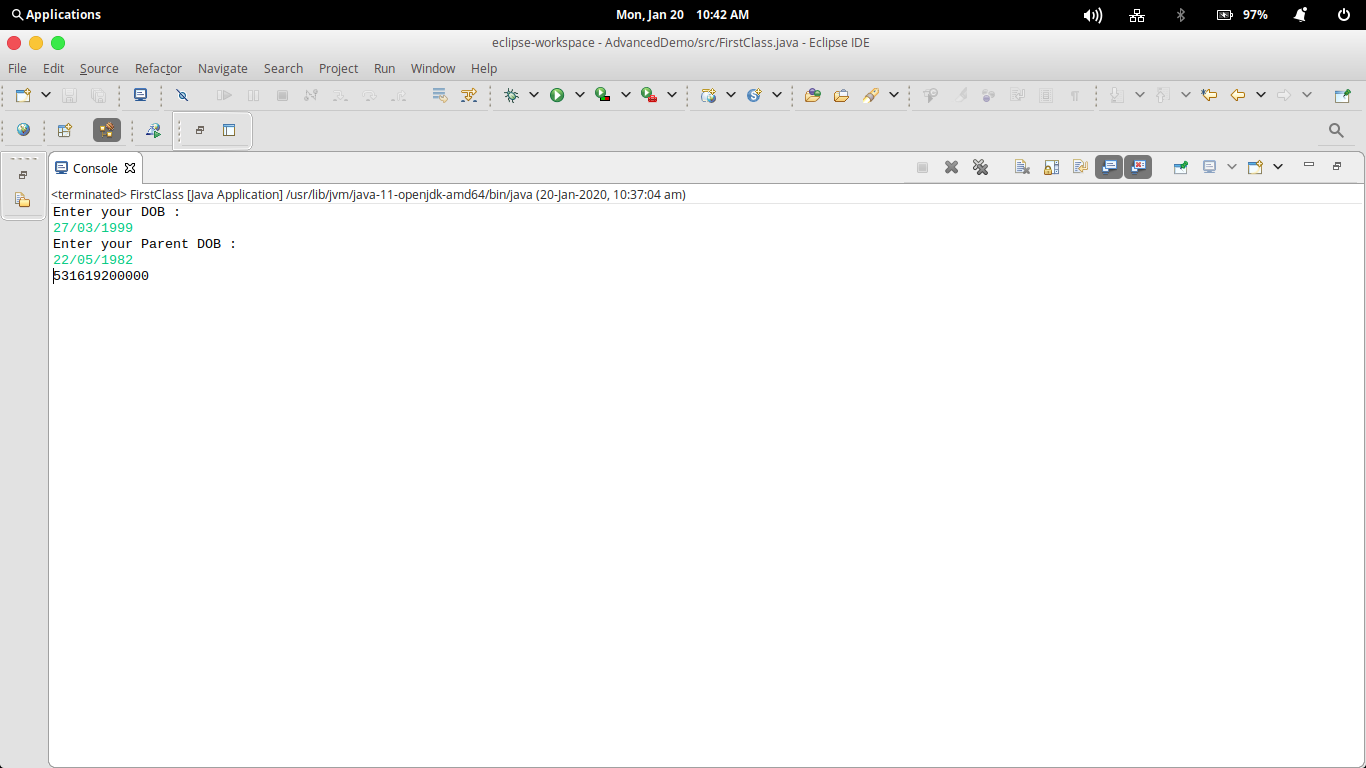
}

System.out.println(getDateDiff(dateParentDOB, dateMyDOB, TimeUnit.NANOSECONDS));

}

}

**OUTPUT :**

****

**2. Consider your dob in different time zone and then convert your parent/sibling dob in that time zone and find difference in days :**

import java.text.ParseException;

import java.time.LocalDateTime;

import java.time.ZonedDateTime;

import java.time.temporal.ChronoUnit;

import java.util.Scanner;

public class DobTimezone{

public static void main(String[] args) throws ParseException

{

String MyDOB,ParentDOB;

try{

Scanner input = new Scanner(System.in);

System.out.println("Enter Your DOB in the Format(YYYY-MM-DDTHH:MM:SS) :");

MyDOB = input.next();

System.out.println("Enter Your Parent DOB in the Format(YYYY-MM-DDTHH:MM:SS) :");

ParentDOB = input.next();

} Catch(Exception e) {

System.out.println("Can not read input.");

}

LocalDateTime MyDOBTime = LocalDateTime.parse(MyDOB);

System.out.println("My DateTime: " + MyDOBTime);

LocalDateTime ParentDOBTime=LocalDateTime.parse(ParentDOB);

System.out.println("Parent DateTime: " + ParentDOBTime);

ZonedDateTime Zone1=ZonedDateTime.parse(MyDOB);

System.out.println(Zone1);

ZonedDateTime Zone2=ZonedDateTime.parse(ParentDOB);

System.out.println(Zone2);

ChronoUnit CDays=ChronoUnit.DAYS;

System.out.println(CDays.between(Zone2, Zone1));

}

}

**OUTPUT :**

