**Question 1 (33 marks)**

The following Java class contains member variables and methods to set up a database connection and to drop a database table called Coffee if it exists.

**package** qst;

**import** java.sql.\*;

**import** oracle.jdbc.pool.OracleDataSource;

**public** **class** CreateCoffeeDB {

**private** Connection conn;

**private** PreparedStatement pstmt;

**private** ResultSet rset;

**public** **void** openDB() {

**try** {

// Load the Oracle JDBC driver

OracleDataSource ods = **new** OracleDataSource();

ods.setURL("jdbc:oracle:thin:HR/@localhost:1521:XE");

ods.setUser("test");

ods.setPassword("test1");

conn = ods.getConnection();

System.*out*.println("connected.");

} **catch** (Exception e) {

System.*out*.print("Unable to load driver " + e);

System.*exit*(1);

}

}

**public** **void** dropTables() {

System.*out*.println("Checking for existing tables.");

**try** {

pstmt = conn.prepareStatement("DROP TABLE Coffee");

**try** {

pstmt.execute();

System.*out*.println("Coffee table dropped.");

} **catch** (SQLException ex) {

}

} **catch** (SQLException ex) {

System.*out*.println("ERROR: " + ex.getMessage());

ex.printStackTrace();

}

}

**public** **void** closeDB() {

**try** {

pstmt.close();

rset.close();

conn.close();

System.*out*.print("Connection closed");

} **catch** (SQLException e) {

System.*out*.print("Could not close connection ");

e.printStackTrace();

}

}

}

package qst;

public class TestCoffeeDB {

public static void main(String[] args) {

CreateCoffeeDB cdb = new CreateCoffeeDB();

cdb.openDB();

cdb.dropTables();

//cdb.fillCoffeeTable();

//cdb.queryDB();

cdb.closeDB();

}

}

1. Using Prepared Statements write the Java code for a method fillCoffeetable() to create a table called Coffee and fill it with the following data:

|  |  |  |
| --- | --- | --- |
| **Description** | **Product Code** | **Price** |
| Bolivian Dark | 14-001 | 8.95 |
| Bolivian Medium | 14-002 | 7.95 |
| Brazilian Medium | 15-002 | 7.95 |

(16 marks)

1. Write the Java code for a queryDB() method that prints out all the data in the Coffee table in the following format:

Bolivian Dark 14-001 8.95

Bolivian Medium 14-002 7.95

Brazilian Medium 15-002 7.95

(9 marks)

1. SQL queries can be built in Java using either the Statement interface or the Prepared Statement interface. Explain the difference between the two.

(5 marks)

1. List three Statement methods used to execute SQL in Java.

(3 Marks)

**Question 2 (33 Marks)**

The following code contains class definitions and annotations for the University and Department classes that are part of the HEA’s college register system. Note that not all getter and setter methods are shown here.

**University Class**

package code;

import java.util.\*;

import javax.persistence.\*;

import static javax.persistence.CascadeType.ALL;

@Entity

@Table (name="UNIVERSITY")

public class University {

@Id

private int id;

private String uName;

@OneToMany(cascade = ALL, mappedBy = "u")

private List<Department> deptlist = new ArrayList<>();

public University() {

}

public University(int id, String uName) {

this.id = id;

this.uName = uName;

}

public List<Department> getDeptlist() {

return deptlist;

}

@Override

public String toString() {

System.out.println("Size"+deptlist.size());

String s = "University ID: " + id + " Name: " + uName + "\n";

for (int i = 0; i < deptlist.size(); i++) {

s += deptlist.get(i);

}

return s;

}

}

Department Class

package code;

import java.io.Serializable;

import javax.persistence.\*;

@Entity

@Table(name = "DEPARTMENT")

public class Department implements Serializable {

@Id

@Column(name="ID")

private int deptID;

@Column(name="NAME")

private String deptName;

@ManyToOne()

@JoinColumn(name = "UNIID")

private University u;

public Department() {

}

public Department(int deptID, String deptName) {

this.deptID = deptID;

this.deptName = deptName;

}

public void setU(University u) {

this.u = u;

}

@Override

public String toString() {

return "Dep ID: " + deptID + " Dep Name: " + deptName +"\n";

}

}

1. Write the Java code for a Test class to create Java objects based on the following data and persist them to the database.

( 16 Marks)

|  |  |
| --- | --- |
| University ID | University Name |
| 1 | DCU |
| 2 | UCD |

|  |  |  |
| --- | --- | --- |
| Department ID | Department ID | University ID |
| 1 | Computing | 1 |
| 2 | Business | 1 |
| 3 | Languages | 2 |

1. The Languages department needs to be deleted from UCD, this can be done a few different ways. Show with the aid of Java code two different ways of performing the delete in the test class.

(13 Marks)

1. What is JPQL and what is it used for?

(4 Marks)

**Question 3 (33 Marks)**

The following code contains class definitions for a simple BankAccount class and a TestBankAccount class.

package bankAcc;

public class BankAccount {

private String owner, acctID;

private double balance;

public BankAccount( String name, String ID, double initialBalance ) {

owner = name;

acctID = ID;

balance = initialBalance;

}

public String getOwner( ) {

return owner;

}

public String getID( ) {

return acctID;

}

public double getBalance( ) {

return balance;

}

public void deposit( double amount ){

balance = balance + amount;

}

public void withdraw( double amount ) {

balance = balance - amount;

}

@Override

public String toString( ) {

return "BankAccount owned by: " + owner + " ID: "

+ acctID + " balance: €" + balance;

}

}

**TestBankAccount**

package bankAcc;

import java.util.\*;

public class TestBankAccount

{

private HashMap<String, BankAccount> accounts;

public TestBankAccount()

{

accounts = new HashMap<>();

}

public static void main(String[] args) {

TestBankAccount d = new TestBankAccount();

d.addAccount("111111", "Mary", 1500.0);

d.addAccount("222222", "John", 10.55);

d.addAccount("333333", "Bill", 10000000001.95);

d.showAccount("333333");

d.showAccount("5");

System.out.println("\*\*\*\*\*\*\*\*\*\*\*");

d.showAllAccounts();

System.out.println("Adding two new accounts...");

d.addAccount("444444", "Kevin", 50000.0);

d.addAccount("555555", "Sam", 500000.0);

d.showAllAccounts();

}

}

The class TestBankAccount has one member variable which is a data type of accounts defined as a HashMap<String, BankAccount>

Write the Java code for the following methods in the TestBankAccount class using the method calls from the main method as a guide:

1. addAccount() which creates a new BankAccount and adds it to the map.

(4 marks)

1. showAccount() which displays a description of the BankAccount associated with the account number provided, or an error message if there is no account with that number.

(6 Marks)

1. showAllAccounts() which displays all the bank accounts in the collection by using an Iterator to iterate through the HashMap's key set

(10 Marks)

1. Describe the output produced when this program is executed.

(4 Marks)

1. Describe three features of a HashMap?

(9 Marks)

**Question 4 (33 Marks)**

1. Write a short Java program that creates a linked list to hold integers, adds three integers to it and print out the contents.

(7 Marks)

1. In relation to the code you have just written explain what you understand by the term “parameterised class”

(4 Marks)

1. The following Java code calls a generic method printAarray to display the contents of the different types of arrays. Write the code for this generic method.

package generics;

public class GenericMethodTestQst {

public static void main(String args[]) {

Integer[] intArray = {1, 2, 3, 4, 5};

Double[] doubleArray = {10.1, 21.2, 33.3, 44.4};

Character[] charArray = {'H', 'E', 'L', 'L', 'O'};

System.out.println("Array integerArray contains:");

printArray(intArray); // pass an Integer array

System.out.println("\nArray doubleArray contains:");

printArray(doubleArray); // pass a Double array

System.out.println("\nArray characterArray contains:");

printArray(charArray); // pass a Character array

}

}

(6 Marks)

1. What is the purpose of bounded type parameters in Generics?

(5 Marks)

1. If you intend to use a HashMap with your own class as the parameterized type then your class must override two methods. Describe these two methods and their function.

(6 Marks)

1. Describe two features of a List in Java.

(5 Marks)

**Question 5 (33 Marks)**

The following code contains database methods to open and close a connection to the Oracle database.

package qst;

import java.sql.\*;

import oracle.jdbc.pool.OracleDataSource;

public class ProductOperations {

private PreparedStatement pstmt;

private ResultSet rset;

private Connection conn;

public ProductOperations()

{

conn = openDB();

}

public Connection openDB() {

try {

OracleDataSource ods = new OracleDataSource();

// Tallaght

ods.setURL("jdbc:oracle:thin:@//10.10.2.7:1521/global1");

ods.setUser("user1");

ods.setPassword("pass1");

conn = ods.getConnection();

System.out.println("connected.");

} catch (Exception e) {

System.out.print("Unable to load driver " + e);

System.exit(1);

}

return conn;

}

public void closeDB() {

try {

conn.close();

System.out.print("Connection closed");

} catch (SQLException e) {

System.out.print("Could not close connection ");

e.printStackTrace();

}

}

}

The names and type of the columns in a products table are shown below:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Product ID  (Sequence | Name  (VARCHAR) | Brand  (VARCHAR) | Price  (DECIMAL) | Qty  (NUMBER) | Category  (VARCHAR) | Branch  (VARCHAR) |

1. Using Prepared Statements write the Java code for a method addProduct() in the ProductOperations class to add a new product into the database table. All values for the columns are passed as parameters into the method except for Product ID which uses a sequence.

(8 Marks)

1. Write the Java code for a getProducts() method in the ProductOperations class that returns a result set where the name has “LED” and the price is below 500.00.

(6 Marks)

1. In relation to the code supplied for connecting a Java application to an Oracle database, explain how the classes used work to perform this connectivity.

(6 Marks)

1. The closedb() method in the code provided at the start contains a try catch block for the SQLException. Why is this required?

(4 Marks)

1. A ResultSet maintains a cursor pointing to a particular row of data. Once positioned at a row, the data can be extracted on a column by column basis. Show with the aid of Java code how the information is extracted from a column.

(9 Marks)

**Question 6 (33 Marks)**

The following code contains class definitions for the Staff and Contact classes that are part of a company’s new staff management system.

package ex1;

import java.util.\*;

public class Staff {

private int id;

private String name;

private List<Contact> clist;

public Staff() {

clist = new ArrayList<>();

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public List<Contact> getClist() {

return clist;

}

public void setClist(List<Contact> clist) {

this.clist = clist;

}

}

package ex1;

import java.util.\*;

public class Contact {

private int id;

private String name;

private List<Staff> slist;

public Contact() {

slist = new ArrayList<>();

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public List<Staff> getSlist() {

return slist;

}

public void setSlist(List<Staff> slist) {

this.slist = slist;

}

public void addStaff(Staff s) {

slist.add(s);

s.getClist().add(this);

}

public void removeStaff(Staff s) {

slist.remove(s);

s.getClist().remove(this);

}

public void remove() {

ArrayList<Staff> temp = new ArrayList<>(slist);

for (Staff temp1 : temp) {

removeStaff(temp1);

}

}

}

The following tables show the structure of the STAFF, CONTACT & STAFF\_CONTACT tables in the Oracle database:

STAFF Table

|  |  |
| --- | --- |
| ID (Primary Key) | Name |

CONTACT Table

|  |  |
| --- | --- |
| ID (Primary Key) | NAME |

CONTACT\_STAFF Table – Associate table

|  |  |
| --- | --- |
| SID (Foreign Key which references ID in STAFF) | CID (Foreign Key which references ID in CONTACT) |

The Java Persistence API provides Java developers with an object/relational mapping facility for managing relational data in Java applications to the underlying database.

1. Annotate the appropriate code segments in each class to mark them as persistent entities and explain the purpose of each annotation. Include any extra imports that may be needed in order to use these annotations.

( 7 Marks)

1. Annotate and explain the appropriate mapping for the two entities bearing in mind that a staff member can have many contacts and the same contact can be shared among many staff.

(13 Marks)

1. Write the Java code for a test class that creates one Contact and two Staff objects and persists these to the database. The contact is added to each staff member’s list of contacts.

(13 Marks)

### Question 7 (33 Marks)

Consider the following class TestGroup which contains a main method to test a generic class called Group. A group has a capacity and holds generic items in it. The add() method is used to add generic items to the group. If the group is full when the add() method is called then one of the existing items in the group is chosen at random, and is replaced by the new one. The get() method returns one of the items, randomly chosen, but does not modify the group.

package ex2;

public class TestGroup {

public static void main(String[] args) {

Group<String> b = new Group<>(3);

b.add("Apple");

b.add("is");

System.out.println(b);

b.add("Green");

System.out.println(b);

String sample = b.get();

System.out.println(sample);

System.out.println(b);

b.add("Juicy");

System.out.println(b);

}

}

1. Write the Java code for the generic Group class using an array to store its contents. You may define additional instance variables as required. When the test program is executed it should produce output similar to that shown below:

Sample Output

Apple is

Apple is Green

Apple

Apple is Green

Apple Juicy Green | Juicy is Green | Apple is Juicy

(17 Marks)

1. The following program stores a collection of strings in an array list and uses a for loop to remove the elements of the list.

package q;

import java.util.\*;

public class TestArrayListEx1 {

public static void main(String[] args) {

List<String> colours = new ArrayList<>();

colours.add("Red");

colours.add("Green");

colours.add("Yellow");

colours.add("Blue");

colours.add("Pink");

for(int i=0;i<colours.size();i++)

{

colours.remove(colours.get(i));

}

System.out.println(colours);

}

}

1. Explain the output this program generates and why?

(5 Marks)

1. If an enhanced loop is used instead to remove the elements in the list, explain what output the program generates and why?

(5 Marks)

1. Write the Java code to traverse through the list using an iterator and remove all the elements.

(4 Marks)

1. Name one method in the ArrayList class that can also be used to empty the contents of an array list.

(2 Marks)

**Question 8 (33 Marks)**

1. Write the method removeAdjacentEvens, which removes from a linked list any even numbers that directly follow another even number in the list.

Sample Output:

list (6 2 5 2 8 4 3)

list after call to removeAdjacentEvens(list): (6 5 2 3)

(15 Marks)

1. Show the results of inserting the following six keys that have the following hash values into an empty hash table that starts out empty, has 10 buckets (indices 0 through 9), and uses chaining. Do the insertions in the order the keys are shown (Mary-Sue).

|  |  |
| --- | --- |
| Key | Hash Value |
| Mary | 9 |
| Joe | 1 |
| Ted | 9 |
| Bob | 9 |
| Dan | 4 |
| Sue | 7 |

(4 Marks)

For each of the following lookups from the table above, after the insertions are done, give the sequence of keys that the target key would have to be compared with to do the lookup

1. lookup Tom; Tom hashes to 0
2. lookup Ted; Ted hashes to 9

(4 Marks)

1. Write the Java code to create a collection of country names and their corresponding capital cities. Use a suitable collection to store this type of data and provide the code to add an entry to the collection, ask the user to enter a country name, look up the capital city of the country and print the capital city to the console.

(10 Marks)

**Question 9 (33 Marks)**

The following code contains database methods to open and close a connection to the Oracle database.

package qst ;

import java.sql.\*;

import oracle.jdbc.pool.OracleDataSource;

public class ProductOperations {

private PreparedStatement pstmt;

private ResultSet rset;

private Connection conn;

public ProductOperations()

{

conn = openDB();

}

public Connection openDB() {

try {

OracleDataSource ods = new OracleDataSource();

// Tallaght

ods.setURL("jdbc:oracle:thin:@//10.10.2.7:1521/global1");

ods.setUser("user1");

ods.setPassword("pass1");

conn = ods.getConnection();

System.out.println("connected.");

} catch (Exception e) {

System.out.print("Unable to load driver " + e);

System.exit(1);

}

return conn;

}

public void closeDB() {

try {

conn.close();

System.out.print("Connection closed");

} catch (SQLException e) {

System.out.print("Could not close connection ");

e.printStackTrace();

}

}

}

The names and type of the columns in a products table are shown below:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Product ID  (Sequence | Name  (VARCHAR) | Brand  (VARCHAR) | Price  (DECIMAL) | Qty  (NUMBER) | Category  (VARCHAR) | Branch  (VARCHAR) |

1. Using Prepared Statements write the Java code for a method in the ProductOperations class called searchByBrand() that takes one parameter representing the brand and returns all information from the products table based on this search parameter.

(6 Marks)

1. Write the Java code for a getProducts() method in the ProductOperations class that returns a result set from the products table where the branch contains the word “Dublin” and the price is between 500.00 and 700.00. All search criteria are passed as parameters into the method.

(7 Marks)

1. Describe the function of the executeQuery and executeUpdate methods of the Statement interface in Java. Make reference to the return type of each method.

(6 Marks)

1. What are the advantages of using an Advanced Result Set?

(6 Marks)

1. A JDBC driver's default is set to autocommit. Discuss i) what this means in a database context and ii) a Business Information Systems example where autocommit may be undesirable.

(8 Marks)

**Question 10 (33 Marks)**

The following code contains class definitions for the Employee and Department classes that are part of a company’s new employee management system.

package q;

public class Employee {

private int id;

private String name;

private double salary;

private Department dep;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public double getSalary() {

return salary;

}

public void setSalary(double salary) {

this.salary = salary;

}

public Department getDepartment() {

return dep;

}

public void setDepartment(Department department) {

this.dep = department;

}

@Override

public String toString() {

return "Employee id: " + getId() + " name: " + getName() +

" with " + getDepartment();

}

}

package q;

import java.util.\*;

public class Department {

private int id;

private String name;

private List<Employee> employees;

public Department() {

employees = new ArrayList<>();

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String deptName) {

this.name = deptName;

}

public void addEmployee(Employee employee) {

this.employees.add(employee);

employee.setDepartment(this);

}

public List<Employee> getEmployees() {

return employees;

}

public void setEmployees(List<Employee> employees) {

this.employees = employees;

}

@Override

public String toString() {

return "Department id: " + getId() +", name: " + getName();

}

}

The following tables show the structure of the Employee and Department tables in the Oracle database:

DEPARTMENT Table

|  |  |
| --- | --- |
| ID (Primary Key) | Name |

EMPLOYEE Table

|  |  |  |  |
| --- | --- | --- | --- |
| ID (Primary Key) | NAME | SALARY | DEPARTMENT\_ID |

The Java Persistence API provides Java developers with an object/relational mapping facility for managing relational data in Java applications to the underlying database.

1. Annotate the appropriate code segments in each class to mark them as persistent entities and explain the purpose of each annotation. Include any extra imports that may be needed in order to use these annotations.

( 8 Marks)

1. Annotate and explain the appropriate mapping for these two entities bearing in mind that a department is made up of a collection of employees and an employee works for just one department.

(13 Marks)

1. Write the Java code for a test class that creates one department and one employee object and persists these objects to the database.

(12 Marks)

**Question 11 (33 Marks)**

1. An implementation is required for a dictionary look up system in Java that will store a collection of words and their corresponding definitions.
   1. Identity and explain the most efficient collection for the implementation of the dictionary? (4 Marks)
   2. Write the Java code to create this collection, add an entry to the collection, ask the user to enter a word, look up the definition of the word and print the definition to the console.

(10 Marks)

1. Consider the following class TestPair which tests a generic class called Pair. Using the test class as a guide, write the Java code for the generic Pair class so that it produces the output shown below.

package qst;

public class TestPair {

public static void main(String[] args) {

System.out.println(Pair.getNumberOfPairs());

Pair<Integer> pairOne = new Pair<>(3, 4);

System.out.println(Pair.getNumberOfPairs());

Pair<String> pairTwo = new Pair<>("Hello", "Goodbye");

System.out.println(Pair.getNumberOfPairs());

System.out.println(pairOne.toString());

System.out.println(pairTwo.toString());

pairOne.setX(8);

System.out.println(pairOne.getX());

}

}

Sample Output

0

1

2

(3, 4)

(Hello, Goodbye)

8

(8 Marks)

1. The following test class creates a collection of Dog objects and stores them in an array list. The Collections.sort() method is used twice to sort the array list by name and then by age as shown in the sample output.

package qst;

import java.util.\*;

public class TestDog {

public static void main(String args[]) {

List<Dog> list = new ArrayList<>();

list.add(new Dog("Shep", 3));

list.add(new Dog("Freckle", 2));

list.add(new Dog("Trixie", 10));

list.add(new Dog("Tommy", 4));

list.add(new Dog("Tammy", 1));

Collections.sort(list);

for (Dog a : list) {

System.out.print(a.getDogName() + ", ");

}

Collections.sort(list, new Dog());

System.out.println(" ");

//printing the sorted list of ages

for (Dog a : list) {

System.out.print(a.getDogName() + " : "

+ a.getDogAge() + ", ");

}

}

}

Sample Output

Freckle, Shep, Tammy, Tommy, Trixie,

Tammy : 1, Freckle : 2, Shep : 3, Tommy : 4, Trixie : 10

Write the relevant code for the Dog class that is needed in order for the sort() method to work. Assume the member variables, getter, setter and toString methods are already written.

(11 Marks)

**Question 12 (33 Marks)**

1. Write a method interleave, which takes two LinkedList objects and returns a new LinkedList where the values from the old lists are interleaved into the new list: that is, the new list starts with the first element from list1, then first element from list2, second element from list1, second element from list2, etc. If one of the lists is longer than the other, it puts all the leftover elements on the end of the resulting list. The original lists are unchanged by this method.

Example:

list1 (1 3 5 7 9 11 13)

list2 (8 6 4 2)

newlist (1 8 3 6 5 4 7 2 9 11 13)

(15 Marks)

1. Show the results of inserting the following six keys that have the following hash values into an empty hash table that starts out empty, has 10 buckets (indices 0 through 9), and uses chaining. Do the insertions in the order the keys are shown (Tony-Janice).

|  |  |
| --- | --- |
| Key | Hash Value |
| Tony | 3 |
| Christopher | 7 |
| Carmela | 3 |
| Silvio | 3 |
| Paul | 7 |
| Janice | 4 |

(4 Marks)

For each of the following lookups from the table above, after the insertions are done, give the sequence of keys that the target key would have to be compared with to do the following lookup

1. lookup Adriana; Adriana’s hash value is 7
2. lookup Sal; Sal’s hash value is 8

(4 Marks)

1. Write the Java code to simulate playing the lottery. You will need to generate 6 random lottery numbers between 1 and 47 (inclusive) and store them in a collection that does not allow duplicates. Use an iterator to display the selection.

(10 Marks)