Lab report - 1:

In Java, multiple inheritance means a class can get features from more than one parent. An abstract class does not support multiple inheritance. A class can extend only one abstract class. An interface supports multiple inheritance in Java. A class can implement more than one interface at the same time. Abstract classes can have both abstract and normal methods. Interfaces contain only abstract methods. We use an abstract class when classes are closely related. We use an interface when multiple inheritance is needed. Therefore, Inheritance are used to solve the multiple inheritance Problem in Java.

Java code example:

```
interface A {
        void show ();
    }
interface B {
        void Print ();
    }
class Test implement A, B {
     Public void show () {
    System. out. Println ("Inheritance A");
    }
   Public void Print () {
    System. Out. Println ("Interface B");
   }
```

Lab report-2:

Encapsulation is an OOP concept that hides data and protect them from direct access. It ensures data sequrity by making variable Private inside a class. Outside classes cannot change data directly. Data can be changed only using Public methods. These methods check the values before saving them. These prevents wrong or harmful data from entering the system Encapsulation also keeps data consistent and connect. If invalid data is given, the method rejects it. In bank system, this is very important for safety. Therefore, encapsulation ensures both data and sequrity.

Java code:

```
class_Account{
  Private string accountNumbers;
  Private double balance;
  Public void SetAccountNumber(string accNO){
  if (accNO!=null && !accNO.isempty()){
      accountNumber = accNO;
    }
    else {
      System.out.Println("Invalid account number");
    }
}Public void setinitialBalance(double ammount){
    if (ammount>0){
        balance = ammount;
    }
else{
      system.out.Println("Balance cannot be negative);
  }
}
```

Lab report-4:

JDBC stands for Java Database connectivity. It is used to connect a Java Program with a relational database. JDBC works as a bridge between Java application and database. The Java Program sends SQL queries using JDBC. JDBC driven receaves the request and talks to the database. The database Process the query and sends to the result back. JDBC driven receaves returns the result to the Java Program. JDBC allow data to be inserted, updated, deleted and read. It also hadles connection and errors management. Thus, JDBC manage smmoth communication between Java and Database. To execute a SELECT query, JDBC follows some fixed steps. First, a connection is created with the database. Then a statement Prepared statement

Object is created. After that, a SELECT SQL query is written. The query is executed using executequery() method. This method return a Resultset. try block is use to hadle database operation safely. Catch block handles SQL or routine errors. Finally block is used to close connection and resources.

Java code

```
import java.sql*;
class selectExample {
Public static void main (String [] args) {
   collection con = null;
try {
      con = DriverManager.get.connection (
"Jdbc:mysql://localhost:3306/testdb", "root",
"password");
 statement st = con.create statement ();
 Resultset rs = st.execute Query ("SELECT*
 Fnom student");
```

```
while (rs.next()) {
System.out.println (rs.getInt(1) + " " +rs.getString(2));
}
}
catch (Exception e ) {
System.out.println (" Errors occured ");
} finally {
try {
      if (con != null)
        con.close();
      }
   catch (Exception e) {
    System.out.println (" connection not closed ");
  }
}
}
}
```

Lab report 5:

In a Java EE application, a servlet works as a controller. ~~manages~~ The controller manages the flow between model and View. The model contains buisness data and logic. The servlet ~~contains~~ gets data from the model. Then it sends this data to the View. Jsp is used as the view to show data to the user. The servlet forward the request to Jsp using Request Dispatcher. Data is sent using request attributes. Jsp reads the data and displays it. Thus, the servlet controller ~~contro~~ controls the application flow.

Java code:

```
import java.io*;
import javax.servlet.*;
import javax.http.*;
Public class Helloservlet exclends Httpservelet {
Protected void doGet (HttpservletRequest req,
HttpservletResponse res)
```

```
throws servletException, IO Exception {
    string name = "student";
    req. setAttribute ("msg", name);
    RequestDispatcher sd = req. getRequestDispatcher
     "hello. jsp");
      sd. forward (req, res);
  }
}
```

Jsp code for view:

```
<html>
 <body>
   <h2> Hello, ${msg} </h>
 </body>
</html>
```

Lab report - 6 :

Prepared statement is used to execute SQL queries safely in JDBC. It improves performance because the query is precompiled by the database. The same query can be used many time with different values. Prepared statement is faster than statement for repated queries. It also improves security by preventing SQL injection attacks. Users input is treated as data, not as SQL code. statement directly executes SQL and is less secure. Prepared statement uses placeholders (?) for values. These values are set using setter methods. So, Prepared statement is better for the performance and security.

Java code (Insert using Prepared statement) :

```
import java.sql.* ;
class insertExample {
    public static void main (string [] args) {
try {
 connection con = DriverManager.getconnection (
"Jdbc:mysql://localhost:3306/testdb","root"
"password");
string sql = "Insert Into student (id, name) values
   (?,?)";
Prepared statement ps = con.preparedstatement(sql);
ps. setInt (1,1) ;
ps. setstring (2, "Rahim");
ps. executeupdate ();
System. out.println ("Record insented");
con. close ();
} catch (Exception e) {
   System. out.println ("Errors occured");
}
}
```

Lab report-7:

Resultset is an object in JDBC that stores data returned from a database query. It is mainly used with select queries, Resultset works like a table with rows and colums. The next() method moves the cursor to the next row. It returns true if data is available. The setstring() method is used to read string type data. Data is read column by column from Resultset Resultset helps java programs fetch databa records easily. Thus, it is very important for retrieving data in JDBC.

Java code (Resultset usage):

```
import java.sql*;
class ResultSet Example {
Public static void main (string [] args) {
try {
```

```
connection con = DriverManager.getConnection (
"Jdbc:mysql://localhost:3306/testdb", "root",
"password");
statement st = con.createstatement();
Resultset rs = st.executeQuery ("SELECT id, name
from student");
while (rs.next()){
int id = rs.getInt("int");
string name = rs.getString("Name");
System.out.println(" id + " " + name);
}
con.close();
} catch (Exception e) {
System.out.println ("Errors occured");
}
}
}
```

Lab report - 08

JPA manage the mapping between Java objects and relational database table through a process called object - Relational mapping. Here's how JPA handle the mapping:

Entity defination:

- Java classes are marked as JPA entities using the @ Entity annotation. This indicates that the class represent a table in the database

Field to column mapping:

By default, JPA maps fields in an entity class to columns in the corresponding database table with the same time.

Relationship mapping:

JPA Provides annotations to manage relation-ships between entities, which corresponds to relationships between tables in the database

Java code :

```
import jakarta.persistance.Entity;
import jakarta.Persistence.Id;
import jakarta.Persistence.GeneratedValue;
@Entity
Public class student {
@Id
 Private int id;
 Private string name;
 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}

}
```

Lab report - 9 :

How Prepared statement improves performance & security over statement in JDBC is given below :

Security :

- Prepared statement automatically escapes user input making it safe from SQL injection attacks.
- With statement, user input is directly concatenated into the SQL string.

Performance :

- Prepared statement allows the database to Precompile the SQL statement and reuse it with diff parameters.
- This reduces parsing time and improves efficiency

Java code :

Inserting a Record using Prepared statement

```
import java.sql.*;
public class Insert Example {
public static void main (string [] args) {
string urls "jdbc:mysql:// localhost:3306/studentdb"
String username = "Gupi";
string pass = "root-1236";
string insertSQL = "insert into students (name.age)
        values (?,?)";
try (connection conn = DriverManager.getConnection (
  url, username, password);
  pstmt.setString(1, "Rafi");
  pstmt.setInt (2, 22);
 int rowInserted = pstmt.executedUpdate();
 if (rowInserted > 0)
  Sout ("Successful insertion");
} catch (SQLException e) {
} e.printstackTrace();
 }
 }
```