Q] Design a java application to demonstrate the use of Core Java fundamentals:

a) Constructor b) Inheritance c) Polymorphism

a] Constructor:

Source Code:

```
public class ass1 {

	String firstname,lastname;

	int age;

	public ass1() {   //default

		firstname="Shreyash";

		lastname="Waghmode";

		age=19;

	}

	public ass1(String firstname, String lastname, int age)//parameterised

	{

		this.firstname=firstname;

		this.lastname=lastname;

		this.age=age;

	}


	public static void main(String[] args) {

		ass1 obj1 = new ass1();
```

```
                    ass1 obj2 = new ass1("Raj", "Kumar", 25);

                    System.out.println(obj1.firstname);

                    System.out.println(obj1.lastname);

                    System.out.println(obj1.age);

                    System.out.println(obj2.firstname);

                    System.out.println(obj2.lastname);

                    System.out.println(obj2.age);

          }


}
```

b] Inheritance:

Source Code:

```
class Base {

          String firstName = "Shreyash";

}

class Child1 extends Base{

          String lastName = "Waghmode";

}

class Child2 extends Child1{

          int rollNo = 48;

}

class Child3 extends Base{
        String role = "Student";
}

public class Inheritance {
        public static void main(String a[])
        {
                Child2 obj = new Child2();
                Child3 obj2 = new Child3();
                System.out.println(obj.firstName);//Single
                System.out.println(obj.lastName);//Single
                System.out.println(obj.rollNo);//Multilevel
                System.out.println(obj2.firstName +" is a " + obj2.role);//Hierarchical
        }
}
```

C] Polymorphism:

Source Code:

```
class Animal {
    public void makeSound() {
        System.out.println("Some generic sound");
    }
```

```
}

class Dog extends Animal {
    @Override
    public void makeSound() {
        System.out.println("Bark! Bark!");
    }

    // Overloaded method for demonstrating compile-time polymorphism
    public void makeSound(int times) {
        for (int i = 0; i < times; i++) {
            System.out.println("Bark!");
        }
    }
}

class Cat extends Animal {
    @Override
    public void makeSound() {
        System.out.println("Meow! Meow!");
    }

    // Overloaded method for demonstrating compile-time polymorphism
    public void makeSound(String emotion) {
        System.out.println("Purr... Feeling " + emotion);
    }
}

public class AnimalDemo {
    public static void main(String[] args) {
        Animal myDog = new Dog();
        Animal myCat = new Cat();

        // Calls the overridden makeSound method based on the actual object type
        myDog.makeSound();  // Output: Bark! Bark!
        myCat.makeSound();  // Output: Meow! Meow!

        // Calls the overloaded makeSound method based on the actual object type
        ((Dog) myDog).makeSound(3);  // Output: Bark! Bark! Bark!
        ((Cat) myCat).makeSound("happy");  // Output: Purr... Feeling happy
    }
}
```

Q] Design a java application to demonstrate the use of Core Java fundamentals:

a) Abstraction b) Encapsulation c) Interface

a) Abstraction

Source Code:

```
// Java Program to implement

// Abstract Keywords


// Parent Class

abstract class gfg {

        abstract void printInfo();

}


// Child Class

class employee extends gfg {

        void printInfo()

        {

                String name = "Yashwant";

                int age = 19;

                float salary = 55552.2F;
                System.out.println(name);

                System.out.println(age);

                System.out.println(salary);

        }

}


// driver Class

class base {

        // main function

        public static void main(String args[])

        {

                // object created

                gfg s = new employee();

                s.printInfo();

        }

}
```

b) Encapsulation

Source Code:

```
// Java Program to implement

// Java Encapsulation
// Person Class

class Person {

        // Encapsulating the name and age

        // only approachable and used using

        // methods defined

        private String name;

        private int age;


        public String getName() { return name; }


        public void setName(String name) { this.name = name; }


        public int getAge() { return age; }


        public void setAge(int age) { this.age = age; }

}


// Driver Class

public class Main {

        // main function

        public static void main(String[] args)

        {

                // person object created

                Person person = new Person();
```

```
			person.setName("Yashwant");

			person.setAge(19);


			// Using methods to get the values from the

			// variables

			System.out.println("Name: " + person.getName());

			System.out.println("Age: " + person.getAge());

		}

}
```

c) Interface

Source Code:

```
interface Animal {

  public void animalSound(); // interface method (does not have a body)

  public void sleep(); // interface method (does not have a body)

}


class Cat implements Animal {

  public void animalSound() {

    System.out.println("The Cat says: meow!");

  }
  public void sleep() {
    System.out.println("Zzz");
  }
}

class InterfaceExample {
  public static void main(String[] args) {
    Cat myCat = new Cat();
    myCat.animalSound();
    myCat.sleep();
  }
}
```

Implement a program to demonstrate the concept of Exception Handling in Java. (try catch, multiple catch, finally)

Source Code:

```
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.Scanner;
```

```
public class Reservation {

    private int roomNumber;

    private String guestName;

    private Date reservationDate;


    public Reservation() {

        try {

            Scanner scanner = new Scanner(System.in);

            System.out.print("Enter room number: ");

            int roomNumberInput = scanner.nextInt();

            if (roomNumberInput <= 0) {

                throw new Exception("Room number must be positive.");

            }

            this.roomNumber = roomNumberInput;


            scanner.nextLine(); // Consume newline character

            System.out.print("Enter guest name: ");

            String guestNameInput = scanner.nextLine();

            if (guestNameInput == null || guestNameInput.isEmpty()) {

                throw new Exception("Guest name cannot be null or empty.");

            }

            this.guestName = guestNameInput;
            System.out.print("Enter reservation date (yyyy-mm-dd): ");
            String dateInput = scanner.nextLine();
            if (dateInput == null || dateInput.isEmpty()) {
                throw new Exception("Reservation date cannot be null or empty.");
            }
            SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd");
            this.reservationDate = dateFormat.parse(dateInput);
        } catch (Exception e) {
            System.err.println("Error initializing reservation: " + e.getMessage());
        }
    }

    public void displayReservationDetails() {
        try {
            System.out.println("Room Number: " + roomNumber);
            System.out.println("Guest Name: " + guestName);
            System.out.println("Reservation Date: " + reservationDate);
        } catch (NullPointerException e) {
            System.err.println("Error displaying reservation details: " + e.getMessage());
        }
    }

    public static void main(String[] args) {
        // Test case
```

```
        try {
            Reservation reservation = new Reservation();
            reservation.displayReservationDetails();
        } catch (Exception e) {
            System.err.println("Exception occurred: " + e.getMessage());
        }
    }
}
```

Q] Develop a Java Application using Multithíeading

Source Code

```
public class main {
public static void main(String[] args) {
int maxCount = 30; // Define the maximum number to count int numThreads
= 3; // Define the number of threads to use

// Create and start threads
CounterThread[] threads = new CounterThread[numThreads]; int
countPerThread = maxCount / numThreads;

int start = 1;

int end = countPerThread;

for (int i = 0; i < numThreads; i++) { threads[i] = new CounterThread(start,
end); threads[i].start();

start = end + 1;

end += countPerThread;

if (i == numThreads - 2) {

end = maxCount; // Adjust the last thread's end value

}

}


// Wait for all threads to finish

for (CounterThread thread : threads) { try {

thread.join();

} catch (InterruptedException e) { e.printStackTrace();

}

}


System.out.println("Counting completed.");

}
```

```
}

class CounterThread extends Thread { private int start;

private int end;


public CounterThread(int start, int end) { this.start = start;

this.end = end;

}


@Override

public void run() {

for (int i = start; i <= end; i++) {
System.out.println(Thread.currentThread().getName() + ": " +i);

}

}

}
```

Develop an application to demonstrate the use of Java Collections Framework: ArrayList.

Source Code:

```
import java.util.*;

class ArrayListDemo {

 public static void main(String args[]) {



 ArrayList<String> al = new ArrayList<String>();

 System.out.println("Initial size of ArrayList: " +

 al.size());



 al.add("1");

 al.add("2");

 al.add("3");

 al.add("4");

 al.add("5");

 al.add("6");



 System.out.println("Size of ArrayList before additions: " +

 al.size());



 System.out.println("Contents of ArrayList: " + al);

 al.add(1, "12");



 System.out.println("Size of ArrayList after additions: " +

 al.size());
 System.out.println("Contents of ArrayList: " + al);

 al.remove("3");
 al.remove(2);

 System.out.println("Size of ArrayList after deletions: " +
 al.size());

 System.out.println("Contents of ArrayList: " + al);
 }
}
```

JDBC Connectivity: Develop an application to demonstrate JDBC connectivity

Source Code:

```
import java.sql.*;

public class FirstExample {
  static final String DB_URL = "jdbc:mysql://localhost/test";

  static final String USER = "root";

  static final String PASS = "Yjn@270304";

  static final String QUERY1 = "select * from jdbc_test";

  static final String QUERY2 = "insert into jdbc_test values
(3,\"Mantu\",\"Savedi\",\"Ahmednagar\",\"MH\",\"005\")";



  public static void main(String[] args) {

    // Open a connection

    try(Connection con = DriverManager.getConnection(DB_URL, USER, PASS);

      Statement stmt = con.createStatement();

      ResultSet rs = stmt.executeQuery(QUERY1);) {

      // Extract data from result set

      while (rs.next()) {

        // Retrieve by column name

        System.out.print("ID: " + rs.getInt("id"));

        System.out.print(", Name: " + rs.getString("name"));

        System.out.print(", Street: " + rs.getString("street"));

        System.out.println(", City: " + rs.getString("city"));

        System.out.println(", State: " + rs.getString("state"));

        System.out.println(", ZIP: " + rs.getString("zip"));

      }

      int rowsAffected = stmt.executeUpdate(QUERY2);
    } catch (SQLException e) {
        e.printStackTrace();
    }
  }
}
```

Servlet

Source Code:

```
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;

@WebServlet("/hello")
public class Hello extends HttpServlet {
    private static final long serialVersionUID = 1L;


    public Hello() {

        super();

    }


    protected void doGet(HttpServletRequest request,
HttpServletResponse response) throws ServletException,
IOException {

        response.setContentType("text/html");

        PrintWriter out = response.getWriter();


        out.println("<html>");

        out.println("<head>");

        out.println("<title>Hello Servlet</title>");

        out.println("<style>");

        out.println("body { font-family: Arial, sans-serif; background-
color: #f0f0f0; margin: 0; padding: 0; }");

        out.println(".container { width: 80%; margin: 0 auto; padding:
20px; background-color: #fff; box-shadow: 0 0 10px rgba(0, 0, 0,
0.1); border-radius: 5px; }");

        out.println(".heading { color: #008080; }");
        out.println(".paragraph { color: #333; font-size: 16px; font-style:
italic; margin-bottom: 10px; }");

        out.println(".list-item { color: #555; }");

        out.println(".deployment-steps { margin-top: 20px; }");

        out.println("</style>");

        out.println("</head>");

        out.println("<body>");



        // Container

        out.println("<div class='container'>");



        // Heading

        out.println("<h1 class='heading'>Hello Servlet</h1>");



        // Introduction

        out.println("<h2 class='heading'>Introduction</h2>");

        out.println("<p class='paragraph'>Servlets are Java classes used
in web development to extend the functionality of web servers. They
are a fundamental component of Java-based web applications,
providing a robust and platform-independent means of handling
HTTP requests and generating dynamic web content.</p>");



        // Getting Started
        out.println("<h2 class='heading'>Getting Started</h2>");

        out.println("<p class='paragraph'>To begin using this servlet,
follow these steps:</p>");

        out.println("<ol class='list-item'>");

        out.println("<li>Create a Java servlet project in your preferred
IDE.</li>");

        out.println("<li>Copy the servlet code into your project's servlet
class.</li>");

        out.println("<li>Deploy the servlet to a servlet container (e.g.,
Apache Tomcat).</li>");

        out.println("</ol>");


        // Usage Examples

        out.println("<h2 class='heading'>Usage Examples</h2>");

        out.println("<p class='paragraph'>Here are some common use
cases for servlets:</p>");

        out.println("<ul class='list-item'>");

        out.println("<li>Creating dynamic web pages.</li>");

        out.println("<li>Handling form submissions.</li>");

        out.println("<li>Interacting with databases.</li>");

        out.println("</ul>");


        // Deployment Instructions
        out.println("<h2 class='heading'>Deployment
Instructions</h2>");

        out.println("<p class='paragraph'>To deploy this servlet, follow
these steps:</p>");

        out.println("<ol class='deployment-steps'>");

        out.println("<li class='list-item'>Compile the servlet code into a
.class file.</li>");

        out.println("<li class='list-item'>Create a WAR (Web Archive)
file containing the servlet class file, along with any other necessary
resources (such as HTML, CSS, or image files).</li>");

        out.println("<li class='list-item'>Deploy the WAR file to a
servlet container (such as Apache Tomcat) by placing it in the
'webapps' directory of the servlet container installation.</li>");

        out.println("<li class='list-item'>Start or restart the servlet
container to deploy the servlet. You can then access the servlet using
its URL, typically in the format
http://localhost:8080/application_name/servlet_mapping.</li>");

        out.println("<li class='list-item'>Configure any additional
settings or resources required by your servlet, such as database
connections or external dependencies.</li>");

        out.println("<li class='list-item'>Test the deployed servlet to
ensure it functions correctly in the production environment.</li>");

        out.println("</ol>");



        // Resources
        out.println("<h2 class='heading'>Resources</h2>");

        out.println("<p class='paragraph'>For further information on
servlets and Java web development, refer to the following
resources:</p>");

        out.println("<ul class='list-item'>");

        out.println("<li><a
href='https://www.oracle.com/java/technologies/java-servlet-
technologies.html'>Oracle Java Servlet Technologies</a></li>");

        out.println("<li><a
href='https://docs.oracle.com/javaee/6/tutorial/doc/bnafd.html'>Java
EE 6 Tutorial - Servlets</a></li>");

        out.println("<li><a
href='https://www.w3schools.com/java/java_servlets.asp'>W3Schools
Java Servlets Tutorial</a></li>");

        out.println("</ul>");



        // Best Practices

        out.println("<h2 class='heading'>Best Practices</h2>");

        out.println("<p class='paragraph'>Follow these best practices
when developing servlets:</p>");

        out.println("<ul class='list-item'>");

        out.println("<li>Keep servlets focused on specific tasks to
maintain simplicity and clarity.</li>");

        out.println("<li>Use the HttpServlet class for handling HTTP-
specific methods (e.g., doGet, doPost).</li>");
```

```
        out.println("<li>Handle exceptions gracefully and provide
meaningful error messages.</li>");

        out.println("</ul>");



        // Troubleshooting

        out.println("<h2 class='heading'>Troubleshooting</h2>");

        out.println("<p class='paragraph'>If you encounter issues with
your servlet, consider the following troubleshooting steps:</p>");

        out.println("<ul class='list-item'>");

        out.println("<li>Check server logs for any error messages or
stack traces.</li>");

        out.println("<li>Ensure servlet mappings are configured
correctly in web.xml or through annotations.</li>");

        out.println("<li>Verify that all required dependencies and
resources are properly configured.</li>");

        out.println("</ul>");



        // Community Support

        out.println("<h2 class='heading'>Community Support</h2>");

        out.println("<p class='paragraph'>Join online communities and
forums for servlet development to seek help, share knowledge, and
stay updated with the latest trends:</p>");

        out.println("<ul class='list-item'>");
        out.println("<li><a
href='https://stackoverflow.com/questions/tagged/servlets'>Stack
Overflow - Servlets Tag</a></li>");

        out.println("<li><a
href='https://www.reddit.com/r/javaservlets/'>Reddit - Java Servlets
Community</a></li>");

        out.println("</ul>");




        // Conclusion

        out.println("<h2 class='heading'>Conclusion</h2>");

        out.println("<p class='paragraph'>Servlets play a crucial role in
Java web development by facilitating the creation of dynamic,
scalable, and interactive web applications.</p>");



        // Closing Container

        out.println("</div>");



        out.println("</body>");

        out.println("</html>");

    }
    protected void doPost(HttpServletRequest request,
HttpServletResponse response) throws ServletException,
IOException {



    }

}
```