PRACTICAL-1.

WAP to implement Date and Time

Q-1 LocalDate Example:

```
import java.time.LocalDate;
public class Main{
  public static void main(String[] args){
    LocalDate myObj = LocalDate.now();
    System.out.println(myObj);
  }
}
```

Output 2025-01-04

Q-2 LocalTime Example

```
import java.time.LocalTime;
public class Main{
  public static void main(String[] args){
    LocalTime myObj = LocalTime.now();
    System.out.println(myObj);
  }
}
```

Output 03:27:37.752993978

Q-3 LocalDateTime example

```
import java.time.LocalDateTime;
public class Main {
    public static void main(String[] args) {
        LocalDateTime myObj = LocalDateTime.now();
        System.out.println(myObj);
    }
}
```

Output 2025-01-14T03:29:39.045418242

Q-4 DateTimeFormatter example

```
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
public class Main {
    public static void main(String[] args) {
        LocalDateTime myDateObj = LocalDateTime.now();
        System.out.println("Before formatting: "
                + myDateObj);

        DateTimeFormatter myFormatObj
            = DateTimeFormatter.ofPattern("dd-MM
                yyyy HH:mm:ss");

        String formattedDate = myDateObj.format
                (myFormatObj);
        System.out.println("After formatting: " + formattedDate
```

Output:

Before formatting: 2025-01-14T03:34:19.673514
After formatting: 14-01-2025 03:34:19

Q-5 Using now() to get current date and using plusDays and minusdays.

```
import java.time.LocalDate;
public class LocalDate.Example1 {
    public static void main(String[] args) {
        LocalDate date = LocalDate.now();
        LocalDate yesterday = date.minusDays(1)
        LocalDate tomorrow = yesterday.plusDays(2
        System.out.println("Today date: "+ date);
        System.out.println("Yesterday date:"+
                yesterday);
        System.out.println("Tomorrow date:"
                + tomorrow);
    }
}
```

Output:

Today date: 2025-01-14
Yesterday date: 2025-01-13
Tomorrow date: 2025-01-15

Q.6 Using LeapYear to check if the given date is a leap year or not.

```
import java.time.LocalDate;
public class LocalDateEx2 {
    public static void main(String[] args) {
        LocalDate date1 = LocalDate.of(2017, 1, 13);
        System.out.println(date1.isLeapYear());
        LocalDate date2 = LocalDate.of(2016, 9, 23);
        System.out.println(date2.isLeapYear());
    }
}
```

Output
False
True

Q.7 Program → Java Program to illustrate MonthDay class by importing MonthDay class Form Java.time

```
import Java.time.MonthDay;
Public class Main {
    public Static void main(String[] args)
    {
        MonthDay m = MonthDay.Parse("--03-14");
```

```
    System.out.println(m.getMonth());
  }
}
```

Output: March

Q. Java Program to implement OffsetTime class via now() method

```
import java.time.OffsetTime;
import java.time.temporal.ChronoField;

Public class Main {

  public static void main(String[] args)
  {
    OffsetTime o = OffsetTime.now();
    int m = o.get(chronofield.MINUTE_OF_DAY);
    System.out.println(m);
    int h = o.get(Chronofield.HOUR-OF-DAY);
    System.out.println(h);
    int s = o.get(Chronofield.SECOND_OF_DAY);
    System.out.println(s);
  }
```

Output:

```
3
231
13861
```

Q. Java Program to implement OffsetTime class via getHour() method.

```
import java.time.OffsetTime;
Public class Main{
  Public static void main(String[] args)
{
    OffsetTime o = offsetTime.now();
    int h = o.getHour();
    System.out.println(h + "hours");
}
}
```

Output 3 hours.

Q. Java Program to demonstrate the format() method

```
import java.time.OffsetDateTime.
import java.time.format.DateTimeFormatter;
Public class Main{
  public static void main(String[] args){
    OffsetDateTime d1 = OffsetDateTime.Parse
                        ("2018-12-12T13:30:30+05
                        :00");
    System.out.println("Date1:"+date1);
```

```
DateTimeFormatter formatter =
        DateTimeFormatter.ISO_TIME;
System.out.println(formatter.format(date1
}
}
```

Output: Date1: 2018-12-12T13:30:30+05:0
13:30:30+05:00

Q-1 Java program for creating instance of Clock

```
import java.time.Clock;
public class GFG {
    public static void main(String[] args) {
        Clock clock = Clock.systemUTC();
        System.out.println("UTC time = " +
                clock.instance());
    }
}
```

Q-2 Java program for creating instance of clock

```
import java.time.Clock;
public class GFG {
    public static void main(String[] args) {
```

```
Clock clock = Clock.systemDefaultZone();
System.out.println(clock);
System.out.println("Time Zone: " +
                   clock.getZone());
}
}
```

Output : 2025-01-14T08:57:55.828351741+2

Q-13 Creating a clock instance using SystemdefaultZone() method of Clock class

```
import java.time.clock;
public class Main(String[] args) {
    clock c = clock.SystemDefaultZone().
    System.out.println(c);
    System.out.println("Time Zone: "
                       + c.getZone(c));
}
}
```

Output System.Clock[Asia/Calcutta]
-TimeZone: Asia/Calcutta

Q-14 Wap to implement yearMonth class

```
import java.time.yearMonth;
public class Main{
    public static void main(String[] args){
        yearMonth ym = yearMonth.now();
    }
```

```
System.out.println("current yearmonth"
                    + ym);

yearMonth ym1 = yearMonth.of(2023, 9);
System.out.println("Specific year Month"
                    + ym1);
    }
}
```

Output: Current yearmonth : 02, 2025
Specific year Month : 2023, 9

Q-16 WAP to implement Period & localdate class.

```
import java.time.LocalDate;
import java.time.period;
public class Main {

  public static void main(String[] args) {

    LocalDate L = LocalDate.of(2023, 1, 1);
    LocalDate L1 = LocalDate.of(2023, 2, 2);
    Period P = Period.between(L, L1);
    System.out.println("Period :" + P);
    System.out.println("years :" + P.getYear());
    System.out.println("Month :" + P.getMonth());
    System.out.println("days :" + p.getDays());
  }
}
```

Output:
Period : P2Y1M1D
years : 2
Month : 1
days : 1

Q-17 WAP to implement Duration and Instant class

```
import java.time.Duration;
import java.time.Instant;

class main {
    public static void main (String[] args) {

        Instant st = Instant.now();
        Instant ed = st.plusSecond.(3600).
        Duration dur = Duration.between(st,ed)
        System.out.println("Seconds:" +
                    dur.getSeconds());
        System.out.println("Minutes" +
                    dur.toMinutes());

        Duration dur1 = dur.plusminutes(30);

        System.out.println("Duration after
            adding 30 minutes:" + dur1);
    }
}
```

Output :

Seconds : 3600

Minutes : 60

Durations after adding 30 minutes : PT1H30M

8/2/25