1. Write a program to display the current date and add 1 week, 1 month,1 year and 10 years to the current date and print.

Program:

```
import java.time.LocalDate;
import java.time.temporal.ChronoUnit;

public class ChronoUnitTest {
   public static void main(String args[]) {
      // Get the current date
      LocalDate today = LocalDate.now();
      System.out.println("Current date: " + today);
      // add 1 week to the current date
      LocalDate nextWeek = today.plus(1, ChronoUnit.WEEKS);
      System.out.println("Next week: " + nextWeek);
      // add 1 month to the current date
      LocalDate nextMonth = today.plus(1, ChronoUnit.MONTHS);
      System.out.println("Next month: " + nextMonth);
      // add 1 year to the current date
      LocalDate nextYear = today.plus(1, ChronoUnit.YEARS);
      System.out.println("Next year: " + nextYear);
      // add 10 years to the current date
      LocalDate nextDecade = today.plus(1, ChronoUnit.DECADES);
      System.out.println("Date after ten year: " + nextDecade);
   }
```

```
}
/*
 * Current date: 2024-03-29
 * Next week: 2024-04-05
 * Next month: 2024-04-29
 * Next year: 2025-03-29
 * Date after ten year: 2034-03-29
 */
```

2. Write a program to print dates in different formats.

Program:

```
import java.text.DateFormat;
import java.util.Date;

public class DateFormatExample2 {
   public static void main(String[] args) {
      Date currentDate = new Date();
      System.out.println("Current Date: " + currentDate);

      String dateToStr = DateFormat.getInstance().format(currentDate);
      System.out.println("Date Format using getInstance(): " + dateToStr);

      dateToStr = DateFormat.getDateInstance().format(currentDate);
      System.out.println("Date Format using getDateInstance(): " + dateToStr);
```

```
        dateToStr = DateFormat.getTimeInstance().format(currentDate);
        System.out.println("Date Format using getTimeInstance(): " + dateToStr);

        dateToStr = DateFormat.getDateTimeInstance().format(currentDate);
        System.out.println("Date Format using getDateTimeInstance(): " + dateToStr);

        dateToStr = DateFormat.getTimeInstance(DateFormat.SHORT).format(currentDate);
        System.out.println("Date Format using getTimeInstance(DateFormat.SHORT): " + dateToStr);

        dateToStr = DateFormat.getTimeInstance(DateFormat.MEDIUM).format(currentDate);
        System.out.println("Date Format using getTimeInstance(DateFormat.MEDIUM): " + dateToStr);

        dateToStr = DateFormat.getTimeInstance(DateFormat.LONG).format(currentDate);
        System.out.println("Date Format using getTimeInstance(DateFormat.LONG): " + dateToStr);

        dateToStr = DateFormat.getDateTimeInstance(DateFormat.LONG,
DateFormat.SHORT).format(currentDate);
        System.out.println("Date Format using
getDateTimeInstance(DateFormat.LONG,DateFormat.SHORT): " + dateToStr);

    }
}

/*
 * Current Date: Thu Mar 07 10:12:34 IST 2024
```

```
 * Date Format using getInstance(): 07/03/24, 10:12?am
 * Date Format using getDateInstance(): 07-Mar-2024
 * Date Format using getTimeInstance(): 10:12:34?am
 * Date Format using getDateTimeInstance(): 07-Mar-2024, 10:12:34?am
 * Date Format using getTimeInstance(DateFormat.SHORT): 10:12?am
 * Date Format using getTimeInstance(DateFormat.MEDIUM): 10:12:34?am
 * Date Format using getTimeInstance(DateFormat.LONG): 10:12:34?am IST
 * Date Format using getDateTimeInstance(DateFormat.LONG,DateFormat.SHORT): 7
 * March 2024, 10:12?am
 */
```

3. Write a program to print local date and check whether the year is leap year or not.

```
import java.time.LocalDate;

public class LocalDateExample1 {
  public static void main(String[] args) {
    LocalDate date = LocalDate.now();
    LocalDate yesterday = date.minusDays(1);
    LocalDate tomorrow = yesterday.plusDays(2);
    boolean var = date.isLeapYear();
    System.out.println("Yesterday date: " + yesterday);
    System.out.println("Tomorrow date: " + tomorrow);
    System.out.println("The Year is Leap?" + var);
  }
}
```

```
/*
 * Yesterday date: 2024-03-31
 * Tomorrow date: 2024-04-02
 * The Year is Leap?true
 */
```

4. Write a program to get the period between two dates and duration between two times.

```
import java.time.temporal.ChronoUnit;
import java.time.LocalDate;
import java.time.LocalTime;
import java.time.Duration;
import java.time.Period;

public class PeriodDurationTest {
   public static void main(String args[]) {
      // Get the current date
      LocalDate date1 = LocalDate.now();
      System.out.println("Current date: " + date1);
      // add 1 month to the current date
      LocalDate date2 = date1.plus(1, ChronoUnit.MONTHS);
      System.out.println("Next month: " + date2);
      Period period = Period.between(date2, date1);
      System.out.println("Period: " + period);
```

```
        LocalTime time1 = LocalTime.now();
        System.out.println("Current Time: " + time1);
        Duration twoHours = Duration.ofHours(2);
        LocalTime time2 = time1.plus(twoHours);
        System.out.println(" Time2: " + time2);
        Duration duration = Duration.between(time1, time2);
        System.out.println("Duration: " + duration);
    }
}
/*
 * Current date: 2024-02-21
 * Next month: 2024-03-21
 * Period: P-1M ("P1M" is a one-month duration)
 * Current Time: 16:19:34.266723900
 * Time2: 21:19:34.266723900
 * Duration: PT2H (Period of Time)
 */
```

5. Write a program to print a zone with date ,zone id and current zone.

```
import java.time.ZonedDateTime;
import java.time.ZoneId;

public class ZoneTest2 {
    public static void main(String args[]) {
```

```
        // Get the current date and time
        ZonedDateTime date1 = ZonedDateTime.parse("2007-12-03T10:15:30+05:30[Asia/Karachi]");
        System.out.println("date1: " + date1);
        ZoneId id = ZoneId.of("Europe/Paris");
        System.out.println("ZoneId: " + id);
        ZoneId currentZone = ZoneId.systemDefault();
        System.out.println("CurrentZone: " + currentZone);
    }
}
/*
 * date1: 2007-12-03T09:45:30+05:00[Asia/Karachi]
 * ZoneId: Europe/Paris
 * CurrentZone: Asia/Calcutta
 * PS C:\Users\nikit>
 */
```

6. Write a program that converts various strings into date objects.

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

public class StringToDateExample2 {
    public static void main(String[] args) throws Exception {
        String sDate1 = "31/12/1998";
        String sDate2 = "31-Dec-1998";
```

```
        String sDate3 = "12 31, 1998";
        String sDate4 = "Thu, Dec 31 1998";
        String sDate5 = "Thu, Dec 31 1998 23:37:50";
        String sDate6 = "31-Dec-1998 23:37:50";
        SimpleDateFormat formatter1 = new SimpleDateFormat("dd/MM/yyyy");
        SimpleDateFormat formatter2 = new SimpleDateFormat("dd-MMM-yyyy");
        SimpleDateFormat formatter3 = new SimpleDateFormat("MM dd, yyyy");
        SimpleDateFormat formatter4 = new SimpleDateFormat("E, MMM dd yyyy");// day of the week ( E )
        SimpleDateFormat formatter5 = new SimpleDateFormat("E, MMM dd yyyy HH:mm:ss");
        SimpleDateFormat formatter6 = new SimpleDateFormat("dd-MMM-yyyy HH:mm:ss");
        Date date1 = formatter1.parse(sDate1);
        Date date2 = formatter2.parse(sDate2);
        Date date3 = formatter3.parse(sDate3);
        Date date4 = formatter4.parse(sDate4);
        Date date5 = formatter5.parse(sDate5);
        Date date6 = formatter6.parse(sDate6);
        System.out.println(sDate1 + "\t" + date1);
        System.out.println(sDate2 + "\t" + date2);
        System.out.println(sDate3 + "\t" + date3);
        System.out.println(sDate4 + "\t" + date4);
        System.out.println(sDate5 + "\t" + date5);
        System.out.println(sDate6 + "\t" + date6);
    }
}
/*
```

```
* 31/12/1998 Thu Dec 31 00:00:00 IST 1998
* 31-Dec-1998 Thu Dec 31 00:00:00 IST 1998
* 12 31, 1998 Thu Dec 31 00:00:00 IST 1998
* Thu, Dec 31 1998 Thu Dec 31 00:00:00 IST 1998
* Thu, Dec 31 1998 23:37:50 Thu Dec 31 23:37:50 IST 1998
* 31-Dec-1998 23:37:50 Thu Dec 31 23:37:50 IST 1998
*/
```

7. Write a program to convert number from binary to decimal.

```
public class BinaryToDecimalExample3 {
   public static int getDecimal(int binary) {
      int decimal = 0;
      int n = 0;
      while (true) {
         if (binary == 0) {
            break;
         } else {
            int temp = binary % 10;
            decimal += temp * Math.pow(2, n);
            binary = binary / 10;
            n++;
         }
      }
      return decimal;
```

```
    }

    public static void main(String args[]) {
        System.out.println("Decimal of 1010 is: " + getDecimal(1010));
        System.out.println("Decimal of 10101 is: " + getDecimal(10101));
        System.out.println("Decimal of 11111 is: " + getDecimal(11111));
    }
}
/*
 * Decimal of 1010 is: 10
 * Decimal of 10101 is: 21
 * Decimal of 11111 is: 31
 */
```

8. Write a program to convert Hexadecimal number to Decimal number.

```
public class HexaToDecimal1 {
    public static void main(String arg[]) {
        String st = "18";
        int bn = Integer.parseInt(st, 16);
        System.out.println(bn);
        System.out.println(Integer.parseInt("f", 16));
        System.out.println(Integer.parseInt("125", 16));

    }
```

}

9. Write a program to encode and decode JSON object in PHP.

Encode:

```
<!DOCTYPE html>
<html>
<body>
<?php
$arr2 = array('firstName' => 'Rahul', 'lastName' => 'Kumar', 'email' => rahul@gmail.com');
echo json_encode($arr2);
?>
</body>
</html>
```

Decode

```
<!DOCTYPE html>

<html>
<body>
```

```
<?php
$json2 = '{"firstName": "Rahul", "lastName": "Kumar","email": "rahul@gmail.com"}';
var_dump(json_decode($json2, true));
?>
</body>
</html>
```

10. Write a program to encode and decode JSON object in JAVA.

Encode:

```
 package com.mycompany.newjson;
import org.json.*;
public class NewJson {
   public static void main(String[] args) {
      JSONObject obj=new JSONObject();
      obj.put("name","sonoo");
      obj.put("age",new Integer(27));
      obj.put("salary",new Double(600000));
      System.out.print(obj);
   }
}
```

O/P:

{"name":"sonoo","salary":600000,"age":27}

Decode:

```
import org.json.simple.JSONObject;
import org.json.simple.JSONValue;
public class JsonDecodeExample1 {
public static void main(String[] args) {
    String s="{\"name\":\"sonoo\",\"salary\":600000.0,\"age\":27}";
    Object obj=JSONValue.parse(s);
    JSONObject jsonObject = (JSONObject) obj;
    String name = (String) jsonObject.get("name");
    double salary = (Double) jsonObject.get("salary");
    long age = (Long) jsonObject.get("age");
    System.out.println(name+" "+salary+" "+age);
}
}
```

11. Write a program to decode JSON object using AJAX.

```
<html>
<head>
<meta content="text/html; charset=utf-8">
<title>AJAX JSON by Javatpoint</title>
<script type="application/javascript">
```

```
function load()
{
  var url = "http://date.jsontest.com/";//use any url that have json data
  var request;

  if(window.XMLHttpRequest){
   request=new XMLHttpRequest();//for Chrome, mozilla etc
  }
  else if(window.ActiveXObject){
   request=new ActiveXObject("Microsoft.XMLHTTP");//for IE only
  }
  request.onreadystatechange  = function(){
    if (request.readyState == 4  )
    {
      var jsonObj = JSON.parse(request.responseText);//JSON.parse() returns JSON object
      document.getElementById("date").innerHTML =  jsonObj.date;
      document.getElementById("time").innerHTML = jsonObj.time;
    }
  }
  request.open("GET", url, true);
  request.send();
}
</script>
</head>
<body>
```

```
Date: <span id="date"></span><br/>

Time: <span id="time"></span><br/>


<button type="button" onclick="load()">Load Information</button>

</body>

</html>
```

O/P:

12.WAP to implement String Decoder

```
const StringDecoder = require('string_decoder').StringDecoder;

const decoder = new StringDecoder('utf8');


const buf1 = new Buffer('this is a test');

console.log(decoder.write(buf1));

const buf2 = new Buffer('74686973206973206120746c3a97374', 'hex');

console.log(decoder.write(buf2));


const buf3 = Buffer.from([0x62,0x75,0x66,0x66,0x65,0x72]);

console.log(decoder.write(buf3));
```

13.WAP to implement Query String

```
querystring = require('querystring');
```

```
const obj1=querystring.parse('name=Ajay&company=XyzLtd');
console.log(obj1);
```

14.WAP to implement events.

```
 var events = require('events');
 var eventEmitter = new events.EventEmitter();
 var connectHandler = function connected() {
   console.log('connection succesful.');
   eventEmitter.emit('data_received');
 }
 eventEmitter.on('connection', connectHandler);
 eventEmitter.on('data_received', function(){
            console.log('data received succesfully.');
 });
 eventEmitter.emit('connection');
 console.log("Program Ended.");
```

15.CRUD Operation in Node JS with Mysql

----Node.js MySQL Create Table----

For creating a table named "employees".

```
var mysql = require('mysql');
```

```
var con = mysql.createConnection({
host: "localhost",
user: "root",
password: "root",
database: "Employee"
});
con.connect(function(err) {
if (err) throw err;
console.log("Connected!");
var sql = "CREATE TABLE employees (id INT, name VARCHAR(255), age INT(3), city
VARCHAR(255))";
con.query(sql, function (err, result) {
if (err) throw err;
console.log("Table created");
});
});
```

------Create Table Having a Primary Key-----

```
var mysql = require('mysql');
var con = mysql.createConnection({
host: "localhost",
user: "root",
```

```
password: "root",
database: "Employee"
});
con.connect(function(err) {
if (err) throw err;
console.log("Connected!");
var sql = "CREATE TABLE employee2 (id INT PRIMARY KEY, name VARCHAR(255), age
INT(3), city VARCHAR(255))";
con.query(sql, function (err, result) {
if (err) throw err;
console.log("Table created");
});
});
```

----Insert Multiple Records NodeJS with MYSQL----

```
var mysql = require('mysql');
var con = mysql.createConnection({
host: "localhost",
user: "root",
password: "root",
database: "Employee"
});
con.connect(function(err) {
```

```
if (err) throw err;
console.log("Connected!");
var sql = "INSERT INTO employees (id, name, age, city) VALUES ?";
var values = [
['2', 'Bharat Kumar', '25', 'Mumbai'],
['3', 'John Cena', '35', 'Las Vegas'],
['4', 'Ryan Cook', '15', 'CA']
];
con.query(sql, [values], function (err, result) {
if (err) throw err;
console.log("Number of records inserted: " + result.affectedRows);
});
});
```

----Node.js MySQL Update Records----

Update city in "employees" table where id is 1.

```
var mysql = require('mysql');
var con = mysql.createConnection({
host: "localhost",
user: "root",
password: "root",
database: "Employee"
```

```
});
con.connect(function(err) {
if (err) throw err;
var sql = "UPDATE employees SET city = 'Delhi' WHERE city = 'Allahabad'";
con.query(sql, function (err, result) {
if (err) throw err;
console.log(result.affectedRows + " record(s) updated");
});
});
```

----------Replace the data of the "employee2" table with the following data:-----

```
var mysql = require('mysql');
var con = mysql.createConnection({
host: "localhost",
user: "root",
password: "root",
database: "Employee"
});
con.connect(function(err) {
if (err) throw err;
console.log("Connected!");
var sql = "ALTER TABLE employee2 ADD COLUMN salary INT(10)";
con.query(sql, function (err, result) {
if (err) throw err;
```

```
console.log("Table altered");
});
});
```

-----Node.js MySQL Insert Records-------

```
var mysql = require('mysql');
var con = mysql.createConnection({
host: "localhost",
user: "root",
password: "root",
database: "Employee"
});
con.connect(function(err) {
if (err) throw err;
console.log("Connected!");
var sql = "INSERT INTO employees (id, name, age, city) VALUES ('1', 'Ajeet Kumar',
'27', 'Allahabad')";
con.query(sql, function (err, result) {
if (err) throw err;
console.log("1 record inserted");
});
});
```

------Node.js MySQL Delete Records------

```
var mysql = require('mysql');
var con = mysql.createConnection({
host: "localhost",
user: "root",
password: "root",
database: "Employee"
});
con.connect(function(err) {
if (err) throw err;
var sql = "DELETE FROM employees WHERE city = 'Delhi'";
con.query(sql, function (err, result) {
if (err) throw err;
console.log("Number of records deleted: " + result.affectedRows);
});
});
```

--------Node.js MySQL Select Records-------

```
var mysql = require('mysql');
var con = mysql.createConnection({
host: "localhost",
user: "root",
```

```
password: "root",
database: "Employee"
});
con.connect(function(err) {
if (err) throw err;
con.query("SELECT * FROM employees", function (err, result) {
if (err) throw err;
console.log(result);
});
});
```

------Node.js MySQL SELECT Unique Record(WHERE Clause)--------

```
var mysql = require('mysql');
var con = mysql.createConnection({
host: "localhost",
user: "root",
password: "root",
database: "Employee"
});
con.connect(function(err) {
if (err) throw err;
con.query("SELECT * FROM employees WHERE id = '2'", function (err, result) {
if (err) throw err;
console.log(result);
```

```
});
});
```

-----Node.js MySQL Select Wildcard----

```
var mysql = require('mysql');
var con = mysql.createConnection({
host: "localhost",
user: "root",
password: "root",
database: "Employee"
});
con.connect(function(err) {
if (err) throw err;
con.query("SELECT * FROM employees WHERE city LIKE 'M%'", function (err, result)
{
if (err) throw err;
console.log(result);
});
});
```

----Node.js MySQL Drop Table----

```
var mysql = require('mysql');
```

```
var con = mysql.createConnection({
host: "localhost",
user: "root",
password: "root",
database: "Employee"
});
con.connect(function(err) {
if (err) throw err;
var sql = "DROP TABLE employee2";
con.query(sql, function (err, result) {
if (err) throw err;
console.log("Table deleted");
});
});
```