IRC_Java_D10_Java io_CE_COD

Test Summary

- No. of Sections: 1
- No. of Questions: 5
- Total Duration: 180 min

Section 1 - CODING

Section Summary

- No. of Questions: 5
- Duration: 180 min

Additional Instructions:

None

Q1. **Problem statement:**
Write a Java program to count the number of days between two given years.

Input Format

The input consists of two integers .start year and end year.

Output Format

The output prints the number of days between two years and if the end year is greater than the start year then it has to print "End year must be greater than the first year !".

Sample Input

```
1995
2021
```

Sample Output

```
Year: 1995 = 365
Year: 1996 = 366
Year: 1997 = 365
Year: 1998 = 365
```

Sample Input

```
1993
1990
```

Sample Output

```
End year must be greater than first year!
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q2. **Problem statement:**
Write a Java program to create a Date object using the Calendar class.
Note:
Jan -0 to Dec -11

Input Format

The input consists of three integers, year, month, and date.

Output Format

The output prints the date. Refer to the sample input and output for the formatting specifications.

Sample Input

```
2016
0
1
```

Sample Output

```
Fri Jan 01 00:00:00 GMT 2016
```

Sample Input

```
2016
1
21
```

Sample Output

```
Sun Feb 21 00:00:00 GMT 2016
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q3. **Problem statement:**
Write a Java program to extract the date, and time from the date string.

Input Format

The input consists of a string and the format for the date and time is, YYYY-MM-DD Time:Hours: Minutes

Example : 2016-07-14 09:00:02

Output Format

The output prints the extract form of Date and Time. The format is MM/DD/YYYY Time: Hours: Minutes.

Sample Input

```
2016-07-14 09:00:02
```

Sample Output

```
07/14/2016, 9:00:02
```

Sample Input

```
2019-03-14 03:00:05
```

Sample Output

```
03/14/2019, 3:00:05
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q4. **Problem statement:**
write a program to find the instant method of clock class.

Input Format

The input consists of a string that represents the theZoneID.

Output Format

The output prints the instant method of the clock class. Refer to the sample input and output for the formatting specifications.

Sample Input

```
Europe/Paris
```

Sample Output

```
Instant for class SystemClock[Europe/Paris] is 2022-06
```

Sample Input

```
US/Hawaii
```

Sample Output

```
Instant for class SystemClock[US/Hawaii] is 2022-06-12
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q5. **Problem statement:**
Write a program to find the day and add the day to get the exact day of the week.

Input Format

The input consists of two integers. The first input to get the day. The second input is to get the number of days to be added with the day.

Output Format

The output prints the day of the first input and then the added value and that day . Refer to the sample input and output for the formatting specifications.

Sample Input

```
2
3
```

Sample Output

```
TUESDAY
5
FRIDAY
```

Sample Input

```
7
5
```

Sample Output

```
SUNDAY
5
```

	FRIDAY

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Answer Key & Solution

Section 1 - CODING

Q1

Test Case

Input

```
2002
2010
```

Output

```
Year: 2002 = 365
Year: 2003 = 365
Year: 2004 = 366
Year: 2005 = 365
```

Weightage - 25

Input

```
2005
2010
```

Output

```
Year: 2005 = 365
Year: 2006 = 365
Year: 2007 = 365
Year: 2008 = 366
```

Weightage - 25

Input

```
2021
2008
```

Output

```
End year must be greater than first year!
```

Weightage - 25

Input

```
2024
2022
```

Output

```
End year must be greater than first year!
```

Weightage - 25

Sample Input

```
1995
2021
```

Sample Output

```
Year: 1995 = 365
Year: 1996 = 366
Year: 1997 = 365
Year: 1998 = 365
```

Sample Input

```
1993
1990
```

Sample Output

```
End year must be greater than first year!
```

Solution

```
import java.util.Scanner;
public class Main
{
     public static void main(String[] args)
     {
        //System.out.println("Input start year:");
        Scanner s = new Scanner(System.in);
        int fy = s.nextInt();
```

```
    //System.out.println("\nInput end year:");
    int ly = s.nextInt();
    if (ly > fy)
    {
         for (int i = fy; i <= ly; i++)
         {
           System.out.println("Year: " + i + " = " + number_of_days(i));
         }
    }
    else
    {
        System.out.println("End year must be greater than first year!");
    }
}
    public static int number_of_days(int year)
    {
       if (is_Leap_Year(year))
            return 366;
       else
            return 365;
    }
    public static boolean is_Leap_Year(int year)
        {
           return (year % 4 == 0 && year % 100 != 0) || year % 400 == 0;
        }
}
```

Q2

Test Case

Input

```
2022
5
9
```

Output

```
Thu Jun 09 00:00:00 GMT 2022
```

Weightage - 25

Input

```
2021
03
21
```

Output

```
Wed Apr 21 00:00:00 GMT 2021
```

Weightage - 25

Input

```
2021
8
28
```

Output

```
Tue Sep 28 00:00:00 GMT 2021
```

Weightage - 25

Input

```
2017
3
24
```

Output

```
Mon Apr 24 00:00:00 GMT 2017
```

Weightage - 25

Sample Input

```
2016
0
1
```

Sample Output

```
Fri Jan 01 00:00:00 GMT 2016
```

Sample Input

```
2016
1
21
```

Sample Output

```
Sun Feb 21 00:00:00 GMT 2016
```

Solution

```
import java.util.*;
public class Main
{
    public static void main(String[] args)
    {
        Scanner  s = new Scanner(System.in);
        int year; //= 2016;
        int month;// = 0; // January
        int date;// = 1;
        year=s.nextInt();
        month=s.nextInt();
        date=s.nextInt();

        Calendar cal = Calendar.getInstance();
        // Sets the given calendar field value and the time value
        // (millisecond offset from the Epoch) of this Calendar undefined.
        cal.clear();
        System.out.println();
        cal.set(Calendar.YEAR, year);
        cal.set(Calendar.MONTH, month);
        cal.set(Calendar.DATE, date);

        System.out.println(cal.getTime());
        System.out.println();
    }
}
```

Q3

Test Case

Input

```
2022-06-21 05:10:05
```

Output

```
06/21/2022, 5:10:05
```

Weightage - 25

Input

```
2022-06-19 02:30:06
```

Output

```
06/19/2022, 2:30:06
```

Weightage - 25

Input

```
2019-03-14 12:08:05
```

Output

```
03/14/2019, 12:08:05
```

Weightage - 25

Input

```
2019-03-14 11:08:10
```

Output

```
03/14/2019, 11:08:10
```

Weightage - 25

Sample Input

```
2016-07-14 09:00:02
```

Sample Output

```
07/14/2016, 9:00:02
```

Sample Input

```
2019-03-14 03:00:05
```

Sample Output

```
03/14/2019, 3:00:05
```

Solution

```
import java.util.*;
import java.text.*;

class Main
{
  public static void main(String[] args)
   {
        try
        {
            String str;
            //= "2016-07-14 09:00:02";
            Scanner  s= new Scanner(System.in);
            str=s.nextLine();
            Date date = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss").parse(str);
            String newstr = new SimpleDateFormat("MM/dd/yyyy, H:mm:ss").format(date);
            System.out.println(newstr);
        }
        catch (ParseException e)
        {
                //Handle exception here
                 e.printStackTrace();
        }
   }
}
```

Q4

Test Case

Input

Output

Africa/Luanda	Instant for class SystemClock[Africa/Luanda] is 20

Weightage - 25

Input **Output**

Brazil/West	Instant for class SystemClock[Brazil/West] is 2022

Weightage - 25

Input **Output**

Canada/Eastern	Instant for class SystemClock[Canada/Eastern] is 2(

Weightage - 25

Input **Output**

US/Eastern	Instant for class SystemClock[US/Eastern] is 2022-(

Weightage - 25

Sample Input **Sample Output**

Europe/Paris	Instant for class SystemClock[Europe/Paris] is 202

Sample Input **Sample Output**

US/Hawaii	Instant for class SystemClock[US/Hawaii] is 2022-0(

Solution

```
import java.time.*;
import java.util.*;
public class Main
{
    public static void main(String[] args)
    {
        String str;
        Scanner s = new Scanner(System.in);
        str=s.nextLine();
        ZoneId zoneId = ZoneId.of(str);//"Europe/Paris"
        Clock clock = Clock.system(zoneId);
        Instant instantObj = clock.instant();
        ZonedDateTime time = instantObj.atZone(clock.getZone());
        System.out.println("Instant for class " + clock + " is " + time.toString());
```

```
    }
}
```

Q5

Test Case

Input

```
1
4
```

Output

```
MONDAY
5
FRIDAY
```

Weightage - 25

Input

```
3
2
```

Output

```
WEDNESDAY
5
FRIDAY
```

Weightage - 25

Input

```
6
1
```

Output

```
SATURDAY
7
SUNDAY
```

Weightage - 25

Input

```
2
1
```

Output

```
TUESDAY
3
WEDNESDAY
```

Weightage - 25

Sample Input

```
2
3
```

Sample Output

```
TUESDAY
5
FRIDAY
```

Sample Input

```
7
5
```

Sample Output

```
SUNDAY
5
FRIDAY
```

Solution

```
import java.time.DayOfWeek;
import java.util.*;
class Main
{
    public static void main(String args[])
    {
        Scanner s = new Scanner(System.in);
        int da;
```

```
        da=s.nextInt();
        long adv;
        adv=s.nextLong();
        DayOfWeek day = DayOfWeek.of(da);
        System.out.println(day.name());
        day = day.plus(adv);
        System.out.println(day.getValue());
        System.out.println(day.name());
    }
}
```