Datumsberechnungen

Angewandte Mathematik, SS11

Michael Unverzart
Manuel Wurth

Teil 1:

- Schaltjahrberechnung
- Datum plus Tage
- Doomsdaymethode

Schaltjahr

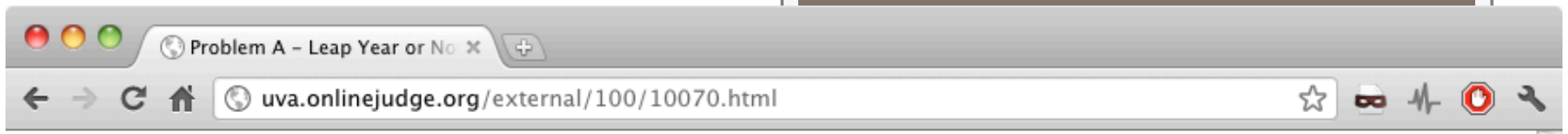
- Angleichung an das Tropischen Jahr
- 29. Februar als Schalttag

Schaltjahrberechnung

- Im Gregorianischen Kalender
  - Jahreszahl durch 4 teilbar
  - Ausnahme 1: Jahreszahl durch 100 teilbar
    -> KEIN Schaltjahr.
  - Ausnahme 2: Jahreszahl durch 400 teilbar
    -> IMMER ein Schaltjahr.

Schaltjahrberechnung

```
boolean schaltjahr =
          ( (jahr % 4   == 0   &&
             jahr % 100 != 0) ||
             jahr % 400 == 0 );
```



Leap Year or Not Leap Year and …

Input: standard input
Output: standard output

The ancient race of Gulamatu is very advanced in their year calculation scheme. They understand what leap year is (A year that is divisible by 4 and not divisible by 100 with the exception that years that are divisible by 400 are also leap year.) and they have also similar festival years. One is the Huluculu festival (happens on years divisible by 15) and the Bulukulu festival (Happens on years divisible by 55 provided that is also a leap year). Given an year you will have to state what properties these years have. If the year is not leap year nor festival year, then print the line 'This is an ordinary year.' The order of printing (if present) the properties is leapyear-->huluculu-->bulukulu.

Input

Input will contain several years as input. Each year will be in separate lines. Input is terminated by end of file. All the years will not be less than 2000 (to avoid the earlier different rules for leap years). Please don't assume anything else.

Output

For each input, output the different properties of the years in different lines according to previous description and sample output. A blank line should separate the output for each line of input. Note that there are four different properties.

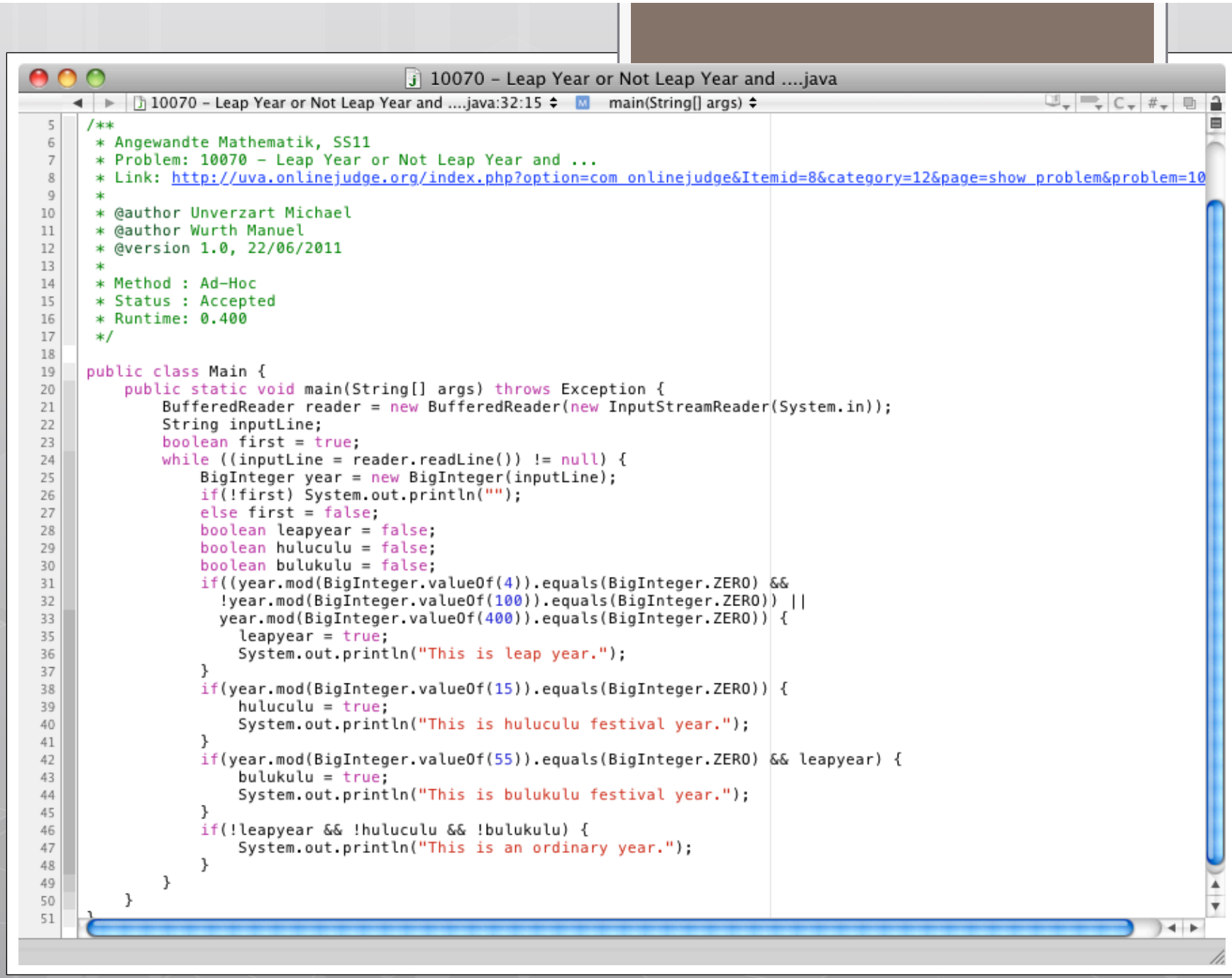
Sample Input

```
2000
3600
4515
2001
```

Sample Output

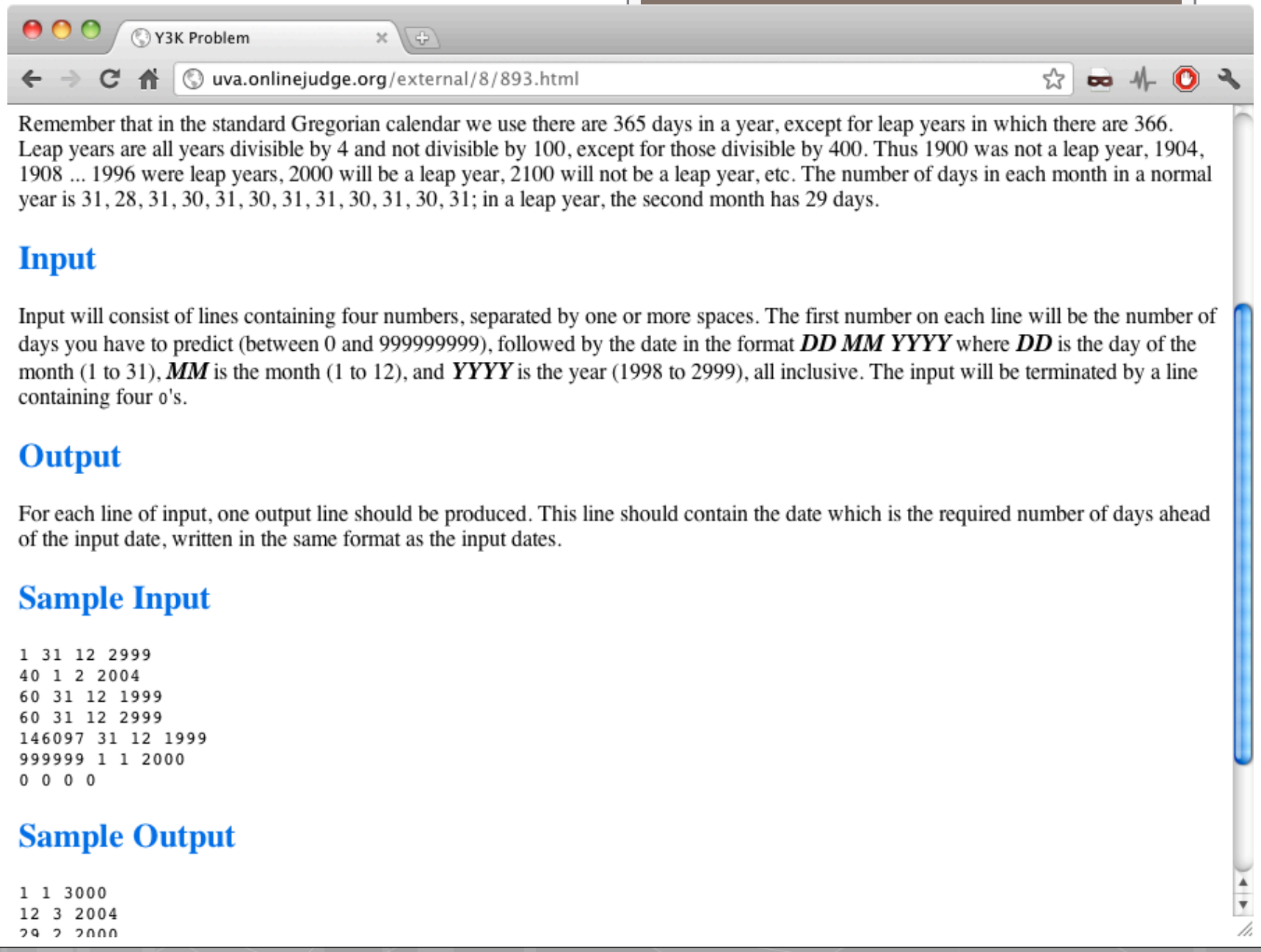
```
This is leap year.

This is leap year.
This is huluculu festival year.
```


10070 – Leap Year or Not Leap Year and ....java
10070 – Leap Year or Not Leap Year and java:32:15
main(String[] args)
/**
* Angewandte Mathematik, SS11
* Problem: 10070 – Leap Year or Not Leap Year and ...
* Link: http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=12&page=show_problem&problem=10
*
* @author Unverzart Michael
* @author Wurth Manuel
* @version 1.0, 22/06/2011
*
* Method : Ad-Hoc
* Status : Accepted
* Runtime: 0.400
*/
public class Main {
public static void main(String[] args) throws Exception {
BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
String inputLine;
boolean first = true;
while ((inputLine = reader.readLine()) != null) {
BigInteger year = new BigInteger(inputLine);
if(!first) System.out.println("");
else first = false;
boolean leapyear = false;
boolean huluculu = false;
boolean bulukulu = false;
if((year.mod(BigInteger.valueOf(4)).equals(BigInteger.ZERO) &&
!year.mod(BigInteger.valueOf(100)).equals(BigInteger.ZERO)) ||
year.mod(BigInteger.valueOf(400)).equals(BigInteger.ZERO)) {
leapyear = true;
System.out.println("This is leap year.");
}
if(year.mod(BigInteger.valueOf(15)).equals(BigInteger.ZERO)) {
huluculu = true;
System.out.println("This is huluculu festival year.");
}
if(year.mod(BigInteger.valueOf(55)).equals(BigInteger.ZERO) && leapyear) {
bulukulu = true;
System.out.println("This is bulukulu festival year.");
}
if(!leapyear && !huluculu && !bulukulu) {
System.out.println("This is an ordinary year.");
}
}
}
}

Datum plus Tage

- 893 - Y3K Problem
- 11356 - Dates



Remember that in the standard Gregorian calendar we use there are 365 days in a year, except for leap years in which there are 366. Leap years are all years divisible by 4 and not divisible by 100, except for those divisible by 400. Thus 1900 was not a leap year, 1904, 1908 ... 1996 were leap years, 2000 will be a leap year, 2100 will not be a leap year, etc. The number of days in each month in a normal year is 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31; in a leap year, the second month has 29 days.

Input

Input will consist of lines containing four numbers, separated by one or more spaces. The first number on each line will be the number of days you have to predict (between 0 and 999999999), followed by the date in the format ***DD MM YYYY*** where ***DD*** is the day of the month (1 to 31), ***MM*** is the month (1 to 12), and ***YYYY*** is the year (1998 to 2999), all inclusive. The input will be terminated by a line containing four 0's.

Output

For each line of input, one output line should be produced. This line should contain the date which is the required number of days ahead of the input date, written in the same format as the input dates.

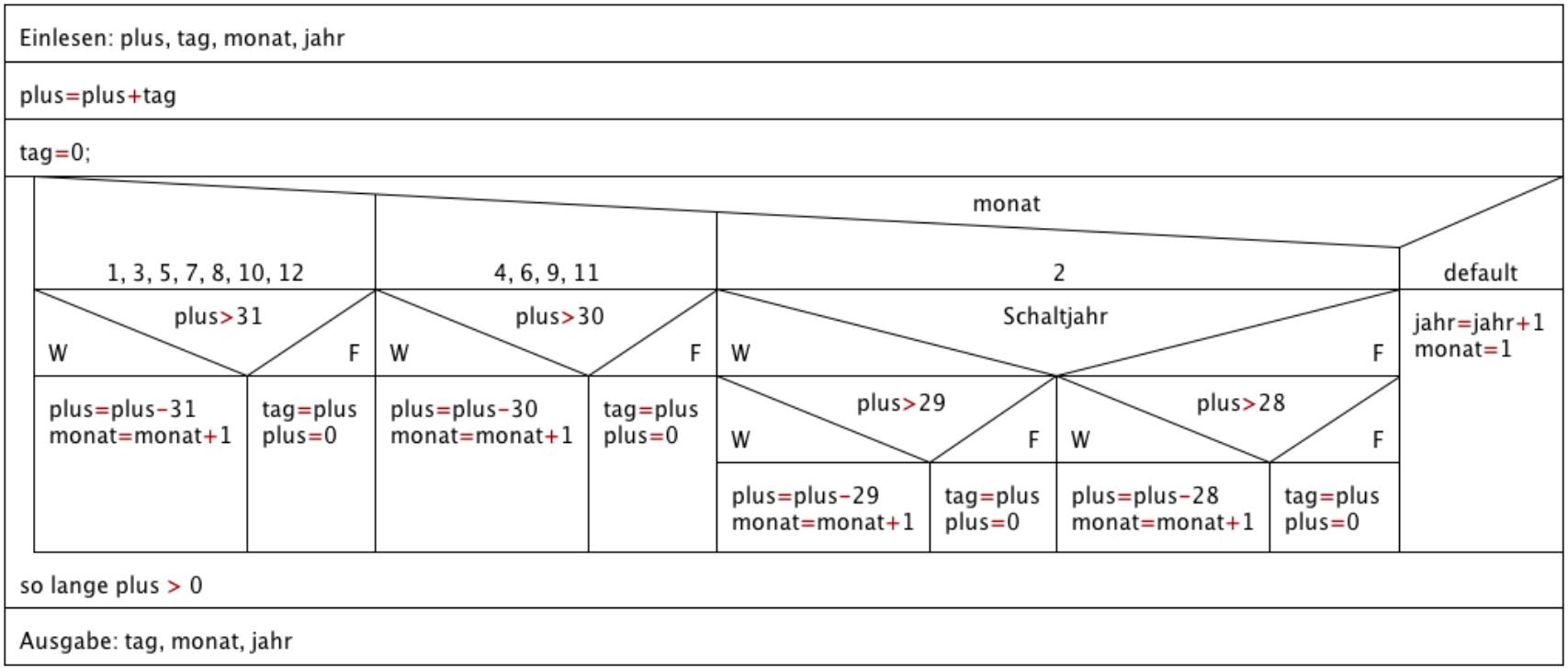
Sample Input

```
1 31 12 2999
40 1 2 2004
60 31 12 1999
60 31 12 2999
146097 31 12 1999
999999 1 1 2000
0 0 0 0
```

Sample Output

```
1 1 3000
12 3 2004
29 2 2000
```

Y3K Problem


Einlesen: plus, tag, monat, jahr
plus=plus+tag
tag=0;
monat
1, 3, 5, 7, 8, 10, 12
4, 6, 9, 11
2
default
plus>31
plus>30
Schaltjahr
W
F
W
F
W
F
jahr=jahr+1
monat=1
plus=plus-31
monat=monat+1
tag=plus
plus=0
plus=plus-30
monat=monat+1
tag=plus
plus=0
plus>29
plus>28
W
F
W
F
plus=plus-29
monat=monat+1
tag=plus
plus=0
plus=plus-28
monat=monat+1
tag=plus
plus=0
so lange plus > 0
Ausgabe: tag, monat, jahr

Doomsdaymethode

- Bestimmung des Wochentags
- Von John Horton Conway
- Durchführbar mit Kopfrechenoperationen

Doomsdaymethode

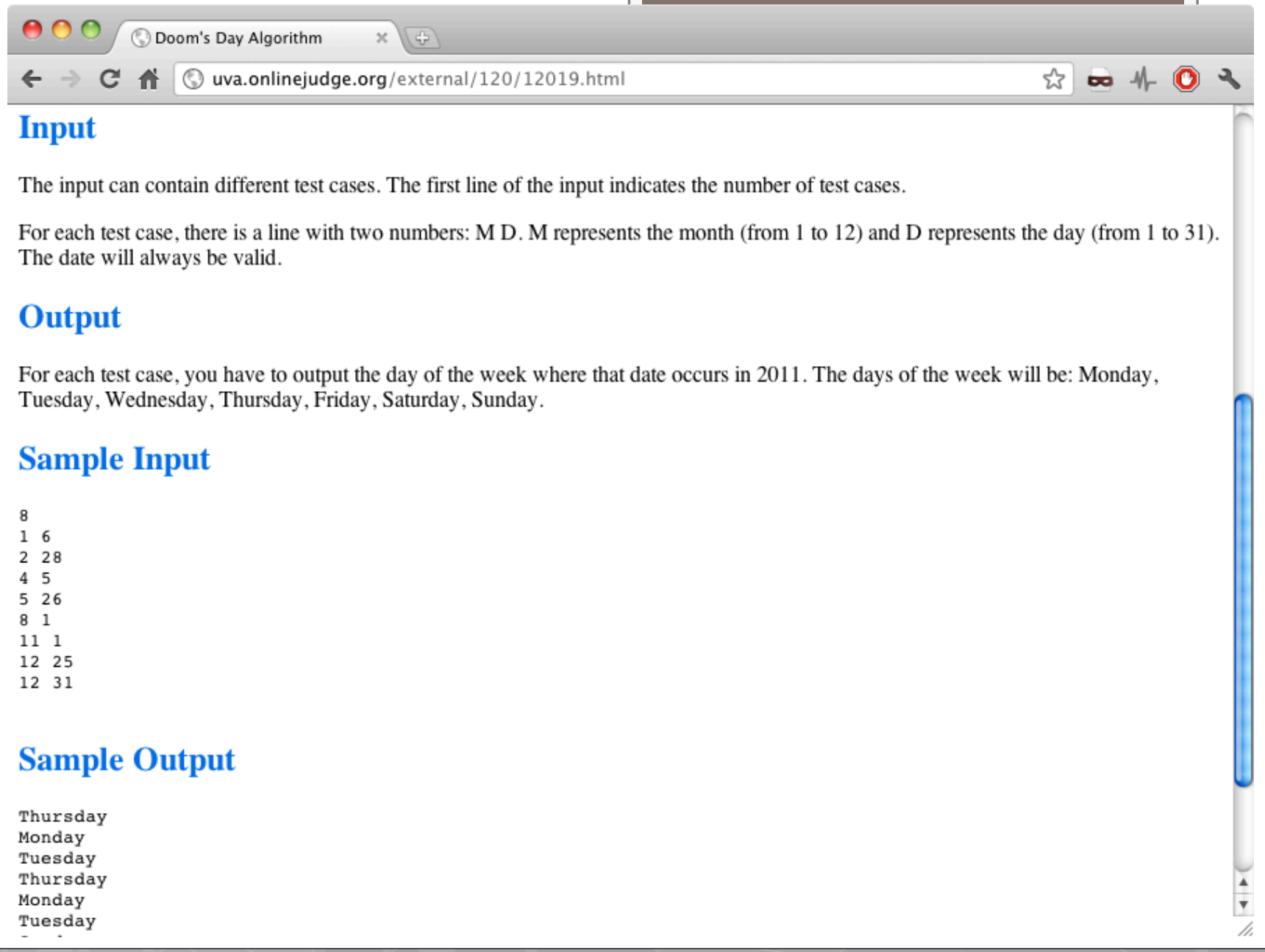
- Doomsday = letzter Tag im Februar

So	Mo	Di	Mi	Do	Fr	Sa
1999		2000	2001	2002	2003	
2004	2005	2006	2007		2008	2009
2010	2011		2012	2013	2014	2015
	2016	2017	2018	2019		2020

Doomsdaymethode

- Weitere Doomsdays:
  - Ab April in geraden Monaten - Tag identisch mit der Monatszahl
  - 9. 5., 5. 9., 11. 7. und 7. 11.
  - 3. Januar, im Schaltjahr 4. Januar
  - Letzter Tag im Februar
  - Im März alle Tage die durch 7 teilbar

Monat	Doomsday	im Schaltjar	Merkhilfe
Januar	3.1	4.1	drei Jahre 3 im vierten 4
Februar	28.2	29.2	letzter
März	0.3, 7.3, ...		nullter, oder durch 7 teilbar
April	4.4		
Mai	9.5		nine to five
Juni	6.6		
Juli	11.7		seven-eleven
August	8.8		
September	5.9		nine to five
Oktober	10.10		
November	7.11		seven-eleven
Dezember	12.12		



Input

The input can contain different test cases. The first line of the input indicates the number of test cases.

For each test case, there is a line with two numbers: M D. M represents the month (from 1 to 12) and D represents the day (from 1 to 31). The date will always be valid.

Output

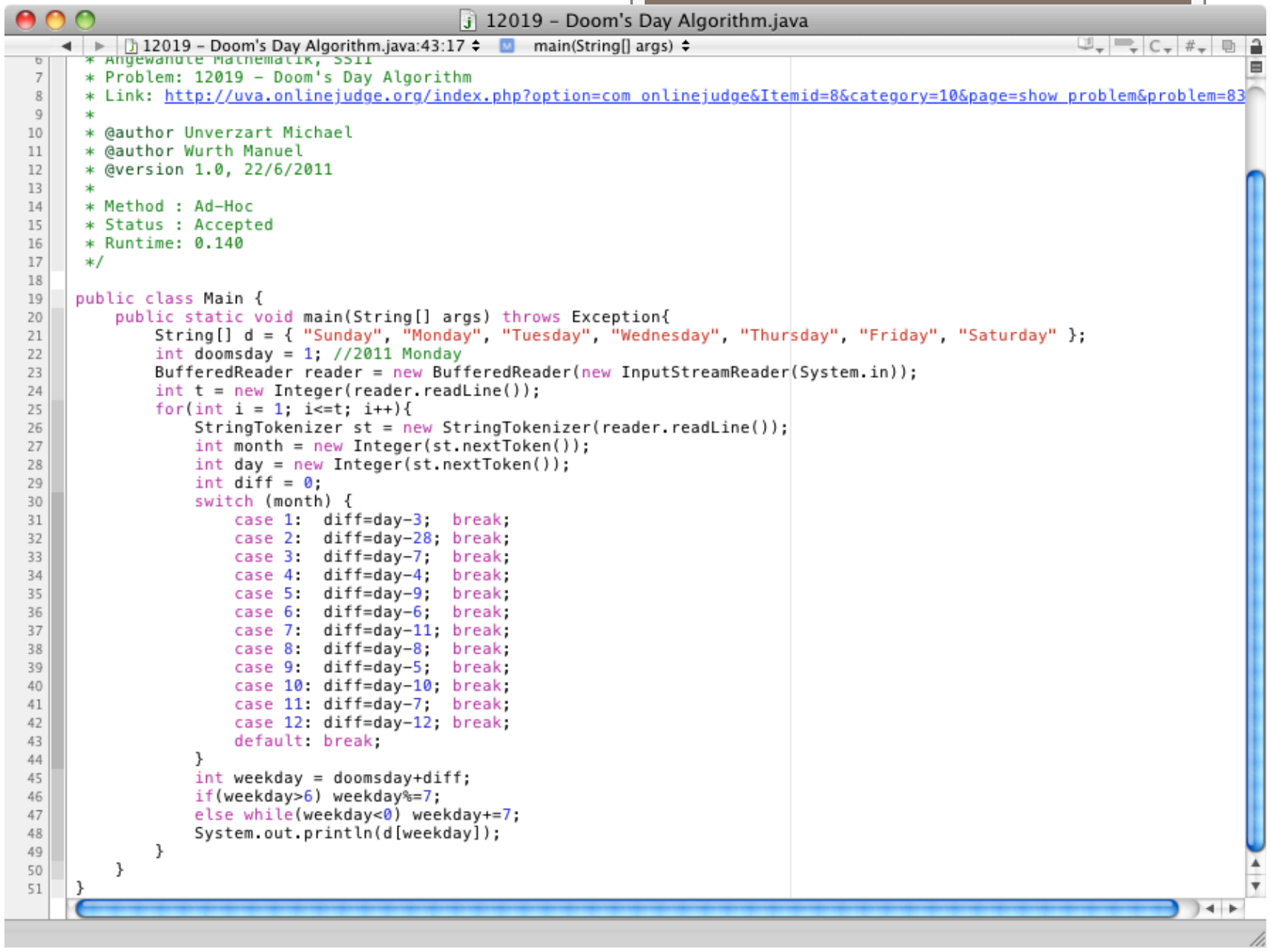
For each test case, you have to output the day of the week where that date occurs in 2011. The days of the week will be: Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday.

Sample Input

```
8
1 6
2 28
4 5
5 26
8 1
11 1
12 25
12 31
```

Sample Output

```
Thursday
Monday
Tuesday
Thursday
Monday
Tuesday
```


12019 - Doom's Day Algorithm.java
12019 - Doom's Day Algorithm.java:43:17
main(String[] args)
* Angewandte Mathematik, SS11
* Problem: 12019 - Doom's Day Algorithm
* Link: http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=10&page=show_problem&problem=83
*
* @author Unverzart Michael
* @author Wurth Manuel
* @version 1.0, 22/6/2011
*
* Method : Ad-Hoc
* Status : Accepted
* Runtime: 0.140
*/
public class Main {
public static void main(String[] args) throws Exception{
String[] d = { "Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday" };
int doomsday = 1; //2011 Monday
BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
int t = new Integer(reader.readLine());
for(int i = 1; i<=t; i++){
StringTokenizer st = new StringTokenizer(reader.readLine());
int month = new Integer(st.nextToken());
int day = new Integer(st.nextToken());
int diff = 0;
switch (month) {
case 1: diff=day-3; break;
case 2: diff=day-28; break;
case 3: diff=day-7; break;
case 4: diff=day-4; break;
case 5: diff=day-9; break;
case 6: diff=day-6; break;
case 7: diff=day-11; break;
case 8: diff=day-8; break;
case 9: diff=day-5; break;
case 10: diff=day-10; break;
case 11: diff=day-7; break;
case 12: diff=day-12; break;
default: break;
}
int weekday = doomsday+diff;
if(weekday>6) weekday%=7;
else while(weekday<0) weekday+=7;
System.out.println(d[weekday]);
}
}
}

Teil 2:

- Julianischer VS Gregorianischer Kalender
- Wochentags Berechnung

Julianischer VS Gregorianischer Kalender

- Julianische Kalender wurde 1582 päpstlich durch den Gregorianischen Kalender abgelöst
- In China erst im Jahre 1949

Hauptunterschiede

- Julianisches Jahr: 365,25 Tage
- Gregorianisches Jahr: 365,2425 Tage
- Sonnenjahr: 365,242190417 Tage
- Schaltjahrregel des Julianischen Kalenders sehr einfach: Alle 4 Jahre (alle 00 Jahre)

Wochentags Berechnung

- Anhand eines gegebenen Datums den Wochentag ermitteln
- Im Julianischen & Gregorianischen Kalender

Einführung von 4 Ziffern

- Tagesziffer
- Monatsziffer
- Jahresziffer
- Jahrhunderts Ziffer

Tagesziffer

- Berechnung: [Tag im Monat] Modulo 7

- Beispiel: 12.09.2011
- 12 / 7 = 1 Rest: 5
- Tagesziffer = 5

Monatsziffer 1/2

- Festlegung: Januar hat Monatsziffer 0
- Errechnung am Beispiel Februar:
- Der Januar hat 31 Tage.
- 31 Modulo 7 = 3
- Der Februar hat daher Monatsziffer 0 + 3 = 3
- Der Februar hat 28 Tage
- 28 Modulo 7 = 0
- Der März hat daher 0+3+0 = ebenfalls 3

Monatsziffer 2/2

- Der März hat 31 Tage
- 31 Modulo 7 = 3
- Der April hat demnach 0+3+0+3 = 6

- Das wird so fortgesetzt
- Wenn die Monatsziffer größer 6 wird, wird die Ziffer nochmals Modulo 7 gerechnet, damit die Ziffer wieder zwischen 1 und 6 liegt

Jahresziffer 1/4

- Betrachtet werden nur die letzten beiden Ziffer einer Jahreszahl, also nicht die Jahrhunderte.
- Beispiel: 1955
- In Frage kommt nur die Zahl 55

Jahresziffer 2/4

- Zur Jahreszahl addiert man zunächst das Ganzzahlergebnis der Division durch 4 derselben Zahl
- Beispiel:
- 55 + (55 / 4)
- 55 + 13 = 68
- Dieses Ergebnis wird nochmals Modulo 7 gerechnet
- Also: 68 Modulo 7 = 5

Jahresziffer 3/4

- Dadurch ergibt sich folgendes Schema:

Jahr	**00**	**01**	**02**	**03**	**04**	**05**	**06**	**07**	**08**
Ziffer	0	1	2	3	5	6	0	1	3
Jahr	**09**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**
Ziffer	4	5	6	1	2	3	4	6	0
Jahr	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**
Ziffer	1	2	4	5	6	0	2	3	4
Jahr	**27**	**28**							
Ziffer	5	0							

Jahreszahl 4/4

- Es wird immer um 1 weitergezählt, nur in Schaltjahren um 2.
- Nach 6 geht der Zyklus wieder bei 0 weiter
- Der Gesamtzyklus wiederholt sich alle 28 Jahre
- Deshalb war z.B. 1931 genau wie das Jahr 1959 und wie das Jahr 1987, was die Wochentage betrifft

Jahrhunderts Ziffer 1/4 Gregorianisch

- Hierfür werden nur die ersten beiden Zahlen der Jahreszahl verwendet. Also das Jahrhundert.
- Beispiel 1955
- In Frage kommt nur die Zahl 19

Jahrhunderts Ziffer 2/4 Gregorianisch

- Die Formel lautet:
- (3 – (Jahrhundert Modulo 4)) * 2
- Sie ist 0 für alle Jahre, die mit 19, 23, 27 anfangen.
- Sie ist 2 für alle Jahre, die mit 18, 22, 26 anfangen.
- Sie ist 4 für alle Jahre, die mit 17, 21, 25 anfangen.
- Sie ist 6 für alle Jahre, die mit 16, 20, 24 anfangen.

Jahrhunderts Ziffer 3/4 Gregorianisch

- Ein Zyklus von 400 Jahren hat 146097 Tage, und ist durch 7 teilbar

- Deshalb wiederholen sich die Wochentage zusätzlich alle 400 Jahre

Jahrhunderts Ziffer 4/4 Julianisch

- Die Formel lautet:
- 6 – ((Jahrhundert – 5) Modulo 7)

- Der Julianische Kalender hat einen Zyklus von 700 Jahren

- Also alle 700 Jahre gleiche Wochentage

Ergebnis

- Alle Ziffern werden addiert
- Wenn es ein Schaltjahr war und der Monat Januar oder Februar ist, nochmal +6 oder -1
- Das Ergebnis Modulo 7
- 0 = So, 1 = Mo, 2 = Di, 3 = Mi, 4 = Do, 5 = Fr, 6 = Sa

Quellen

- wikipedia.org
- uva.onlinejudge.org