**Übung zu enum NEXT DAY**

Schreiben Sie ein Java-Programm (Klasse NextDay), welches durch Eingabe des Tages, des Monats und des Jahres das Datum des nächsten Tages berechnet und ausgibt. Dazu ist es notwendig, dass Sie eine Klasse Date mit den Attributen Tag, Monat und Jahr de nieren. Es soll möglich sein ein Objekt dieser Klasse durch einen Konstruktor mit genau diesen Parametern (Tag, Monat, Jahr) zu erzeugen. Die Klasse Date soll die Methode public Date getNextDay() besitzen, die ein neues Objekt der Klasse Date zurückgibt, das den nächsten Tag im Kalender repräsentiert. Überlegen Sie welche weiteren (evtl. privaten) Methoden zur Berechnung des Folgetages benötigt werden. Achtung, berücksichtigen Sie die folgenden Punkte:

Ein Jahr ist ein Schaltjahr, wenn es durch 4 aber nicht zugleich durch 100 teilbar ist. Ist das Jahr durch 400 teilbar, ist es aber wieder ein Schaltjahr.

Das Ausgangsdatum soll durch Kommandozeilenargumente übergeben werden. Programmstart von der Konsole (Die ersten 3 Kommandozeilenargumente sollen für den Tag, den Monat und das Jahr verwendet werden):

java NextDay <Tag> <Monat> <Jahr>

Beispielaufruf und -ausgabe:

java NextDay 19.3.2013

Der naechste Tag ist der 20.3.2013

Die Eingabe sollte auch auf Korrektheit geprüft werden! Falls einefehlerhafte Datumskonstellation (z.B. 32 12 2012 ) eingegeben wurde, soll die Ausnahme InvalidDateArgumentsException geworfen werden. De nieren Sie diese Klasse in geeigneter Form als Subklasse der Klasse java.lang.Exception.

Freiwillige Zusatzaufgabe:

Programmieren Sie Ihre Klasse so um, dass für die Repräsentation der Monate ein geeigneter Aufzählungstyp (enum) verwendet wird.

**Lösungsvorschlag**

public enum Enum\_Date

{

// Attribute

JANUAR ( "Januar"),

FEBRUAR ("Februar " ),

MAERZ ( "März " ),

APRIL ( "APRIL" ),

MAI ( "Mai " ),

JUNI ( "Juni" ),

JULI ( "Juli " ),

AUGUST ( "August" ),

SEPTEMBER ( "September" ),

OKTOBER ( "Oktober" ),

NOVEMBER ( "November" ),

DEZEMBER ( "Dezember " );

private String monat;

// privater Konstruktor

private Enum\_Date ( String monat )

{

this.monat = monat;

}

public String gibMonat ( )

{

return monat;

}

public static Enum\_Date gibMonatsObjekt ( int monat )

{

Enum\_Date m = null;

switch ( monat )

{

case 1 : m = Enum\_Date.JANUAR; break;

case 2 : m = Enum\_Date.FEBRUAR; break;

case 3: m = Enum\_Date.MAERZ; break;

case 4 : m = Enum\_Date.APRIL; break;

case 5 : m = Enum\_Date.MAI; break;

case 6 : m = Enum\_Date.JUNI; break;

case 7 : m = Enum\_Date.JULI; break;

case 8 : m = Enum\_Date.AUGUST; break;

case 9 : m = Enum\_Date.SEPTEMBER; break;

case 10 : m = Enum\_Date.OKTOBER; break;

case 11 : m = Enum\_Date.NOVEMBER; break;

case 12 : m = Enum\_Date.DEZEMBER; break;

default : System.out.println ( "fehler in gibMonatsObjekt");

}

return m;

}

public static int gibMonatsZahl ( Enum\_Date monat )

{

return monat.ordinal ( ) +1;

/\*

int zurueck = 0;

switch ( monat )

{

case JANUAR : zurueck = 1; break;

case FEBRUAR : zurueck = 2; break;

case MAERZ :zurueck = 3; break;

case APRIL :zurueck = 4; break;

case MAI :zurueck = 5; break;

case JUNI :zurueck = 6; break;

case JULI :zurueck = 7; break;

case AUGUST :zurueck = 8; break;

case SEPTEMBER :zurueck = 9; break;

case OKTOBER :zurueck = 10; break;

case NOVEMBER :zurueck = 11; break;

case DEZEMBER :zurueck =12; break;

default : System.out.println ( "fehler in gibMonatsZahl");

}

return zurueck;

\*/

}

// weitere Methode

/\*

public String toString ( )

{

return monat;

}

\*/

}

//……………………………………………………

public class Date {

private byte day;

private Enum\_Date month;

private short year;

public Date( byte day, Enum\_Date month, short year) throws InvalidDateException

{

if (!isCorrect( day, month ) ) //, year ))

throw new InvalidDateException ( "falscher Wert fuer Tag oder Monat ");

this.day=day;

this.month=month;

this.year=year;

}

public Date getNextDay() throws InvalidDateException

{

int max = getNumberOfDaysInMonth ( );

if ( this.day == max )

{

this.day = 1;

this.month = getNextMonth();

int zahl = Enum\_Date.gibMonatsZahl ( month );

if ( zahl == 1 )

{

this.year++;

//this.day = 1;

}

}

else

{

this.day++;

}

return new Date ( this.day, this.month, this.year );

}

private Enum\_Date getNextMonth ( )

{

int zahl = Enum\_Date.gibMonatsZahl ( month );

switch ( zahl )

{

case 12: Enum\_Date m = Enum\_Date.JANUAR; return m;

default: zahl ++;

return zurueck;

}

}

private byte getNumberOfDaysInMonth ( )

{

return getNumberOfDaysInMonth ( month );

}

private byte getNumberOfDaysInMonth ( Enum\_Date month )

{

int zahl = month.gibMonatsZahl ( month );

switch ( zahl )

{

case 1:

case 3:

case 5:

case 7:

case 8:

case 10:

case 12:

return 31;

case 2:

if (isLeapYear( ))

{

return 29;

} else return 28;

default: return 30;

}

}

public String toString() {

return day + "." + month + " " + year;

}

private boolean isLeapYear( )

{

return Math.abs( year ) % 400 == 0 ||

( Math.abs ( year ) % 4 == 0 && !( Math.abs ( year ) % 100 == 0 ));

}

private boolean isCorrect ( byte day, Enum\_Date month ) //), short year)

{

int zahl = month.gibMonatsZahl ( month );

if ( zahl < 1 || zahl > 12 || day < 1 || day > getNumberOfDaysInMonth( month ) )

return false;

else return true;

}

}

//……………………………….

public class Anwendung\_NextDay

{

public static void main(String[] args)

{

// mit runConfiguration ausführen ////////////////////////

/\*

Enum\_Date m = Enum\_Date.DEZEMBER;

System.out.println ( m );

System.out.println ( m.gibMonat ( ) );

System.out.println( m.gibMonatsZahl ( m ) );

System.out.println ( m.gibMonatsObjekt ( 10 ) );

\*/

// In args stehen die Daten, sie werden ( run configuratins ) eingefügt

if (args.length!=3)

{

System.out.println ( "usage: java NextDay <day> <month> <year>" );

}

int monatsZahl = Integer.parseInt ( args [ 1 ] ) ;

Enum\_Date monatsObjekt = Enum\_Date.gibMonatsObjekt ( monatsZahl );

try

{

System.out.println

( new Date(Byte.parseByte(args[0]), monatsObjekt,

Short.parseShort(args[2])).getNextDay());

}

catch ( NumberFormatException n )

{

System.out.println

( "Error: you did not numbers for day, month or ar!" );

n.printStackTrace ( );

}

catch ( InvalidDateException e )

{

System.out.println(

"Error: you entered an invalid combination of day, month an year!" );

e.printStackTrace ( );

}

}

}