

ALGORITHM GregorianCalendar

```
BEGIN
  VARIABLE day, month, year, yearH, yearR,
    help, weekDayNum TYPE Integer
  Read day, month, year
  IF month < 3 THEN
    Increase month by 10
    Decrease year by 1
  ELSE Decrease month by 2
  Divide year by 100,
    store the integer part into yearH,
    store the remainder into yearR
  Store day + yearR - 2*yearH into help
  Add to help
    integer part of (13*month-1)/5,
    integer part of yearR/4,
    integer part of yearH/4
  WHILE help < 0 DO Add 7 to help
  Divide help by 7, store the remainder
    into weekDayNum
  CASE DISTINCTION ACCORDING TO
    weekDayNum
    0: Write "Sun"
    1: Write "Mon"
    2: Write "Tue"
    3: Write "Wed"
    4: Write "Thu"
    5: Write "Fri"
    6: Write "Sat"
END
```

ALGORITHM MagicSquare

```
BEGIN
  STORAGE for a square with  $n \times n$  boxes (integer
    numbers),  $n$  an odd number
  Store the number 1 into the middle box of the first line
  WHILE there is still an empty box DO
    IF current box is in row 1 THEN
      IF current box is in utmost right column THEN
        Go one row downwards
      ELSE
        Go one column to the right
        Go to the utmost lower row
      ELSE
        Go one row upwards
    IF current box is in the utmost right column THEN
      Go into the utmost left column
    ELSE
      Go one column to the right
      IF there is already a number THEN
        Go one column to the left
        Go two rows downwards
      Store the next number (i. e. increased by 1)
        into this box
  END
```

Exercise 1 (belonging to the algorithm "GregorianCalendar")

Note

The Gregorian Calendar that replaced the Julian Calendar was introduced on different dates, e. g.:
Italy and other Catholic countries: on October 15, 1582 (day before: October 4),
Protestant German areas, Scandinavian countries: on March 1, 1700 (day before: February 18),
Great Britain, North America: on September 14, 1752 (day before: Sept. 2).

- Go through the program by hand with several dates, write the variables' values into a table (old values crossed out, but still readable afterwards!). Take (e. g.) the values February 29, 2000; March 1, 2000; February 29, 1900; March 1, 1900; your own birth day
- Generate the program flow chart and the structure chart, belonging to this algorithm!

Exercise 2 (belonging to the algorithm "MagicSquare")

- Go through the algorithm with $n = 3$ und $n = 5$! Try to do so with $n = 4$, too.
- Draw the program flow chart and the structure chart belonging to it.

Exercise 3

Formulate an algorithm for imitating executing a children's counting-out rhyme, i. e. as pseudo code and additionally as a structure chart:

Children of a fixed (selectable) number are standing in a circle; the counting-out rhyme which has a definite (selectable) number of syllables is started at one of the children. The child which is hit by the last syllable is eliminated. The algorithm shall find the winner (the last child) and output it.

There should be at least the following memory storages which shall get a value by the user:

| | |
|------------|--|
| numChild | number of children—can be a preset constant |
| childArray | array of numChild names of the children, numbered from 0 to numChild-1 |
| startChild | number of the child at which the counting-out rhyme shall start |
| numSyll | number of syllables of the counting-out rhyme |