Gregorian Calendar

This program is based on a code that I have found a long time ago somewhere in a computer magazine (It was given there: A$ sequences, methods of computing variables W and L). I made following calendars out of it:

- English, where the week starts on Sunday - **Kalen_en** program,
- another one, where the week starts on Monday - **Kalen_pl** program.

Each of these programs has the ability to:

- displaying only one month of a specific year,
- displaying the yearly calendar when nothing is entered after (press [Enter] immediately):

  `Month (1..12)   :` <-- in **Kalen_en** program

  or

  `Miesiąc (1..12) :` <-- in **Kalen_pl** program

If your language is neither English nor Polish (e.g. German) and you want to have this program then:

1. In the **Kalen_en** program, replace the English (or Polish **Kalen_pl** - depending on the name of the first day of the week) words that appear on the screen with your own (e.g. German) words, following the English names **Kalen_en**,
2. Save the calendar as e.g. **Kalen_du** (here: German version; remember: No more than 8 characters),
3. Start PC-Basic with the appropriate codepage option:

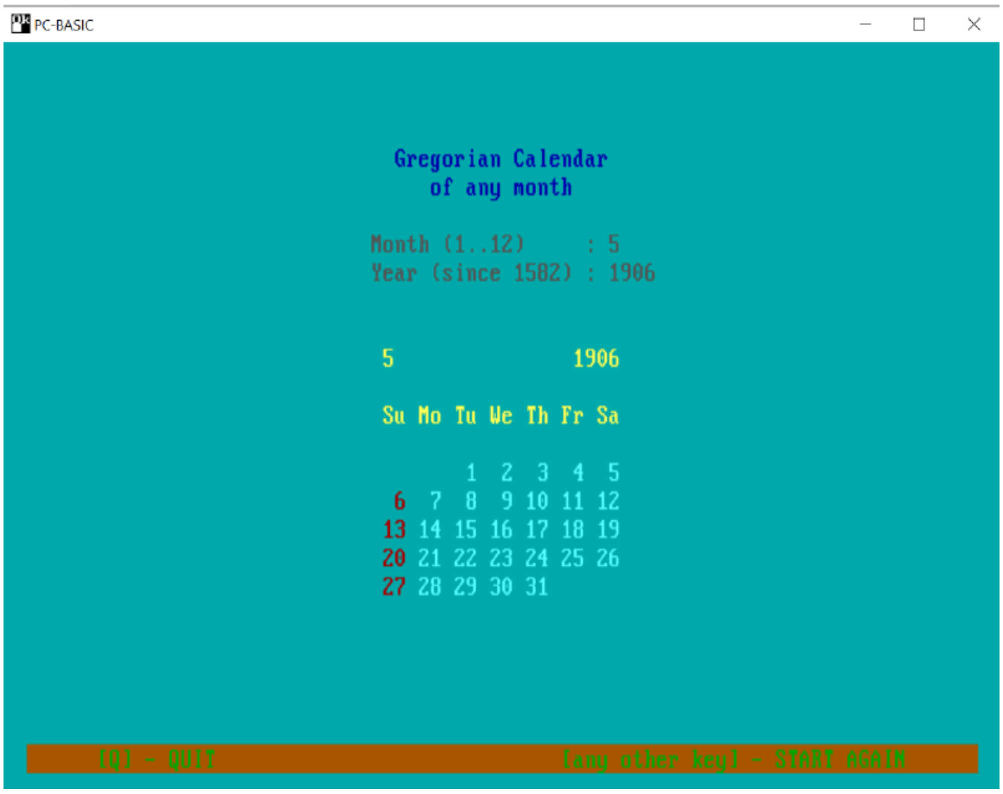
```
C:\GWBasic\pcbasic --codepage= Your code page
```

4. Load ([F3], e.g. LOAD "Kalen_du) and run ([F2], RUN) this program.

- - - - - - - - - -

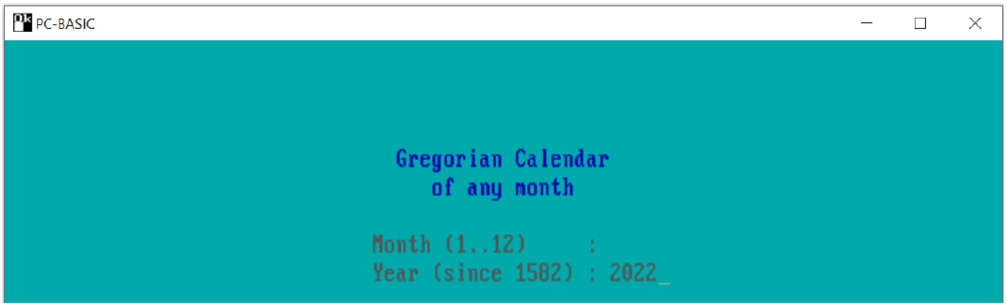
An example of how the program Kalen_en works

1. I choose a month and year - I see that my hardworking grandfather was not born on Sunday.



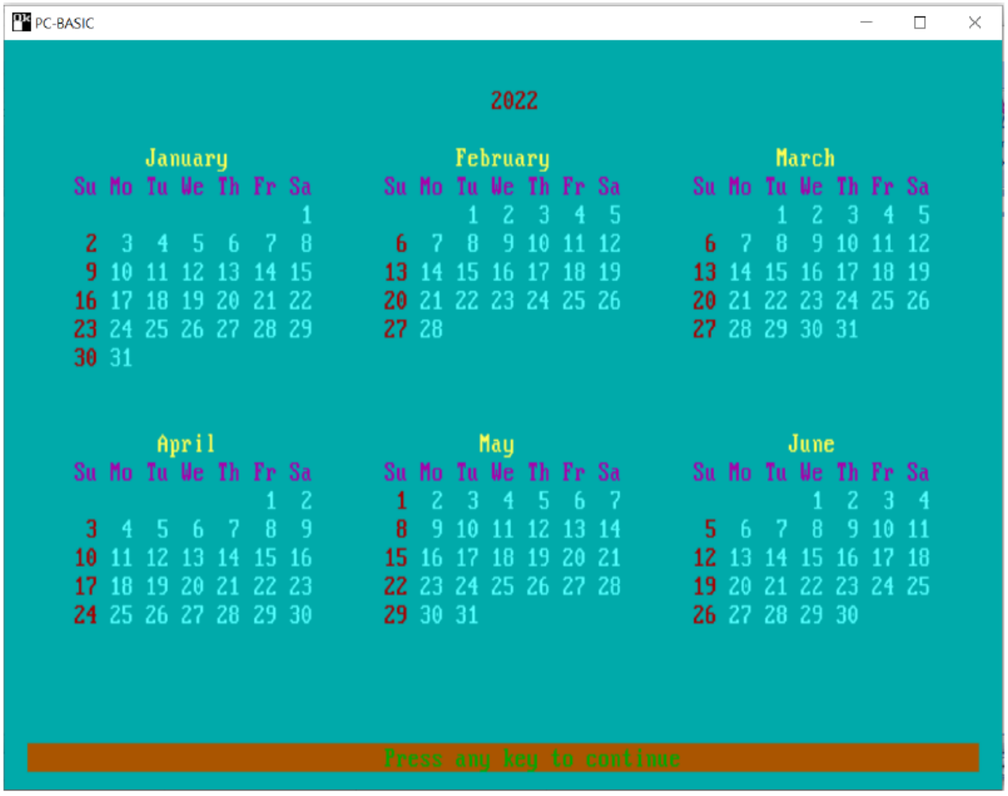
```
        Gregorian Calendar
          of any month

      Month (1..12)      : 5
      Year (since 1582)  : 1906


       5                1906

       Su Mo Tu We Th Fr Sa

                 1  2  3  4  5
        6  7  8  9 10 11 12
       13 14 15 16 17 18 19
       20 21 22 23 24 25 26
       27 28 29 30 31



   [Q] - QUIT                    [any other key] - START AGAIN
```

2. I ignore a month selection and enter a year only:

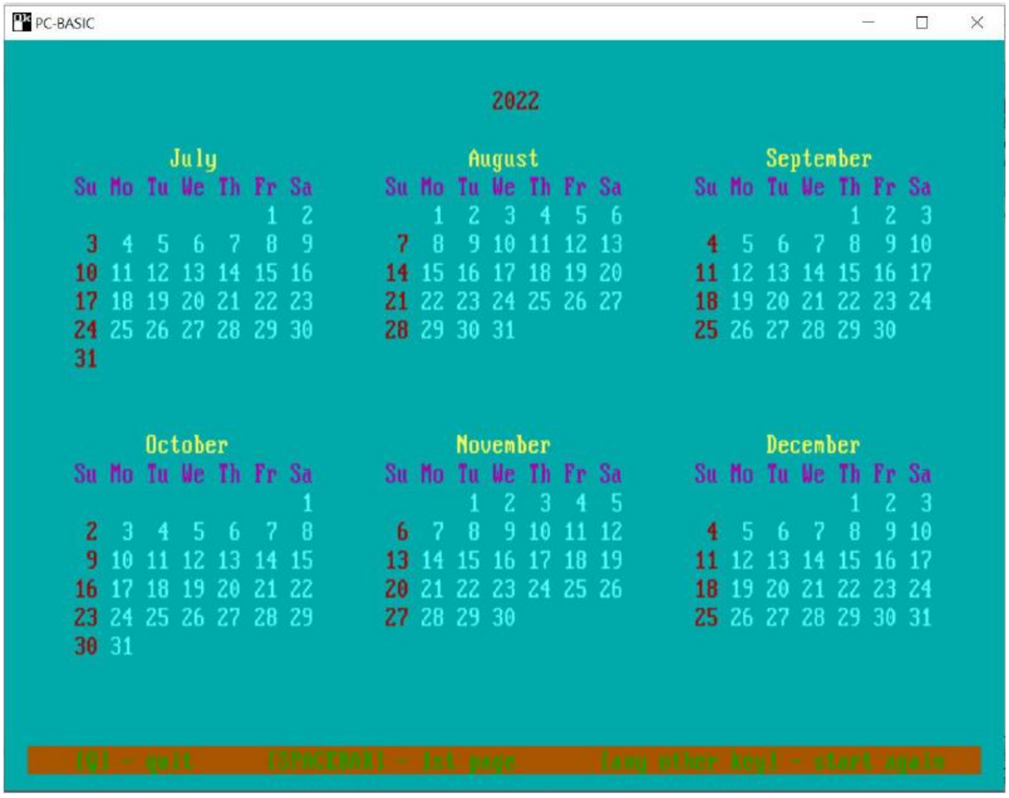


```
                                 Gregorian Calendar
                                    of any month

                               Month (1..12)     :
                               Year (since 1582) : 2022_
```

[Enter]



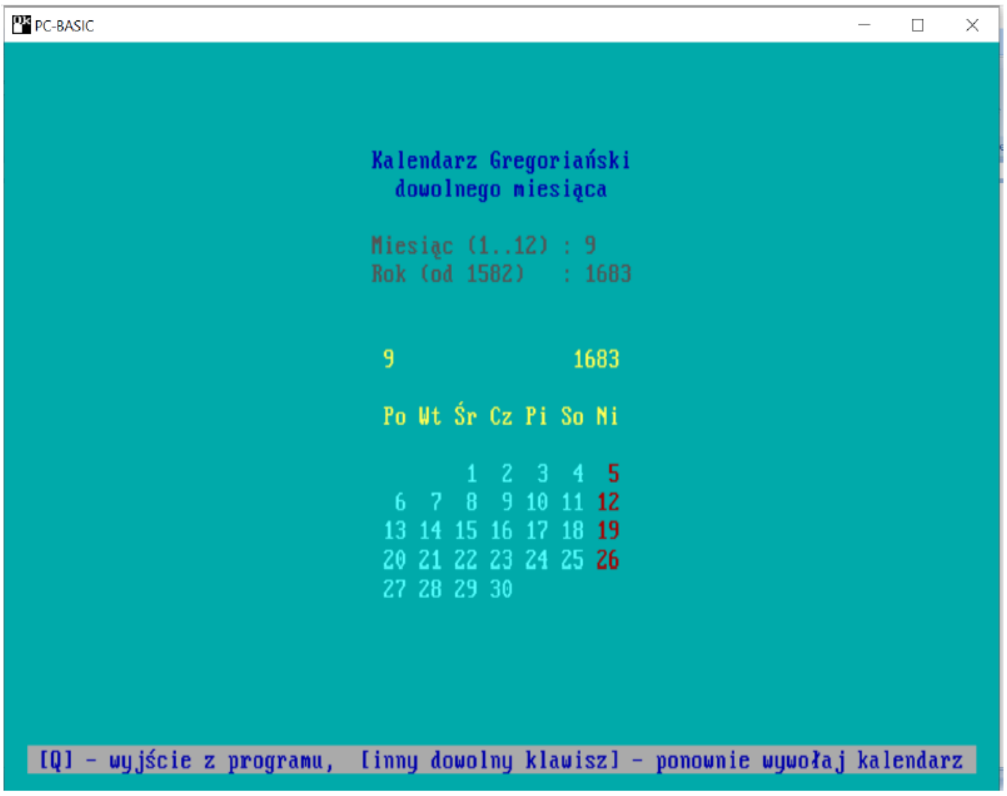
```
                                         2022

            January                   February                   March
      Su Mo Tu We Th Fr Sa      Su Mo Tu We Th Fr Sa      Su Mo Tu We Th Fr Sa
                         1             1  2  3  4  5             1  2  3  4  5
       2  3  4  5  6  7  8       6  7  8  9 10 11 12       6  7  8  9 10 11 12
       9 10 11 12 13 14 15      13 14 15 16 17 18 19      13 14 15 16 17 18 19
      16 17 18 19 20 21 22      20 21 22 23 24 25 26      20 21 22 23 24 25 26
      23 24 25 26 27 28 29      27 28                     27 28 29 30 31
      30 31


             April                       May                        June
      Su Mo Tu We Th Fr Sa      Su Mo Tu We Th Fr Sa      Su Mo Tu We Th Fr Sa
                      1  2       1  2  3  4  5  6  7                1  2  3  4
       3  4  5  6  7  8  9       8  9 10 11 12 13 14       5  6  7  8  9 10 11
      10 11 12 13 14 15 16      15 16 17 18 19 20 21      12 13 14 15 16 17 18
      17 18 19 20 21 22 23      22 23 24 25 26 27 28      19 20 21 22 23 24 25
      24 25 26 27 28 29 30      29 30 31                  26 27 28 29 30



                                Press any key to continue
```



```
                                         2022

              July                     August                   September
      Su Mo Tu We Th Fr Sa      Su Mo Tu We Th Fr Sa      Su Mo Tu We Th Fr Sa
                      1  2          1  2  3  4  5  6                   1  2  3
       3  4  5  6  7  8  9       7  8  9 10 11 12 13       4  5  6  7  8  9 10
      10 11 12 13 14 15 16      14 15 16 17 18 19 20      11 12 13 14 15 16 17
      17 18 19 20 21 22 23      21 22 23 24 25 26 27      18 19 20 21 22 23 24
      24 25 26 27 28 29 30      28 29 30 31               25 26 27 28 29 30
      31


            October                   November                  December
      Su Mo Tu We Th Fr Sa      Su Mo Tu We Th Fr Sa      Su Mo Tu We Th Fr Sa
                         1             1  2  3  4  5                   1  2  3
       2  3  4  5  6  7  8       6  7  8  9 10 11 12       4  5  6  7  8  9 10
       9 10 11 12 13 14 15      13 14 15 16 17 18 19      11 12 13 14 15 16 17
      16 17 18 19 20 21 22      20 21 22 23 24 25 26      18 19 20 21 22 23 24
      23 24 25 26 27 28 29      27 28 29 30               25 26 27 28 29 30 31
      30 31



      [Q] - quit      [SPACEBAR] - 1st page        [any other key] - start again
```

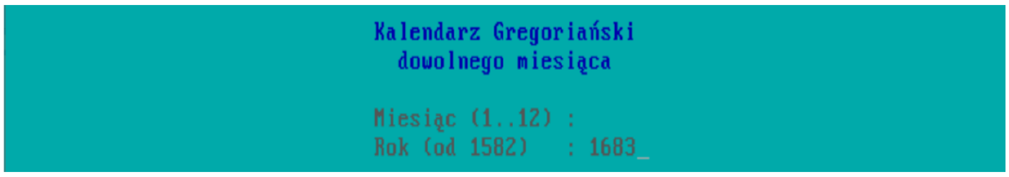
An example of how the program Kalen_pl works

Check with the program day of the week King John III Sobieski attacked the Ottomans near Vienna on September 12, 1683. Sobieski, without a rest, after a strenuous march of his troops, fought a battle on a Christian holy day.



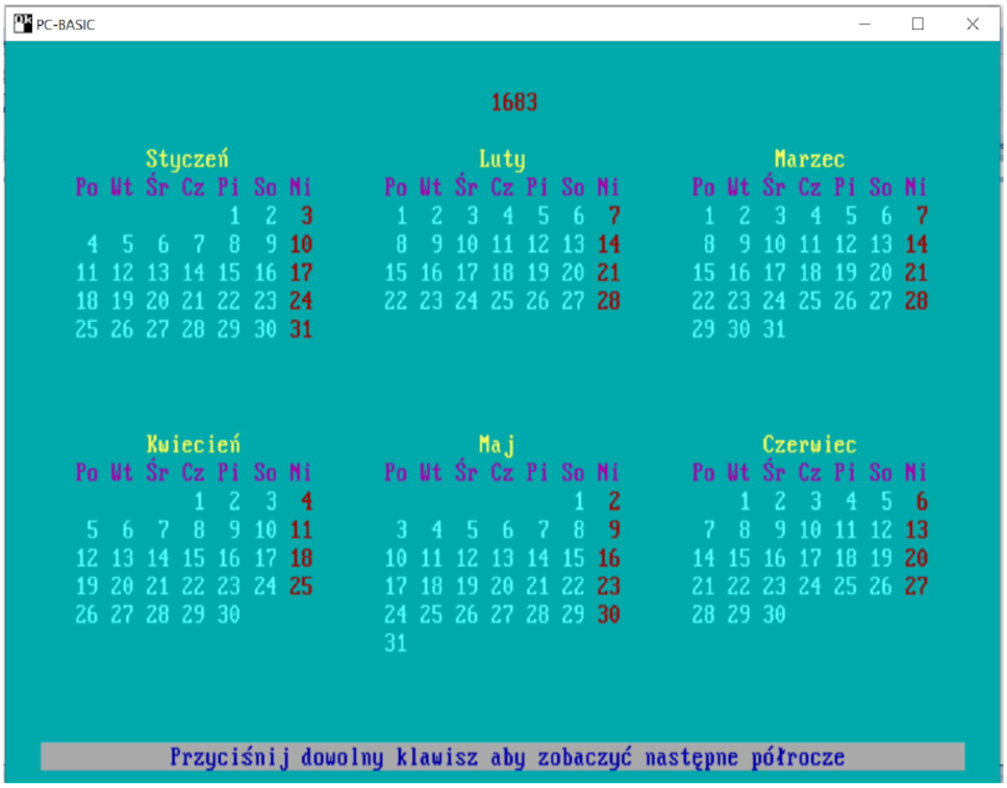
```
PC-BASIC

                    Kalendarz Gregoriański
                      dowolnego miesiąca

                    Miesiąc (1..12) : 9
                    Rok (od 1582)   : 1683


                     9               1683

                     Po Wt Śr Cz Pi So Ni

                              1  2  3  4  5
                      6  7  8  9 10 11 12
                     13 14 15 16 17 18 19
                     20 21 22 23 24 25 26
                     27 28 29 30

 [Q] - wyjście z programu,  [inny dowolny klawisz] - ponownie wywołaj kalendarz
```

or the entire year 1683, when we skip (ignore) the month by pressing [Enter]:

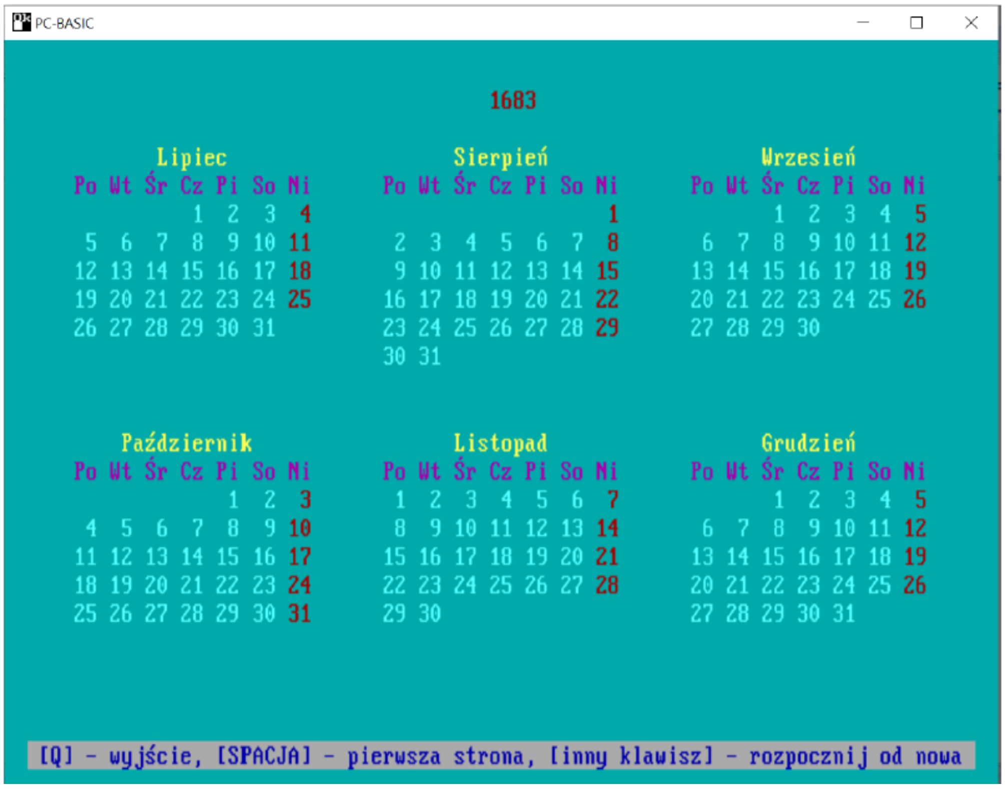


```
                    Kalendarz Gregoriański
                      dowolnego miesiąca

                    Miesiąc (1..12) :
                    Rok (od 1582)   : 1683_
```

Result:



```
PC-BASIC

                             1683

       Styczeń                   Luty                    Marzec
 Po Wt Śr Cz Pi So Ni    Po Wt Śr Cz Pi So Ni    Po Wt Śr Cz Pi So Ni
                1  2  3   1  2  3  4  5  6  7     1  2  3  4  5  6  7
  4  5  6  7  8  9 10     8  9 10 11 12 13 14     8  9 10 11 12 13 14
 11 12 13 14 15 16 17    15 16 17 18 19 20 21    15 16 17 18 19 20 21
 18 19 20 21 22 23 24    22 23 24 25 26 27 28    22 23 24 25 26 27 28
 25 26 27 28 29 30 31                            29 30 31


       Kwiecień                  Maj                     Czerwiec
 Po Wt Śr Cz Pi So Ni    Po Wt Śr Cz Pi So Ni    Po Wt Śr Cz Pi So Ni
             1  2  3  4                    1  2      1  2  3  4  5  6
  5  6  7  8  9 10 11     3  4  5  6  7  8  9     7  8  9 10 11 12 13
 12 13 14 15 16 17 18    10 11 12 13 14 15 16    14 15 16 17 18 19 20
 19 20 21 22 23 24 25    17 18 19 20 21 22 23    21 22 23 24 25 26 27
 26 27 28 29 30          24 25 26 27 28 29 30    28 29 30
                         31

       Przyciśnij dowolny klawisz aby zobaczyć następne półrocze
```



```
PC-BASIC

                                   1683

        Lipiec                    Sierpień                  Wrzesień
 Po Wt Śr Cz Pi So Ni      Po Wt Śr Cz Pi So Ni      Po Wt Śr Cz Pi So Ni
           1  2  3  4                         1             1  2  3  4  5
  5  6  7  8  9 10 11       2  3  4  5  6  7  8       6  7  8  9 10 11 12
 12 13 14 15 16 17 18       9 10 11 12 13 14 15      13 14 15 16 17 18 19
 19 20 21 22 23 24 25      16 17 18 19 20 21 22      20 21 22 23 24 25 26
 26 27 28 29 30 31         23 24 25 26 27 28 29      27 28 29 30
                           30 31

      Październik                 Listopad                  Grudzień
 Po Wt Śr Cz Pi So Ni      Po Wt Śr Cz Pi So Ni      Po Wt Śr Cz Pi So Ni
              1  2  3       1  2  3  4  5  6  7             1  2  3  4  5
  4  5  6  7  8  9 10       8  9 10 11 12 13 14       6  7  8  9 10 11 12
 11 12 13 14 15 16 17      15 16 17 18 19 20 21      13 14 15 16 17 18 19
 18 19 20 21 22 23 24      22 23 24 25 26 27 28      20 21 22 23 24 25 26
 25 26 27 28 29 30 31      29 30                     27 28 29 30 31


 [Q] - wyjście, [SPACJA] - pierwsza strona, [inny klawisz] - rozpocznij od nowa
```

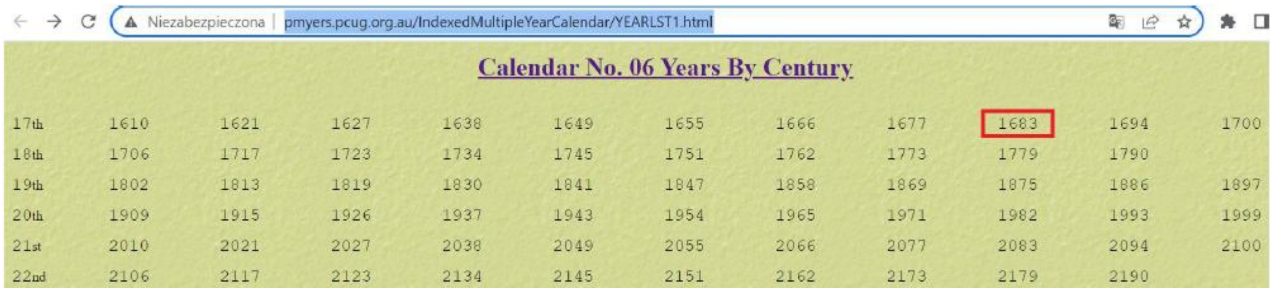
Description of the most important program variables and verification of the work of the program.

First of all: **Verifying of the data on the printout:**

Internet website:

http://pmyers.pcug.org.au/IndexedMultipleYearCalendar/YEARLST1.html

<-- Complete Calendars for years 1601 to 2200 inclusive. By Century within Calendar Number



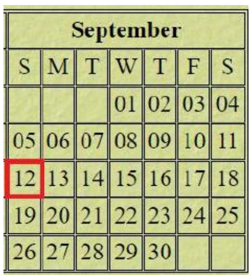
Niezabezpieczona | pmyers.pcug.org.au/IndexedMultipleYearCalendar/YEARLST1.html

Calendar No. 06 Years By Century

17th	1610	1621	1627	1638	1649	1655	1666	1677	1683	1694	1700
18th	1706	1717	1723	1734	1745	1751	1762	1773	1779	1790	
19th	1802	1813	1819	1830	1841	1847	1858	1869	1875	1886	1897
20th	1909	1915	1926	1937	1943	1954	1965	1971	1982	1993	1999
21st	2010	2021	2027	2038	2049	2055	2066	2077	2083	2094	2100
22nd	2106	2117	2123	2134	2145	2151	2162	2173	2179	2190	

All these years have the same calendar. I click on the "**Calendar No. 06 Years By Century**" subpage.

'Subpage' http://pmyers.pcug.org.au/IndexedMultipleYearCalendar/Calendar06.html



September

S	M	T	W	T	F	S
			01	02	03	04
05	06	07	08	09	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

...what fits (**S** after Sunday - the first English day of the week) our calendar.

Description of the most important program variables

There are 14 'subpages' - seven for leap years and seven for non-leap years. So, there are no more than 14 versions of the calendar.
Why seven versions? Because there are seven days a week and any year can start on any day of the week.

So, to build a calendar of a specific year, we only need to specify two things:
- is this year a leap year or not. As a result, this determines the set of digits of the string **A$**,
- which day of the week the year begins (the program can specify the day of the week at the beginning of any month). Finally, after some calculations, it is the variable **W**.

I have chosen some selected years for an additional verification of the **Kalen_en** program, where each year starts on a different day of a week:

Not leap years	**Leap years**
01-01-2023 --> Sunday	**01-01-2012 --> Sunday**
01-01-2018 --> Monday	**01-01-1996 --> Monday**
01-01-2019 --> Tuesday	**01-01-2008 --> Tuesday**
01-01-2014 --> Wednesday	**01-01-2020 --> Wednesday**
01-01-2015 --> Thursday	**01-01-2004 --> Thursday**
01-01-2021 --> Friday	**01-01-2016 --> Friday**
01-01-2022 --> Saturday	**01-01-2000 --> Saturday**

And more detailed, for each month of the year 2000:

Year Y	Month M	What year (leap/not leap)?	M	W1	val(A$,M,1)	W=W1+val	Final W
2000	1	Leap A$="**0**340250361462"	1	2484	**0**	2484	6 Sa (0 --> Su)
2000	2	Leap A$="0**3**40250361462"	2	2484	**3**	2487	2 Tu
2000	3	Leap A$="03**4**0250361462"	3	2484	**4**	2488	3 We
2000	4	Leap A$="034**0**250361462"	4	2484	**0**	2484	6 Sa
2000	5	Leap A$="0340**2**50361462"	5	2484	**2**	2486	1 Mo
2000	6	Leap A$="03402**5**0361462"	6	2484	**5**	2489	4 Th
2000	7	Leap A$="034025**0**361462"	7	2484	**0**	2484	6 Sa
2000	8	Leap A$="0340250**3**61462"	8	2484	**3**	2487	2 Tu
2000	9	Leap A$="03402503**6**1462"	9	2484	**6**	2490	5 Fr
2000	10	Leap A$="034025036**1**462"	10	2484	**1**	2485	0 Su
2000	11	Leap A$="0340250361**4**62"	11	2484	**4**	2488	3 We
2000	12	Leap A$="03402503614**6**2"	12	2484	**6**	2490	5 Fr
2000		Leap A$="034025036146**2**"					

The 13th digit was not used here. It turns out that its lack can lead to distortions. For example, the last day of December 1996 would be the 29th day. After all, the numbers in the sequence A$ are based on empirical values!

How to change the English calendar into another one in few seconds?

What are the differences between the code of the English calendar (here the first day of the week is Sunday) from another one, where the first day of the week is Monday? Apart from the language differences, of course!

Only adding line 265 and replacing W = 0 with W = 6 (to see Sunday on red) changes the English calendar into another one (here: Polish). These changes are shown in **bold** below.

Kalen_pl

```
        ...
        250 IF VAL(MID$(A$,M,1))>VAL(MID$(A$,M+1,1)) THEN B=7
        260 W=W1+VAL(MID$(A$,M,1))
        265 W=W+6                    ' For the Polish calendar to have a Monday as a first day of each week
```

```
270 W=W-7*INT(W/7)                          ' W = (0..6) means from Monday (0) to Sunday (6)
...
440 IF W=6 THEN COLOR 4 ELSE COLOR 11       ' Sunday on red (COLOR 4)
...
```

Kalen_en

```
...
250 IF VAL(MID$(A$,M,1))>VAL(MID$(A$,M+1,1)) THEN B=7
260 W=W1+VAL(MID$(A$,M,1))
270 W=W-7*INT(W/7)                          ' W = (0..6) means from Sunday (0) to Saturday (6)
...
440 IF W=0 THEN COLOR 4 ELSE COLOR 11       ' Sunday on red (COLOR 4)
...
```

In addition to the A$ sequence, following variables deserve attention:

- Y <-- year
- M <-- month
- W <-- number of the day of a week
    - from Sunday (0) to Saturday (6) - English calendar
    - from Monday (0) to Sunday (6) - non-English calendar
- L <-- number of days in a month

You can make this program more personal by marking holidays and birthdays of your loved ones. I did it and found out that ... it's not a deal to do that since there are no less than a hundred, unique, new programs to write.

Final remarks

1. Jerzy Wyrozumski, *Historia Polski do roku 1505 (History of Poland to year 1505)*, PWN, Warszawa 1983, ISBN 83-01-03732-6, page 41: "… the medieval custom was to start the week on Sunday, not on Monday, …".

2. In the Muslim countries the calendar may look completely different from the versions presented here (Polish and English).

3. Which week can be considered the first week of the year? It depends on a country!

4. Christophe Galfard, *The Universe in Your Hand*, Published in Poland as *Wszechświat w twojej dłoni*, OTWARTE, Kraków 2017, ISBN 978-83-7515-456-6, page 30, at the very end of the chapter 3:
" Scientists agree that the Sun will explode in 5 000 000 000 years, on Thursday, plus or minus three days."