4. Series

There are many fans of series who follow their favourite series continuously. A fan of series in English language took notes of the favourite series in an 8-month period.

File `list.txt` contains the date of broadcasting, the English title, the season and the episode number, the duration of the episode in minutes and a flag whether the creator of the list has already watched the given episode for the series liked by the fan. These data are given one after the other, in separate lines. The file contains the data of fewer than 400 episodes, there are 5 lines per episode.

Example:

```
…
2018.01.19
Puzzles
3x10
43
0
NI
Puzzles
3x11
43
0
…
```

The example shows that episode 10 of season 3 of series Puzzles was broadcasted on 2018. 01. 19. The duration of the episode is 43 minutes, and it has not been watched by the creator of the list yet.

- Dates were always recorded in “yyyy.mm.dd” format. There are episodes where the date of broadcasting was not known yet when the list was created. For these episodes instead of the date abbreviation “***NI***” is given.
- The season number is given without leading zeros and the episode number is always recorded with two digits. The season number and the episode number are separated by an “***x***”.
- Each episode of a given series has the same duration.
- The last data of a given episode is either “***0***” or “***1***”. 1 indicates that the given episode has already been watched by the creator of the list, 0 indicates that he/she has not watched it yet.

Create a program to process the data in file `list.txt`. Save the source code of the program as `series`. (Upon the creation of the program you do not have to check the correctness or validity of the data given by the user, you can assume that the available data correspond to the description.)

Before displaying the result of exercises that require writing data on the screen, display the number of the exercise on the screen (for example `Exercise 2:`). If you request data from the user, display the nature of expected data on the screen. Displaying texts without accents is acceptable.

1. Read and store the contents of file `list.txt`.
2. Display the number of episodes in the file where the date of broadcasting was already known.
3. Determine the percentage of the episodes in the file that have already been watched by the creator of the list. Display the percent value on the screen according to the example, with two decimal digits.
4. Calculate the total time spent by the fan watching the episodes already watched. Display the result according to the example, in day, hour, minute format.
5. Request a date in “yyyy.mm.dd” format from the user. Determine which of the episodes broadcasted by that date have not been watched by the fan yet. Include the episodes that were broadcasted on the given date as well. For episodes that fulfil the condition display the season number, the episode number and the title of the series on the screen, separated by tabs, according to the example.

6. Create a function that determines the day of the week using the following algorithm. Let the name of the function be *Dayoftheweek*. After giving the year, the month and the day the function returns which day of the week correspond to the given date in textual format. (In the case of remainder division of integers `a` and `b` expression `a div b` gives the ratio and `a mod b` gives the remainder, for example `17 div 7 = 2` and `17 mod 7 = 3`.)

```
Function Dayoftheweek(year, month, day : integer) : text
  days: Array(0..6: text)= ("Sun", "Mon", "Tue", "Wed",
                            "Thu", "Fri", "Sat")
  months: Array(0..11: integer)= (0, 3, 2, 5, 0, 3, 5, 1, 4, 6, 2, 4)
  If month < 3 then year := year -1
  dayoftheweek:= days[(year + year div 4 – year div 100 +
                  year div 400 + months[month-1] + day) mod 7]
End of Function
```

7. Request the name of a day in the abbreviated form given in the previous example. Give the days in the standard three-letter form, the first being a capital letter. Determine which of the series given in the file was broadcasted on the given day of the week. Display the names of the series on the screen according to the example. If on the given day of the week no series was broadcasted, then display message "No series was broadcasted on the given day."

8. Determine the total broadcasting time and the number of episodes for each series. In the calculation also take episodes without a known broadcasting date into account. In the solution you can use the fact that the data of the episodes of a given series are after each other in the source file. Write the data into file *sum.txt*. One line of the file should contain the title of the series, the total broadcasting time for the given series in minutes and the number of episodes separated by spaces.

45 marks

Example for the format of the textual output:

```
Exercise 2:
The list contains 202 episodes with known broadcasting date.

Exercise 3:
The fan has watched 45.66% of the episodes in the list.

Exercise 4:
The fan spent 2 days 15 hours and 32 minutes watching series.

Exercise 5:
Enter a date. Date= 2017.10.18
7x01    The Fable
7x02    The Fable
15x04   Military Police
5x03    Spy School
5x04    Spy School
4x04    The Elite Minds

Exercise 7:
Enter a day of the week (e.g. Thu)! Day= Thu
The Hospital
Spectacular Power
Upper Story
Chicago Flame
Shrinktime
```

Example for the format of file *sum.txt*:

```
Games 420 7
The Fable 588 14
The IT Guy 450 10
```