Homework 4 (File processing)

Select and solve **5** tasks from this list (exercises 04.05 and 04.06 are mandatory).

Include the solutions to tasks 04.01–04.04 in **one** R script file, see `homework4_template.R` for a template. For 04.05 are 04.06, I'll need 2 .Rmd files (knitr source files) and 2 .pdf files (use the *Knit PDF* button, the results). When you're done, send the 5 files via courses.ipipan.edu.pl.

Note that all the source files will be examined by plagiarism detection software.

Exercise 04.01. Write a function `CSVToCSV2()` to substitute CSV file field separators „`,`" for „`;`" and decimal separators „`.`" for „`,`". Parameters:

- `infname` – single string, input file's path,
- `outfname` – single string, output file's path.

Change neither "`,`" nor "`.`" within character fields. Moreover, do not call `read.table()`, `write.table()` and any of their variants.

Exemplary input – contents of some `infname` file:

```
write.csv(data.frame(a=c('"test".1,1', '"test".2,2'), b=c(0.1, 0.2)),
   file="/tmp/test.csv", row.names=FALSE)
cat(readLines("/tmp/test.csv"), sep="\n")
## "a","b"
## """test"".1,1",0.1
## """test"".2,2",0.2
```

Desired output – contents of some `outfname` file:

```
## "a";"b"
## """test"".1,1";0,1
## """test"".2,2";0,2
```

Exercise 04.02. Write a function `template()` with the following arguments:

- `dirname` – a single string with a directory name,
- `data` – a data frame with two columns, `key` and `value`.

The function should process each `.tpl` file in `dirname` (for testing purposes, create a bunch of text files with the `.tpl` extension) according to the following rules:

1. Detect all occurrences of strings of the form `%name%`, where `name` is a syntactically valid R name (cf. `?make.names`).
2. If a `name` matches some `key` in the `data` data frame, substitute `%name%` for its corresponding `value`.
3. If a `%name%` is matches of the following, substitute it with a given value:
   - `%filename%` – base name of the file being processed,
   - `%filepath%` – full path to the file;
   - `%curtime%` – current time (hh:mm:ss);
   - `%curdate%` – current date (yyyy-mm-dd).
4. For any other `%name%`s, generate a warning and leave them as-is.
5. Store the result in a `.txt` file (`somename.tpl` → `somename.txt`).

Exercise 04.03. Implement a function `BibTeX2data.frame()` to convert a given BibTEX file (argument `filename`) to a data frame. Each row in the resulting data frame represents a BibTEX entry. Columns:

1. entry type (e.g. `article`, `book`, ...),

HUMAN CAPITAL NATIONAL COHESION STRATEGY
EUROPEAN UNION EUROPEAN SOCIAL FUND
Project co-financed by the European Union under the European Social Fund

2. entry id (e.g.. `Smith2015:howtoteachnaughtystudents`),

3,4,… additional fields extracted from the file (of the `field=value` form, e.g. `journal`, `authors`, `pages`, …). Insert `NA` for missing field values (e.g. `journal` is often undefined in case of `books`)

An exemplary BibTEX file:

```
@article{paper1,
   author={J. Kowalski and Y. Wu},
   journal={Journal of Interesting Issues},
   title={P = NP},
   year={1999}
}

@book{xyz,
   author="G. Schmidt",
   publisher="PWN",
   year={2013},
   title={A general theory of everything}
}
```

Desired output:

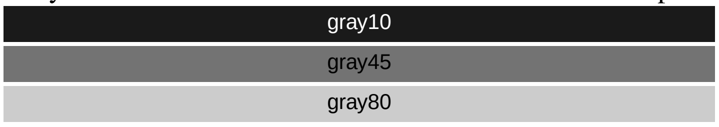
```
     type     id                 author                       journal
1 article paper1  J. Kowalski and Y. Wu Journal of Interesting Issues
2    book    xyz             G. Schmidt                          <NA>

                          title year publisher
                         P = NP 1999      <NA>
 A general theory of everything 2013       PWN
```

Exercise 04.04. Given an URL to a HTML file, e.g. http://www.example.com/something/x.html, write a function `webbot()` to extract all the HTML links of the form `<a ... href="address2extract" ...>` (consult a HTML language specification for reference). Download all the documents under the links extracted if they are located on the same domain as the starting document (http://www.example.com in our example). Then apply the same procedure recursively to all the downloaded files. Stop the process if there are no more files to download of if you have already fetched `limit` documents (by default 100).

In other words, implement a website copier (like e.g. HTTrack) which downloads all the documents reachable (perhaps with more than 1 mouse click) from a given URL and such that they are located on the same domain.

Exercise 04.05 (mandatory). Generate a knitr/RMarkdown report that illustrates how each of the colors returned by the `colors()` function looks like. Here's an excerpt of the desired output:



For each element in the vector returned by `colors()`, draw a rectangle filled with the corresponding color. Include text labels with colors' names – they should be readable, so don't typeset them all in black.

Hint: see `?plot.new`, `?plot.window`, `?rect`, `?text`, and `?col2rgb`.

Exercise 04.06 (mandatory). Generate a knitr/RMarkdown report that automatically performs a preliminary exploratory data analysis of all the variables in a data frame.

The data frame of concern is given at the very beginning of the report, for example:

```r
input <- MASS::Cars93 # must work with any data frame
```

Each variable (column) should be analyzed in a separate section of the resulting report. For factor and character variables, draw a pie chart, a bar plot, and output a textual table with all the variable's levels, counts, and percentages.

For numeric variables, draw a histogram, a box-and-whisker plot, and output a textual table with basic descriptive statistics, like mean, notable quantiles, standard deviation, etc.

Variables of other types (e.g. date/time) may be ignored.

HUMAN CAPITAL
NATIONAL COHESION STRATEGY

EUROPEAN UNION
EUROPEAN
SOCIAL FUND

Project co-financed by the European Union under the European Social Fund