Parsing the KNMI data (part 1)

Introduction

In this exercise you will build a program that can load the KNMI data into Python. The goal is to learn / practice the following:

- How to open a file and read the lines in it.
- How to split a string.
- How to convert strings to actual numbers.
- How to deal with nested lists.
- What `if __name__ == '__main__'` does.
- Understand that programming problems can be broken down into smaller more easily solved problems.

Assignment

1. Take the first 500 lines from the full data set you downloaded last week and save them to a file. Make sure that you include the header as well.
2. Start your favorite text editor and begin a new Python file called knmi-1-studentno.py (with the studentno being your student number).
3. Look up how you can access a file line-by-line. Find out how you can split a string on a certain character. Using this write a simple program that opens the small data file, skips the header, reads every line, splits it on an appropriate character and turns the strings into numbers.
4. How do you deal with missing data and why? What type of numbers can you use for the data in this data file? (Add to the report.)
5. Re-factor your program so that it consists of a function `def read_data(filename):` that returns the list of rows of numbers.
6. Look `if __name__ == '__main__'` up in the Python.org documentation, try to understand what it does and add it to your program. Feel free to ask one of the teachers if you cannot get this to work.
7. The header consists of 3 parts. Separate these out into several files that we will use in building a programs to read the header.
8. Build a program that can read the part of the header where the stations and their locations are defined. The format of this data should be a list containing one list for each station. That list for each station should contain 5 entries (station number, longitude, latitude, altitude, its full name) in appropriate data types. Explain in you report why you chose the data types that you did.
9. Build a program that can read the part of the header that explains the meaning of the column names. The output of this program should be a dictionary mapping the column name to its description.

10. Build a program to read the line of column headers, its output should be a list of column names.
11. Refactor the 3 programs that deal with the header into 3 functions and make them work together to read the full header information.
12. Combine the header reading part of your programs with the part that reads the rest of the data.