# Peer-review of assignment 4 for *INF3331-Ina*

Marie Samuelsen, Git-repo INF3331-Marie, mariesam@student.matnat.uio.no
Herman Loennechen, Git-repo INF3331-Herman, hermanlo@student.matnat.uio.no
Andre Kåsen, Git-repo INF3331-Andre, andre.kaasen@gmail.com

Deadline: Tuesday, 13. October 2015, 23:59:59.

## 1 Feedback

- In general you seem to have understood the overall logic of the assignment and structured it pretty well.
- You have completed all the tasks in the assignment.
- The program is quite readable, but there are very longparts and a lot of repeated code
- Regular expressions are used appropriately. We have suggested some readability adjustments below.
- Functions and variables have been given sensible names.

## 2 Review

Python 2.7.6 on a Linux Mint 17.2 Rafaela was used for reviewing the code.

### Assignment 4.1: Retrieve web page

The first subtask is very well implemented. Both readable and intuitive.

### Assignment 4.2: Find link to location

Your code do not support wildcards. This is fairly easy implement.

```
1  if "*" in place: place = place.replace("*", ".*?")
```

### Assignment 4.3: Retrieve weather information

Regular expressions are notorious hard to read. A better solution would be to set the verbose flag and then comment every part of it, like this:

```
1  all_data_pattern = re.compile(r"""
2                                 (?x)                                          # Set the verbose flag
3                                 <time\sfrom=\"(?P<fromDate>\d+-\d+-\d+)       # Capture fromDate
4                                 T(?P<fromtime>\d+:\d+:\d+)                    # Capture fromTime
5                                 \"\sto=\"(?P<toDate>\d+-\d+-\d+)              # Capture toDate
6                                 T(?P<totime>\d+:\d+:\d+)                      # Capture toTime
7                                 \".*\n.*\n.*name=\"(?P<symbol>.*)             # Capture weathersummary
8                                 \"\sv.*\n.*\svalue=\"(?P<prec>\d+\.?\d?).*    # Capture amount of rain
9                                 \n.*\n.*\n.*mps=\"(?P<wind>\d+.\d).*          # Capture windspeed
10                                \n.*\svalue=\"(?P<temp>\d+)\".*               # Capture temprature
11                                """,
12                                re.U|re.M|re.I)
```

### Assignment 4.4: Buffer all internet activity

The buffer is in place and operates smoothly, but you might want to consider putting all the .txt-files in a seperate directory.

```
1  if not os.path.exits("DIRECTORY FOR .TXT-FILES"):
2          os.makedirs("DIRECTORY FOR .TXT-FILES")
```

Instead of dumping the files in the local directory you can now do all the dumping in the directory you created when you first initialized the buffer.

## Assignment 4.5: Create weather forecast

The print_forecast function functions well. The only downside is that it is very long. For example, when you check whether selected_epoch is greater than current_epoch you could you could replace the whole if-else structure with one if block where you manipulate the variable d, and then do all the necessary operations to get the forecast. Here follows a slightly rewritten part of your program.

```
d = datetime.datetime.now()

if not selected_epoch > current_epoch:
    d_now = datetime.datetime.now()
    d = d_now.replace(hour = int(hour), minute = int(minute))
    d += datetime.timedelta(days=1)

# Get all the necesary data
next_date = ("{}-{:0>2}-{:0>2}").format(d.year, d.month, d.day)
# Get the right format
next_date_format = ("{}-{:0>2}-{:0>2} {:0>2}:{:0>2}:00").format(d.year, d.month, d.day, d.hour, d.minute)
# Convert to epoch
next_date_epoch = int(time.mktime(time.strptime(next_date_format, pattern)))

i = 0
for entry in toHour_list:
    # for each iteration, calculate new epoch
    from_data = str("{} {}").format(next_date, fromHour_list[i])
    to_data = str("{} {}").format(next_date, toHour_list[i])
    epoch_from = int(time.mktime(time.strptime(from_data, pattern)))
    epoch_to = int(time.mktime(time.strptime(to_data, pattern)))

    # Need to get new selected epoch for next day
    if ((next_date == fromDate_list[i]) and (next_date_epoch >= epoch_from) and (next_date_epoch <= epoch_to)):
        print "\n-----------------------------"
        print "{} {} to {} {}".format(fromDate_list[i], fromHour_list[i], toDate_list[i], toHour_list[i])
        print "{}: {}, rain:{}mm, wind:{}mps, temp:{}deg c".format(place, str(symbol_list[i]), \
              str(prec_list[i]), str(wind_list[i]), str(temp_list[i]))
        print "-----------------------------"
        i += 1
```

## Assignment 4.6: Testing the code

All your tests work fine, however we can't find any docstring-tests. We suggest something like the following.

```
def get_xml_links(self, webpage, place):
        """
Returns a dictionary with the place as key and the urls as values.

Parameters
----------
webpage: string
        Input argument
place: string
        Input argument

Returns
-------
xml_link_dict: dictionary
        Dictionary with urls to the place

Example
-------
>>> from forecast import Weather
>>> i = Weather()
>>> print j.get_xml_links(j.get_webpage("http://fil.nrk.no/yr/viktigestader/noreg.txt"), "Nannestad")
{'Akershus-Nannestad-Nannestad~86366': 'http://www.yr.no/place/Norway/Akershus/Nannestad/Nannestad~86366/forecast.xml\r',
'Akershus-Nannestad-Nannestad': 'http://www.yr.no/place/Norway/Akershus/Nannestad/Nannestad/forecast.xml\r'}

        """
```

## Assignment 4.7: Extreme places in Norway

This task repeat a lot of the code you wrote in the previous tasks. It might be an idea to import the other forecast.py in extreme.py and then just call the needed functions in your extreme_weather function.

## 2.1 Advice and comments

A general remark is that you have used the "old-style" classes, i.e.

```
class Weather:
```

instead of

```
class Weather(object):
```

Furthermore, both ”forecast.py” and ”extreme.py” have the same class name which created some initial confusion on our part.

# 3 Estimated points

- -2 for Regex wildcard not properly implemented.
- -2 One to three tests not working/missing (no docstring-test)

Total **26 out of 30 points**