Operating Systems
Constructor University
Dr. Jürgen Schönwälder

Module: CO-562
Date: 2026-10-30
Due: 2026-11-06

Problem Sheet #9

Problem 9.1: *weather forecast in the terminal* (3+3+3+1 = 10 points)

Write a program that prints a weather forecast to the standard output.

```
$ ./weather

* Wednesday 2026-10-28

  00:00     10.1 C    76.0 %     0.0 mm     7.5 km/h
  01:00      9.4 C    79.0 %     0.0 mm     6.9 km/h
  02:00      8.9 C    82.0 %     0.0 mm     5.4 km/h
  03:00      8.3 C    85.0 %     0.0 mm     4.5 km/h
  04:00      7.4 C    90.0 %     0.0 mm     4.6 km/h
  05:00      6.9 C    91.0 %     0.0 mm     3.3 km/h
  06:00      7.3 C    89.0 %     0.0 mm     4.5 km/h
  07:00      8.7 C    91.0 %     0.0 mm     6.8 km/h
  08:00      9.4 C    92.0 %     0.0 mm     7.7 km/h
  09:00     10.3 C    92.0 %     0.0 mm     8.5 km/h
  10:00     10.9 C    92.0 %     0.1 mm     8.2 km/h
  11:00     11.3 C    91.0 %     0.2 mm     9.5 km/h
  12:00     11.8 C    89.0 %     0.0 mm    10.3 km/h
  13:00     12.1 C    87.0 %     0.0 mm     9.8 km/h
  14:00     12.1 C    86.0 %     0.0 mm     8.7 km/h
  15:00     12.6 C    82.0 %     0.0 mm     9.7 km/h
  16:00     12.6 C    79.0 %     0.0 mm    10.4 km/h
> 17:00     12.1 C    83.0 %     0.0 mm    10.0 km/h
  18:00     11.5 C    84.0 %     0.0 mm     7.8 km/h
  19:00     10.9 C    87.0 %     0.0 mm     6.4 km/h
  20:00     10.4 C    90.0 %     0.0 mm     6.6 km/h
  21:00     10.0 C    92.0 %     0.0 mm     6.4 km/h
  22:00      9.3 C    95.0 %     0.0 mm     5.8 km/h
  23:00      8.8 C    95.0 %     0.0 mm     5.2 km/h
```

Your program obtains the weather data from a web service. Since you do not want to implement a network protocol yourself (yet), you simply run a tool like `curl` to fetch weather data and you parse the output produced by curl. An example invocation of `curl` in your shell may look like this:

```
curl -s -v -v https://api.open-meteo.com/v1/forecast \
     --data-urlencode "latitude=52.52" \
     --data-urlencode "longitude=13.405" \
     --data-urlencode "timezone=Europe/Berlin" \
     --data-urlencode "forecast_days=1" \
     --data-urlencode "hourly=temperature_2m,relative_humidity_2m,precipitation,wind_speed_10m"
```

The request retrieves weather information for the given location and timezone with hourly data about temperature, humidity, precipitation, and wind. The data is returned as a JSON document. We use the JSON parsing library `json-c`.

Further information how to structure your implementation is provided in the following header file:

```
#ifndef _WEATHER_H
#define _WEATHER_H
```

```c
#include <json-c/json.h>
#include <time.h>

typedef struct {
    struct tm time;
    double temperature;
    double humidity;
    double precipitation;
    double wind_speed;
} weather_record_t;

typedef struct {
    size_t size;
    size_t capacity;
    weather_record_t *records;
} weather_t;

extern weather_t* weather_new(void);
extern weather_t* weather_new_from_meteo(void);
extern void weather_free(weather_t *weather);
extern void weather_reset(weather_t *weather);
extern void weather_append(weather_t *weather, weather_record_t *record);
extern void weather_print(weather_t *weather);

extern json_object* meteo_fetch_data(void);
extern weather_t* meteo_decode(json_object *data);

#endif
```

a) Implement the function `json_object* meteo_fetch_data(void)` in the file `meteo_fetch.c` that opens a pipe to `curl` running as a child process. The function returns the response obtained by `curl` as a pointer to a `json_object` or NULL on failure.

   Do not use `popen()` or `system()`. Do not use the `libcurl` C API.

b) Implement the function `weather_t* weather_from_meteo(json_object *data)` in the file `meteo_decode.c` that parses the JSON input and returns weather data.

c) Implement the remaining functions declared in `weather.h` in `weather.c`.

d) Implement a main program in the file `main.c` that uses the other functions to retrieve weather data and to print it to the standard output.

Make sure that your program handles error situations and is robust regarding malicious inputs. It is recommended that you test your program using `valgrind`, a tool that can identify a number of programming errors.