Test-cases for alarm clock

Compiling the program

```
gcc -std=gnu99 -Wformat-overflow=1 -o alarmclock src/lib.c src/alarmclock.c
```

Describe each test case: what do you do to test this case, what does the test do, what is the expected result?

Case 1
Create an alarm and check that the alarm shows with the *l* (list) command

To do the test:

1. Invoke the program with *./alarmclock.*
2. Call the *s* command.
3. Provide a valid date and time (in the future and right format, described in the program).
4. Call the *l* command and verify that there exists an alarm with the provided date and time.

What the test does:

- It will fork the process after a scheduled alarm.
- Then it will watch the pid.
- The *l* command will check if the length of active alarm pid is greater than 0.

Expected result:

- A list with the alarm you created, if it was a valid input.

Case 2
Create an alarm, then cancel the alarm and finally verify that the alarm does not longer exist

To do the test:

1. Invoke the program with *./alarmclock*
2. Call the *s* command
3. Provide a valid date and time (in the future and right format, described in the program).
4. Call the *c* command.
5. Provide the letter *y* for the alarms that should be canceled.
6. Call the *l* command and verify that the chosen alarm does now not exist.

What the test does:

- It will fork the process after a scheduled alarm.
- Then it will watch the pid.
- The *c* command will check for any active alarmpid, then ask if you want to cancel the alarm for the given time.
- On *y* command it will give a signal to kill the give alarmpid.
- The *l* command will check if the length of active alarm pid is greater than 0.

Expected result:

- The alarm you created is successfully canceled, “You have no active alarms” will be displayed.

Case 3
Create more than three alarms, this should get an error of too many alarms. Check the list for the three first alarms created.

To do the test:
1. Invoke the program with *./alarmclock.*
2. Call the *s* command.
3. Provide a valid date and time (in the future and right format, described in the program).
4. Repeat steps 2 and 3 three more times. (Four in total)
5. Call the *l* command and verify that the three alarms exist with the provided date and time.

What the test does:
- The test will fork the process after a scheduled alarm. x3
- Then it will watch the pid. x3
- The schedule will check if there are 3 or more alarms, if it is it will return “Cannot create more alarms”. The process will not be initiated.
- The *l* command will check if the length of active alarm pid is greater than 0.

Expected result:
- The three first alarms created will be listed.

Case 4
Creating an alarm with a time that is in the past, should return an error with invalid time. Then create a new alarm with a valid alarm time, and check if it goes off after the set time is passed.

To do the test:
1. Invoke the program with *./alarmclock*
2. Call the *s* command
3. Provide an invalid date and time (in the past and right format, described in the program).
4. Call the *s* command
5. Provide a valid date and time (in the future and right format, described in the program).
6. Wait for the time to pass, then check for a notification in your program.

What the test does:
- The schedule command will check if the input is higher than the time right now. It won’t create a process if it is in the past.
- The test will fork the process after a scheduled alarm
- Then it will watch the pid
- When the alarm time comes, it will send a signal from the pid to the program, then print “ALARM!” x3 and if possible play the alarm sound.

Expected result:
- Alarm goes off at the given valid input.