# Posix threads monitor exercise

For a C++ program you save your program with the extension .cpp to your program name. For a C++ program compile your program with the following command:

g++ –o yourprogram yourprogram.cpp –l pthread

for a C program you save your program with the extension .c to your program name and you just use gcc instead like this

gcc –o yourprogram headerfile.c yourprogram.c –l pthread

If you have no errors then you can execute your program with the following command: ./yourprogram

**Exercise: a screen monitor**

It is perfectly fine to implement the monitor below in C. Put the class interface in a .h file and the implementation including the member variables in a .c file, where you include the .h fil. Finally, in your main file you include your .h file. You will find an example of how to build an object in C in its'learning.

Make a monitor for protecting the screen. Thus make a class called Screen. It has (at least) 3 member functions write_string, write_number and new_line. You might also need the member function write_string_and_number with 2 parameters (why?). In the private part you declare a mutex, and you initialize it in the constructor. So if you declare screen as an object of the class Screen, then an example of use might be:

screen.write_string(" hello world ");

A different approach typically adopted for library classes is to declare the member functions of the Screen class static (write the keyword static in front of the function definitions and the mutex variable must also be declared static and initialized below the class using the value PTHREAD_MUTEX_INITIALIZER . See "Advanced Linux Programming" p.80. No constructor in this class is needed. Thus you do not declare an object of the class, but call the member functions directly like this: Screen::write_string(" hello world")