```cpp
// lift.cpp: Source file for utilities relating to the lift
// Copyright (C) 2017 Ethan Wells
//
// This program is free software: you can redistribute it and/or modify
// it under the terms of the GNU Lesser General Public License as published by
// the Free Software Foundation, either version 3 of the License, or
// (at your option) any later version.
//
// This program is distributed in the hope that it will be useful,
// but WITHOUT ANY WARRANTY; without even the implied warranty of
// MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
// GNU Lesser General Public License for more details.
//
// You should have received a copy of the GNU Lesser General Public License
// along with this program.  If not, see <http://www.gnu.org/licenses/>.

#include "../include/lift.hpp"

namespace lift {
  side_t left;
  side_t right;
  sensors::pot_t* sensor = &sensors::lift;

  void side_t::set(int power) {
    side_t::topM.set(power);
    side_t::midM.set(power);
    side_t::lowM.set(power);
  }

  void init(void) {
    left.topM    = motors::init(2, 1, .5, .8);
    left.midM    = motors::init(3, -1, .5, .8);
    left.lowM    = motors::init(4, 1, .5, .8);
    right.topM   = motors::init(7, -1, .5, .8);
    right.midM   = motors::init(8, 1, .5, .8);
    right.lowM   = motors::init(9, -1, .5, .8);
    left.sensor  = &sensors::lift;
    right.sensor = &sensors::lift;
  }

  void set(int power) {
    left.set(power);
    right.set(power);
  }

  void lock(void) {
```

```
    set(lockN);
  }

  void to(position pos, int int_pos, int tolerance) {
    if (int_pos == -1)
      int_pos = pos;
    do {
      set((int_pos > sensor->value() + tolerance ||
           int_pos < sensor->value() - tolerance)
              ? (sensor->value() - int_pos) * 1.5
              : (sensor->value() - int_pos));
      delay(15);
    } while (int_pos > sensor->value() + tolerance ||
             int_pos < sensor->value() - tolerance);
    lock();
  }

  void control(void) {
    int power = (joystick::digital(5, joystick::Up) * 127) +
                (joystick::digital(5, joystick::Down) * 127);
    power = (power == 0 && sensor->value() > threshold)
                ? lockN
                : ((sensor->value() < threshold) ? 0 : power);
    set(power);
  }
} // namespace lift
```