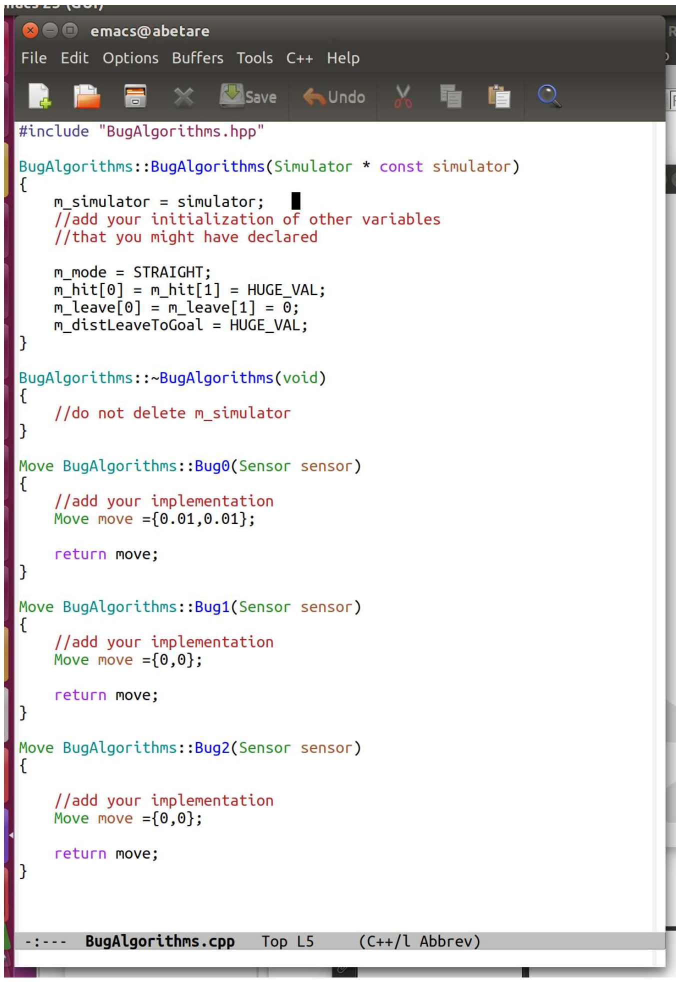
Supporting High-Complexity Coding

Amanda Shehu, Computer Science

CS 485: Autonomous Robotics

My main goal has been to give students assignments that do not feel like toy ones, but ones that indeed they can point to as accomplishments in their vitas when applying in industry or graduate school. However, such assignments demand time of students. To allow for complexity and yet feasibility, I now provide students with support code. Effectively, I implement the low-level behaviors, so that students can focus on implementing the high-level behaviors and see the concepts taught in class in action. The picture below shows how I provide snippets of the support code, highlighting where students are asked to add their implementations. This has allowed more complex assignments. Such assignments also give students a real and immediate purpose for their learning and focus their energies very effectively.



```
#include "BugAlgorithms.hpp"

BugAlgorithms::BugAlgorithms(Simulator * const simulator)
{
    m_simulator = simulator;
    //add your initialization of other variables
    //that you might have declared

    m_mode = STRAIGHT;
    m_hit[0] = m_hit[1] = HUGE_VAL;
    m_leave[0] = m_leave[1] = 0;
    m_distLeaveToGoal = HUGE_VAL;
}

BugAlgorithms::~BugAlgorithms(void)
{
    //do not delete m_simulator
}

Move BugAlgorithms::Bug0(Sensor sensor)
{
    //add your implementation
    Move move ={0.01,0.01};

    return move;
}

Move BugAlgorithms::Bug1(Sensor sensor)
{
    //add your implementation
    Move move ={0,0};

    return move;
}

Move BugAlgorithms::Bug2(Sensor sensor)
{

    //add your implementation
    Move move ={0,0};

    return move;
}

-:---  BugAlgorithms.cpp   Top L5     (C++/l Abbrev)
```