```
(runMouseRobot)
import java.awt.Robot;
robotHeader  % this initializes some simulation variables; don't mess with it
global mouse impulse_response;
global x_center_of_agario y_center_of_agario;
global upcoming_speed;
global COM_PORT;
% YOU MUST SET THIS TO THE CORRECT COM PORT FOR YOUR ARDUINO
COM_PORT = 'COM3'; % NOTE ALSO CHANGE IT IN LINE 83 BELOW

% YOU MUST CHOOSE VALUES FOR THESE NEXT TWO VARIABLES
% SO THAT THE START POSITION OF THE MOUSE POINTER
% IS RIGHT IN THE CENTER OF THE AGARIO SCREEN
% THESE ARE SET UP FOR PROF. REINKENSMEYER'S COMPUTER WITH TWO SCREENS
% AND WILL LIKELY NOT WORK WITH YOUR COMPUTER
% x_center_of_agario should be more like 1/2 the horizontal resolution
% of your screen, if you are using 1 screen
x_center_of_agario = 1920;
y_center_of_agario = 1100;


% TO CONTROL YOUR ROBOT CAR, YOU MUST DEFINE TWO THINGS
% 1: The steering controller
%       Do this by changing the function robot_steering_controller
%       Right now it's set-up as a P controller
% 2: The impulse response of your robot car, which you do next

% DEFINE THE IMPULSE RESPONSE OF YOUR ROBOT CAR
% This is where you create your impulse response for car velocity
% This specifies the velocity of the car as a function of sample #
% After the piston has fired
% As an example, this code creates an impulse response
% using the lognormal function, then plots it
numptsImpulseResponse = 1000;
upcoming_speed = zeros(1,numptsImpulseResponse);
pd = makedist('Lognormal','mu',log(100),'sigma',.5);
x = [1:numptsImpulseResponse]';
impulse_response = 15000*pdf(pd,x)';
figure(1); clf
subplot(211);
n = [1:numptsImpulseResponse];
plot(n,impulse_response);
xlabel('Sample #');
ylabel('Robot Car Velocity');
title('Response of Car to a Single Push of the Cylinder')
subplot(212);
threeimpulses = impulse_response+ ...
    [zeros(1,100) impulse_response(1:end-100)] + ...
    [zeros(1,200) impulse_response(1:end-200)];
plot(n,threeimpulses);
```

```
xlabel('Sample #');
ylabel('Robot Car Velocity');
title('Response of Car to a Three Pushes of the Cylinder 100 samples apart')


% THIS RUNS THE SIMULATION
% Don't change anything below here

disp('If you stop this program and the mouse control is still on')
disp('You can type clear in the command window to stop the mouse control');
disp('This works by clearing the callback function');
stopLoop_p = 0;
mouseControlOn_p = 1;
mouseSetUp_p = 1;

while(stopLoop_p == 0)
    if (mouseSetUp_p == 1)
        mouse = Robot;
        arduinoObj = serialport(COM_PORT,9600);
        configureTerminator(arduinoObj,"CR/LF");
        flush(arduinoObj);

        %Uncomment this is you want to store the Arduino data.
        %The Data field of the struct can be used to save the read values
        %and the Count field can be used to saves the sample number.
        %arduinoObj.UserData = struct("Data",[],"Count",1)

        %Set the BytesAvailableFcnMode property to "terminator" and
        %the BytesAvailableFcn property to @readAGDSimRobotCar.
        %The callback function readArduinoGamingDevice is triggered when data
        %(with the terminator) is available to be read from the Arduino.
        configureCallback(arduinoObj,"terminator",@readAGDSimRobotCar);

        mouseSetUp_p = 0;
    end;
    if mouseControlOn_p == 1
        x = input('Arduino mouse control ON (Hit return to stop, ctrl-c to quit)');
        clear;
        stopLoop_p = 0;
        mouseControlOn_p = 0;
        mouseSetUp_p = 0;
        COM_PORT = 'COM3';
    else
        x = input('Arduino mouse control OFF (Hit return to start, ctrl-c to quit)');
        stopLoop_p = 0;
        mouseControlOn_p = 1;
        mouseSetUp_p = 1;
    end
end
```

```
(robotSteeringController)
function steering_torque = robotSteeringController(theta, thetad)
KP = 5;  %This is the proportional gain (experiment with changing)
% Proportional feedback control law for the steering
% you can change the type of control law if you want
steering_torque = -KP*(theta-thetad);
end
```