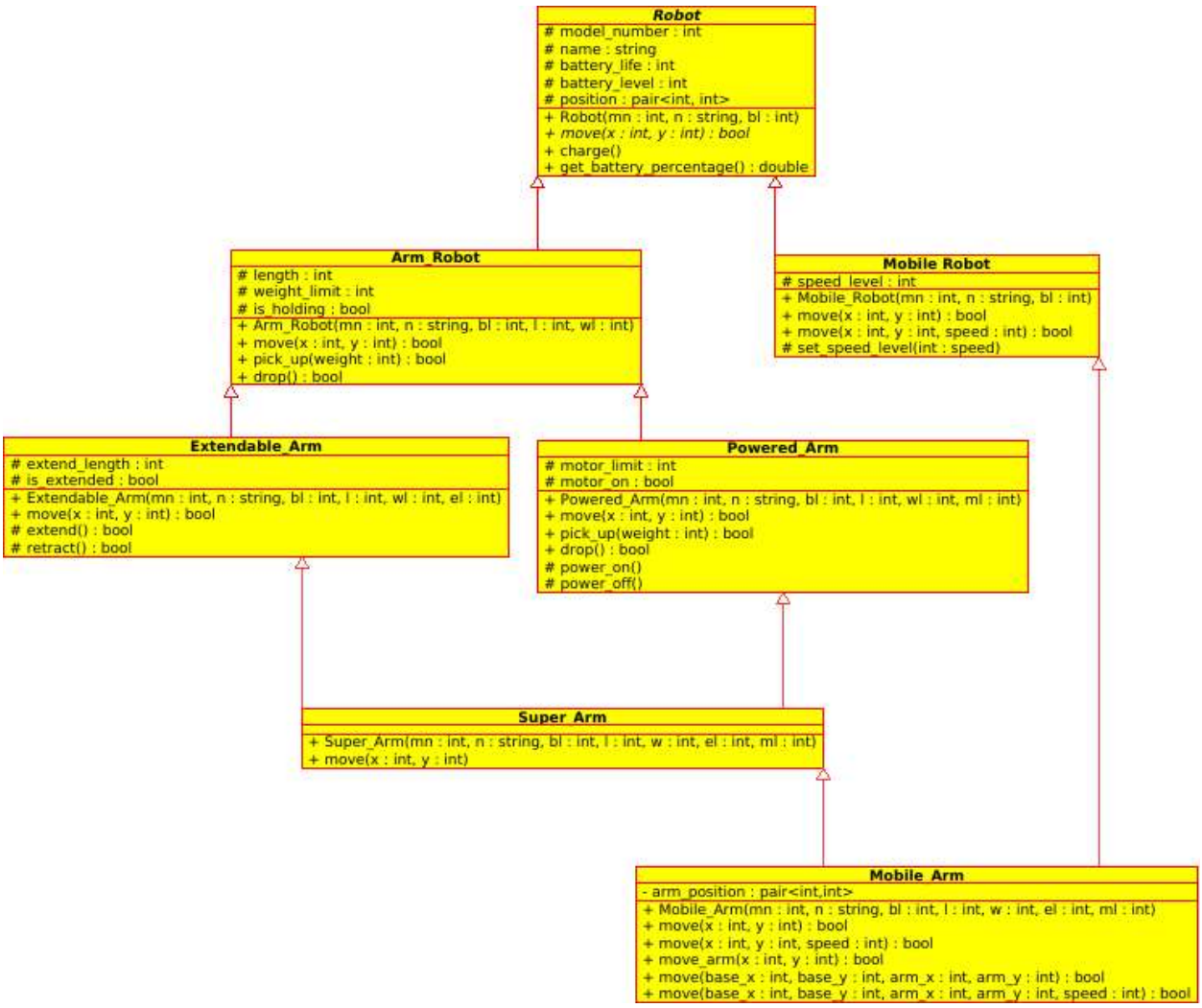
CSE 1325-001 Homework #7 – Multiple Inheritance and Polymorphism

In this homework assignment, you will be extending your Homework 6. You may use your existing HW6 solution, or the supplied HW6 sample solution (coming soon).

See the below UML Diagram



Part 1: Changes from HW6 UML that's not a new class.

- Robot
  - Robot is an Abstract Class, meaning there is at least one Abstract Function. An Abstract Function is a function that is declared but not defined. In C++, an abstract function is a pure virtual function.
  - move is now a pure virtual function.
- Mobile_Robot
  - Typo from HW6 has been corrected

    - speed_level and set_speed_level() are now protected
- Extendable_Arm
    - extended_length, is_extended, extend(), and retract() are now protected
- Powered_Arm
    - motor_limit, motor_on, power_on(), power_off() are now protected4

Part 2: Super_Arm

Super_Arm inherits from Extendable_Arm and Powered_Arm. There are no variables.

The Constructor calls the base class' constructer. Move will now take into account if it is extended and if the motor is on. So now it will take 1 battery unit to move 1 distance, 2 if moving and holding an object, 2 if moving extended, 3 if moving, holding an object and extended, 4 if h moving, holding an object and motor is on, 5 if moving, holding an object, motor is on, and arm is extended.

Part 3: Mobile_Arm

Mobile_Arm inherits from Super_Arm and Mobile_Robot. There is one new variable to help differentiate the different between the base's position (the mobile robot position) and the arm's position (the end of the arm). position is the base's position. arm_position is the arm's position.

The Constructor calls the base class' constructer. There are now 5 move functions

- move(int, int) moves the base to the new coordinates at the current speed level. It just calls Mobile_Robot's move(int, int) method.
- move(int, int, int) move the base to the new coordinates at the new speed level. It just calls Mobile_Robot's move(int, int, int) method.
- move_arm(int, int) move the arm to the new coordinates. It just calls Super_Arm's move method.
- move(int, int, int, int) moves both the base and the arm to the new coordinates at the current speed level. The UML shows which input variables go to which.
- move(int, int, int, int, int) moves both the base and the arm to the new coordinates at the new speed level. The UML show which input variables go to which.

Part 4: Main without Polymorphism

Part 5: Main with Polymorphism

You will make a second main file (abc1234_main_two.cpp). In this file, you will create a main function that contains a list of Robots. Make sure at least one of each robot is in the list. (insert more here)

For compiling this main, you have to use the same make file to compile from part 4. (Hint: you can type more than just make in terminal. Think back to HW 3.)

Bonus:

Something with enumeration here.

Deliverables: