Write the following phone app. Make sure to comment your code and that your name is in a comment line at the beginning of code. You will only have to write the client side and the server is provided for you.

**Battle Bot Arena Game**

The object is attempt to destroy the other bot(s), before they destroys you. You will write a client program to control a bot in the arena. The game’s timing is based on the number of messages received by the server from the client. When movement rate, number of shots and recharge rate are discussed below, it is based on the messages received by the server, but the server protocol will determine a regular the amount of time between messages.

**About the bot:**

Your client will control the movement and firing of the bot. You will also be able to scan the area around you bot for other bots and bullets. During the setup of your bot, you will be able to choose to add to the attributes of your bot. You have five “points” to use, but you don't have to use any or all 5 points.

1. The amount of armor your bot has is between 1 and 5. The more armor the more hits your bot can take, before it dies. But the more armor you have the slower your bot will move. Movement rate is based on this equation: $\text{MoveRate} = \text{Armor} - 1$. If you armor is 2, then the bot can move every other message and with an Armor of 5, it can move every 4 messages. A Note, Even if your armor is damaged by shooting and your bot is still alive, the bot's MoveRate **does not** change.

2. The power of the bullets your bot is firing is between 1 and 5. A bullet power of 1 which cause 1 point of damage, while 5 will inflict 5 points of damage. The more powerful, the less often you can fire and it will go a shorter distance. How often you can shoot is based on shot power * 10. The distance a bullet travels is based on the diagonal length of the arena divided by bullet power. So if the bullet power is 1, the bullet can travel the entire length of the arena. Last, shooting draws power from the drive engine to shoot, the movement will be affected, by the following equation: $\text{MoveCount} = \text{MoveCount} - \text{BulletPower}$. So when firing a BulletPower of 5, the bot will not be able to move for 5 messages and cannot shoot again for 50 messages.

3. Scan distance is between 1 and 5. Scan distance is the value times 100, so a value of 1, allows you to scan 100 pixels radius around the bot, while 5 allows you to scan a 500 radius around the bot. You can always scan and it does not affect shooting or movement.

**The default for all three attributes is 1 and then you may choose to add to it, with the 5 additional points.** You do not have to use all 5 points, but you cannot use more than a total of 5 points across the 3 attributes. Remember any given attribute can have at most 5, so the max you can add to any attribute is 4.

**Protocol to talk to the server**

**Setup stage:**
After the connection, the server sends you:
PID WidthArena HeighArena NumberOfBots
Example message: 1 250 250 2 //You are player 1, arena is 250x 250, and there are 2 players
Client sends back:
NameOfBot ArmourValue BulletValue ScanValue
Example message : Testbot 0 0 3 #note didn't use all 5 points

If clients sends back information, then the server sends "setup error" and info needs to be resent correctly otherwise sends back bot info
name ArmourValue MoveRate ScanDistance BulletPower RateOfFire BulletDistance
Example message: Testbot 1 400 110 707
Once the server returns the final information about your bot, you will wait until the first “status” message
Full example:
SERVER: setup 0 500 500 2
CLIENT: me 00 3
SERVER: me 1 0 400 110 707

Game stage:
Once clients receive the first status message the game has started and the server sends a status message every time the server is ready to accept a new action from the client.
status message:
Status X Y MoveCount ShotCount HP
example: Status 162 110 0 0 2
    at position 162,110 You can shoot and move, Armor value of 2
example: Status 162 110 0 -10 2
    at position 162,110 You can move. 10 messages until you can shoot. Armor value of 2

Once the Status message is read, the server is now ready to receive one command. The client can send 1 of 4 different messages: noop, move, fire, scan

1. Noop, do nothing, but MoveCount and ShotCount counts will increment if non-zero.
message: noop

2. Move, move 1 pixel in 1 of 8 directions, based on X and Y. If bot is able to move, then the bot will be moved 1 pixel based on the direction indicated by x and y value. ShotCount will increment if non-zero. If the bot cannot move, then MoveRate is decremented (as if you did move) but the bot will not move and ShotCount will remain unchanged.
message: move X Y
example message: move 0 -1 #move up 1 pixel
example message: move 1 1 #move down and to right 1 pixel
Note: X and Y can be any positive or negative number, but the bot only moves 1 pixel.

3. Fire, shoot a bullet based on a angle. If the bot is able to fire a bullet, then a bullet will be fired in the direction of the angle. Remember, MoveCount will be decremented (see above). If the bot is not able to fire, then ShotCount will be decremented (as if you did shoot) and MoveCount will remain unchanged.
message: shot ANGLE
example message: shot 0  #shoot a bullet straight up
example message: shot 270 #shot a bullet to the left

4. Scan, then server sends information about the bullets and bots within the scan distance, but it
doesn't return information about your bot or your bullets. ShotCount and MoveCount will
increment if non-zero.
message: scan

Return info from the server message
if there are bots within range the following message will be sent
scan bot PID X Y
if there are bullets within range the following message will be sent
scan shot PID SHOTID SHOTPOWER X Y
When the scan is done, the following will be sent.
scan done
eexample: nothing in range
scan done
eexample: 1 bot and 2 bullets in range
scan bot 2 150 75
scan shot 2 21 1 150 90
scan shot 2 22 1 150 80
scan done

Other information your bot may receive.
Before the Status message, they maybe Info messages about what has happened.
Info hit by PID SHOTID  #You've been shot by PID, see status message for armor
Info Dead      #Your Dead, game over NO Status message will follow
Info GameOver  #Game over, you have won. Same, no status message
                Both Once either Info Dead and GameOver is sent, the server closes the connection.
Info Alive NUMBER  # A bot has been killed, there are NUMBER left in play.

Example:
Info Alive 2  # There are now 2 bots alive (1 of them is you!).
Info hit by 2 22  #player 2's bullet hit you
Status 162 110 -1 -19 1
        You are at 162,100, you can't move for 1 message, can't shot for 19 messages and you have
        an armor value of 1

About how bots and bullets are drawn in the arena and collisions
    All bots are drawn as a 10x10 square. The position show by the status message is the upper
    left point. So if the bot status shows 150 150, the bot is drawn from 150,150 to 150,160, to 160,160
    and 160,150, and back to 150,150. Bullets are also drawn as a square and the position shown in
    the scan is the upper left point. The bullet power + 1 is the length of each side, so bullet power of 1
    is drawn as 2x2 square and a bullet of 5 is a 6x6 square.
    If a bot tries to move into a wall, then the move is prevented by the server (the message is
    ignored) and no damage is taken by the bot.
PROGRAM REQUIREMENTS:
You are to write an app that basically turns the phone into a joy stick. How you design the UI is up to you. But you app should able to read all the messages and send out all 4 messages as well. You UI gets user input and sends it to the server, plus deal with errors, like User tried to move, but they are not allowed to yet.

How to run the server:
The server is provided for you and 2 client bots. The server is expected to be run from the command line. It can take 1 command line argument. Command: java -jar BatBot.jar <Number of players> If you leave off number of players, it defaults to 1 player. This is a test mode, so you can test your bot. To exit the server, press q or ESC over the battle Arena window. The server listens on port 3012 for the clients.

You are also provided with a test bots to play against. To run the clients command: java -jar uinput.jar This is for you to test the protocol. The clients assume localhost, so you must run them on the same machine as the server.

Lastly, Master of the Battle Arena:
To encourage more interesting bots, There will be two challenges run during class time to determine the Master of the Battle Arena. First a pairwise challenge, which will determine the best bot in 1 to 1 challenge. Second will be a free for all, with all bots battling out in the same arena, to determine a champion. Ties (when the bots kill each other and there is no winner) will be replayed until there is a winner. The winner of each challenge will receive 5 extra credit points. If one bot should win both challenges, the student will be declared the “Master of the Battle Arena” and of course get both extra credit points. The Instructor may enter their bot as well, should they win no extra points will be awarded for the challenge. No students will be harmed in the battle, only their bots.

TURN IN and GRADING:
Hard copy:
1. A copy page with Name, program #5, and blackberry or Android, cosc 4730 or 5730 depending on which class you enrolled in.
2. Printed directions for how to play, which simulator is need to run this (include special information like screen size for the android)
3. Print the java code (Android users don’t need to print xml documents).

Soft copy:
1. Put an exported copy of the code in \\
   \lamont.cs.uwyo.edu\courses\cosc4730\students\<lastname>\program5\ the source code (project directory)
2. directions for how to play, which simulator is need to run this (include special information like screen size for the android)

Code will be graded on correctness, comments, and coding style.