**Lab 4**

In this lab you build a Behaviour algorithm in Excel. The Behaviour is being designed to control a theoretical Robot. Our Robot lives in a Euclidean world and can move to the coordinates that it is given. Movement in the Euclidean world is ideally suited to making use of vectors. Other Robots operate in this world also, and some are bad Robots. Your job will be to develop a simple algorithm which will identify where a Robot should move to if they are adopting a specific Behaviour.

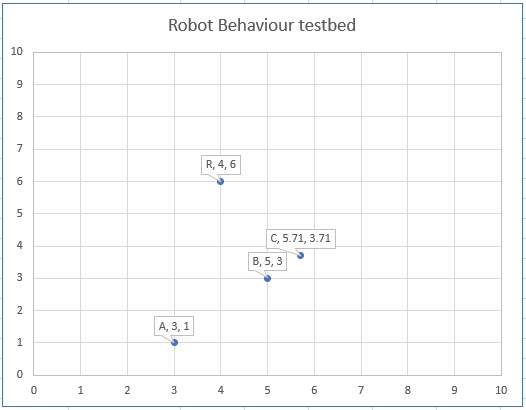
**Setup**

You will need to Excel.

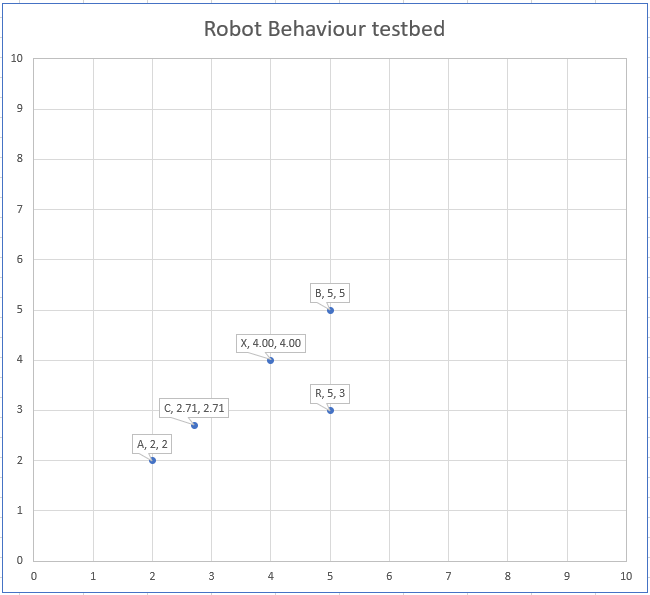
Download the Excel spreadsheet from Blackboard “absExample-lab.xlsx”.

**To do:**

1. There are two sheets. One called “Hide” and one called “Protect”. The Hide sheet implements a hide behaviour for our Robot (R). This behaviour is complete. You will implement the Protect behaviour.
2. In the Hide sheet there are 2 other robots (A and B). The coordinates of all our Robots are known, so we can plot them on a graph like in the figure below. We want to implement a “Hide” behaviour for our Robot. To be specific, we want our Robot R to hide from Robot A by moving to a point behind Robot B. We do this by calculating an appropriate point C in the world. That’s it. Our behaviour is complete and we would simply hand off the actual movement of R to its “Move” behaviour (if we had one).



1. Examine how the point C is calculated. Play with the coordinates of A, B and R to make sure the algorithm works consistently. The graph updates automatically. You can also adjust the value of “d” which is simply used to adjust how far away from B the point C should be.
2. Once you are happy with the Hide behaviour, move to the Protect sheet. You are to finish the implementation of the Protect behaviour. We want to make a brave Robot which will go to the defence of another. In particular we want our Robot R to protect Robot A from bad Robot B. So, in the figure below, R should take up a position between A and B. But where exactly?
3. Similar to the Hide behaviour **you are to calculate a point C** but this time just **in front** of A. That would seem to make good sense. I have hard coded in the coordinates for C as an example.
4. However, if the situation is critical, C may not be the closest defensive position for R to take up. In the case below, the position at X would be the best location to move to. In the spreadsheet I have hard coded in the coordinates for X as an example. **You are to calculate X** and then check to see whether C or X is closer. You should have a cell in the spreadsheet which states whether to go to X or C.



1. Notes: The point X is the projection of the point R onto the line AB. In Vector language this means the projection of vector AR onto vector AB. This can be found by using the dot product of AR with AB. Check this out for more info: <https://math.libretexts.org/Bookshelves/Applied_Mathematics/Mathematics_for_Game_Developers_(Burzynski)/02%3A_Vectors_In_Two_Dimensions/2.06%3A_The_Vector_Projection_of_One_Vector_onto_Another>