SYSC5103: Assignment 2, Question 1
Luke Newton (100999309)

Selection of Algorithm: For this assignment, I created a RoboCup player as an expert system. I think that writing rules using only AND may not be quite as expressive as some of the other options, but this is fine for a relatively simple environment like RoboCup and was preferred over the others as it is quite easy to represent the knowledge base in a way that is easily understood by both the program and myself.

Running the Reactive Agent: The "ExpertSystem" folder should replace the demo Krislet folder. The agent is executed identically to Krislet (i.e., "java Krislet"). An agent's behavior can be changed without recompiling the code by editing AgentSpec.txt.

Agent Input: The AgentSpec.txt file has four parts; each are described below (as well as in the file itself):

1. **Facts** are single terms (no spaces, but it may contain dashes or underscores) that represent things the agent knows. The existence of a term in the agent's fact list means that the fact is true. The word(s) in a term have no meaning to the agent; facts are simply pattern-matched with terms in the antecedent of each rule. As the agent evaluates rules, it adds information to its list of facts. Any facts included in the specification file are things that the agent immediately knows on startup.
2. **Rules** are used to deduce more facts that can be used in further rules, or to indicate the agent should do and action. Rules are of the form: <antecedent> -> <consequent>. The antecedent is one or more terms separated by AND, each of which can be either a fact, a visual check, or an auditory check. Visual checks take the form: <object> <attribute> <operator> <value>, where <object> is the name of an object on the field (e.g. "ball", one of the goals, any flags), <attribute> is one of {DISTANCE, DIRECTION, VISIBLE, NOTVISIBLE}, <operator> is one of {EQUAL, LESSTHAN, GREATERTHAN, LESSTHANEQUAL, GREATERTHANEQUAL, NOTEQUAL}, and <value> is a real number. For attributes VISIBLE and NOTVISIBLE, operator and value are not used. Auditory checks take the form: HEARD <event>, where <event> is any message broadcast by the server (e.g. "drop ball", "offside", "goal"). The consequent is either a fact, or an action of the form: do <action> <value>, where <action> is one of {turn, kick, dash} and <value> is a real number that specifies the power of that action.
3. **Transient Facts** are one way that an agent can forget facts. Any fact listed here will be removed from the agent's knowledge base after the agent performs an action, the remaining facts will be remembered for the next simulation step.
4. **Conflicting Facts** are another way an agent can forget. This section contains lists of comma-separated facts. Only one fact from each list can exist in the agent's knowledge base at any time. If a fact is to be added to the knowledge base which is a list here, all other facts in that list are first removed from the agent's knowledge base

Agent Algorithm and Output: The agent searches through its rules forwards in a breadth-first manner. It cycles through each rule in its knowledge base, adding facts, until either a rule directs the agent to do an action, or a complete cycle through all the rules has been performed and no new knowledge is gained. If no new knowledge can be gained, the agent does nothing until the next simulation step.

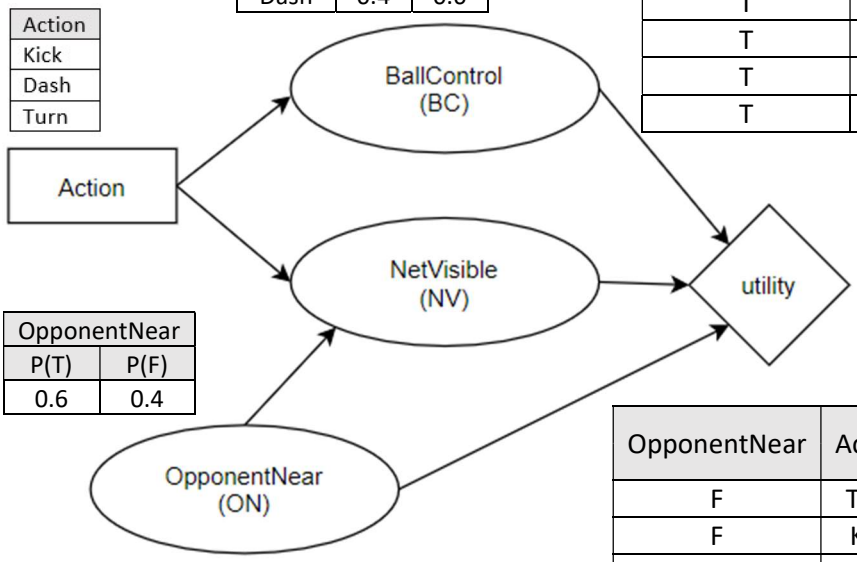
Expected Behavior: Agents following the provided specification file follow the ball around and attempt to kick it into whatever goal they first find. They do this by looking for the ball by turning, then running towards it, then searching for a goal by turning. It may be the case that the agent can no longer see the ball once it has turned to a goal, so it remembers that the ball was close the last time it was visible and backs up until both the ball and goal are in view. The agents also remember whether the ball was to their left or right the last time they saw it, hopefully making it easier to find the ball again.

SYSC5103: Assignment 2, Question 2
Luke Newton (100999309)

The simple decision network of a soccer-playing agent is shown below, with dependencies, probabilities, and assigned utilities:

OpponentNear	BallControl	NetVisible	Utility
F	F	F	0
F	F	T	1
F	T	F	4
F	T	T	5
T	F	F	0
T	F	T	1
T	T	F	3
T	T	T	4

Action	BallControl	
	P(T)	P(F)
Turn	0.15	0.85
Kick	0.9	0.1
Dash	0.4	0.6



OpponentNear
P(T)	P(F)
0.6	0.4

OpponentNear	Action	NetVisible	
		P(T)	P(F)
F	Turn	0.7	0.3
F	Kick	0.5	0.5
F	Dash	0.5	0.5
T	Turn	0.5	0.5
T	Kick	0.35	0.65
T	Dash	0.35	0.65

BallControl (BC) describes whether the agent is in control of the ball and is true (T), false (F). NetVisible is either true (T) or false (F) and represents the agents having a clear view of the opponent's goal. OpponentNear is either true (T) or false (F) and represents there being other visible agents being near this agent.

Given this network, the expected utility of an action α is:

$$EU(\alpha|BC=x, NV=y, ON=z) = Utility(BC=x, NV=y, ON=z) * [P(ON=z) + P(BC=x|\alpha) + P(NV=y|\alpha, ON=z)]$$

With this, we can determine what actions should be taken under which circumstances. For example, when the Ball is in the agent's control, the opponent's goal is in view, and there are no opponents near, we calculate:

$$EU(Turn\ |BC=T, NV=T, ON=F) = Utility(BC=T, NV=T, ON=F) * [P(ON=F) + P(BC=T \mid Turn) + P(NV=T \mid Turn, ON=F)]$$
$$= 5 * [0.4 + 0.15 + 0.7] = 6.2$$
$$EU(Kick\ |BC=T, NV=T, ON=F) = Utility(BC=T, NV=T, ON=F) * [P(ON=F) + P(BC=T \mid Kick) + P(NV=T \mid Kick, ON=F)]$$
$$= 5 * [0.4 + 0.9 + 0.5] = 9$$
$$EU(Dash\ |BC=T, NV=T, ON=F) = Utility(BC=T, NV=T, ON=F) * [P(ON=F) + P(BC=T \mid Dash) + P(NV=T \mid Dash, ON=F)]$$
$$= 5 * [0.4 + 0.4 + 0.5] = 6.5$$

And select Kick as our action since it has the highest expected utility.