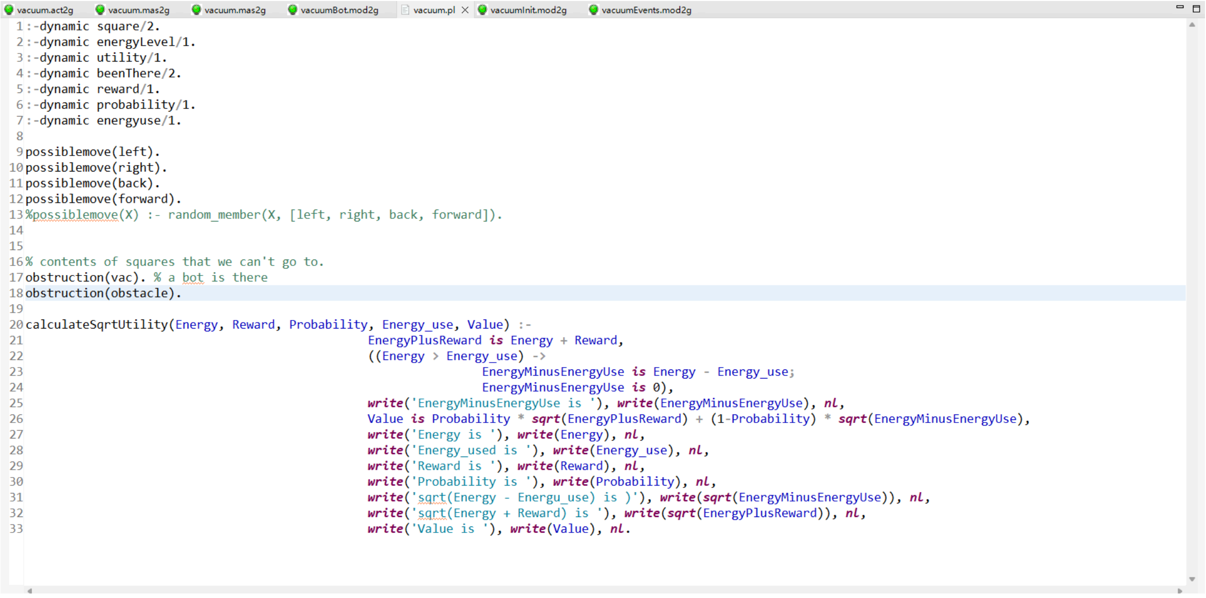
Name: Donghao Li
SU email: dli106@syr.edu

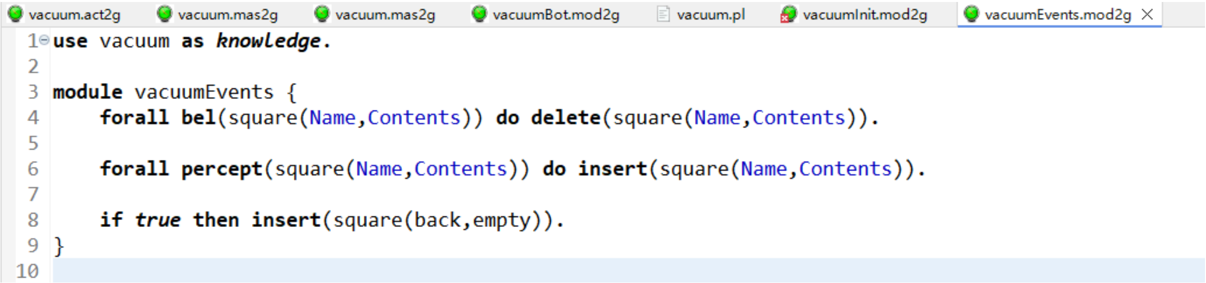
Exercise1:
By using version 1, everytime robot trying to escape the square after clean the first ust, it will choose to move a random direction from thequery(left, right, forward, back). However, for the version 2, the robot will clean up the start point dust and dust at it left then start rotating counterclockwise in the 2 by 2 square until power off. This makes the version 1 robot be able to skip the square.

Exercise2:

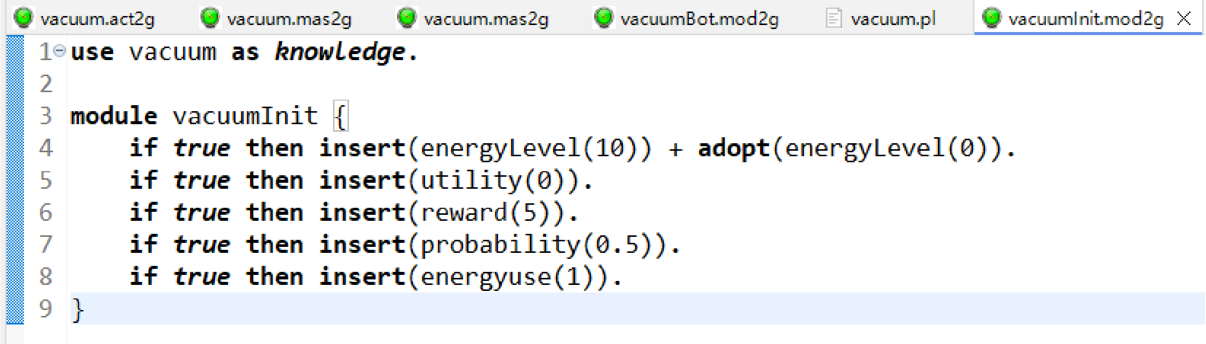
1.



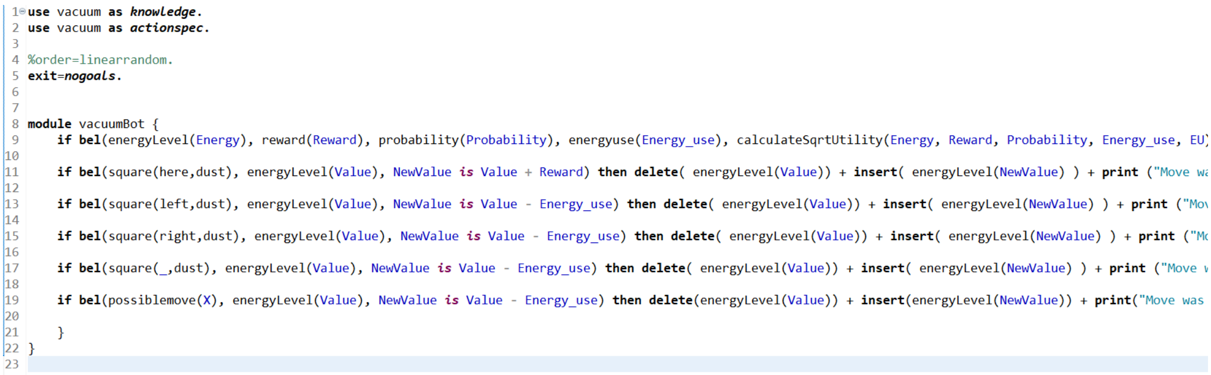
```
:-dynamic square/2.
:-dynamic energyLevel/1.
:-dynamic utility/1.
:-dynamic beenThere/2.
:-dynamic reward/1.
:-dynamic probability/1.
:-dynamic energyuse/1.

possiblemove(left).
possiblemove(right).
possiblemove(back).
possiblemove(forward).
%possiblemove(X) :- random_member(X, [left, right, back, forward]).


% contents of squares that we can't go to.
obstruction(vac). % a bot is there
obstruction(obstacle).

calculateSqrtUtility(Energy, Reward, Probability, Energy_use, Value) :-
                                                EnergyPlusReward is Energy + Reward,
                                                ((Energy > Energy_use) ->
                                                                EnergyMinusEnergyUse is Energy - Energy_use;
                                                                EnergyMinusEnergyUse is 0),
                                                write('EnergyMinusEnergyUse is '), write(EnergyMinusEnergyUse), nl,
                                                Value is Probability * sqrt(EnergyPlusReward) + (1-Probability) * sqrt(EnergyMinusEnergyUse),
                                                write('Energy is '), write(Energy), nl,
                                                write('Energy_used is '), write(Energy_use), nl,
                                                write('Reward is '), write(Reward), nl,
                                                write('Probability is '), write(Probability), nl,
                                                write('sqrt(Energy - Energu_use) is )'), write(sqrt(EnergyMinusEnergyUse)), nl,
                                                write('sqrt(Energy + Reward) is '), write(sqrt(EnergyPlusReward)), nl,
                                                write('Value is '), write(Value), nl.
```



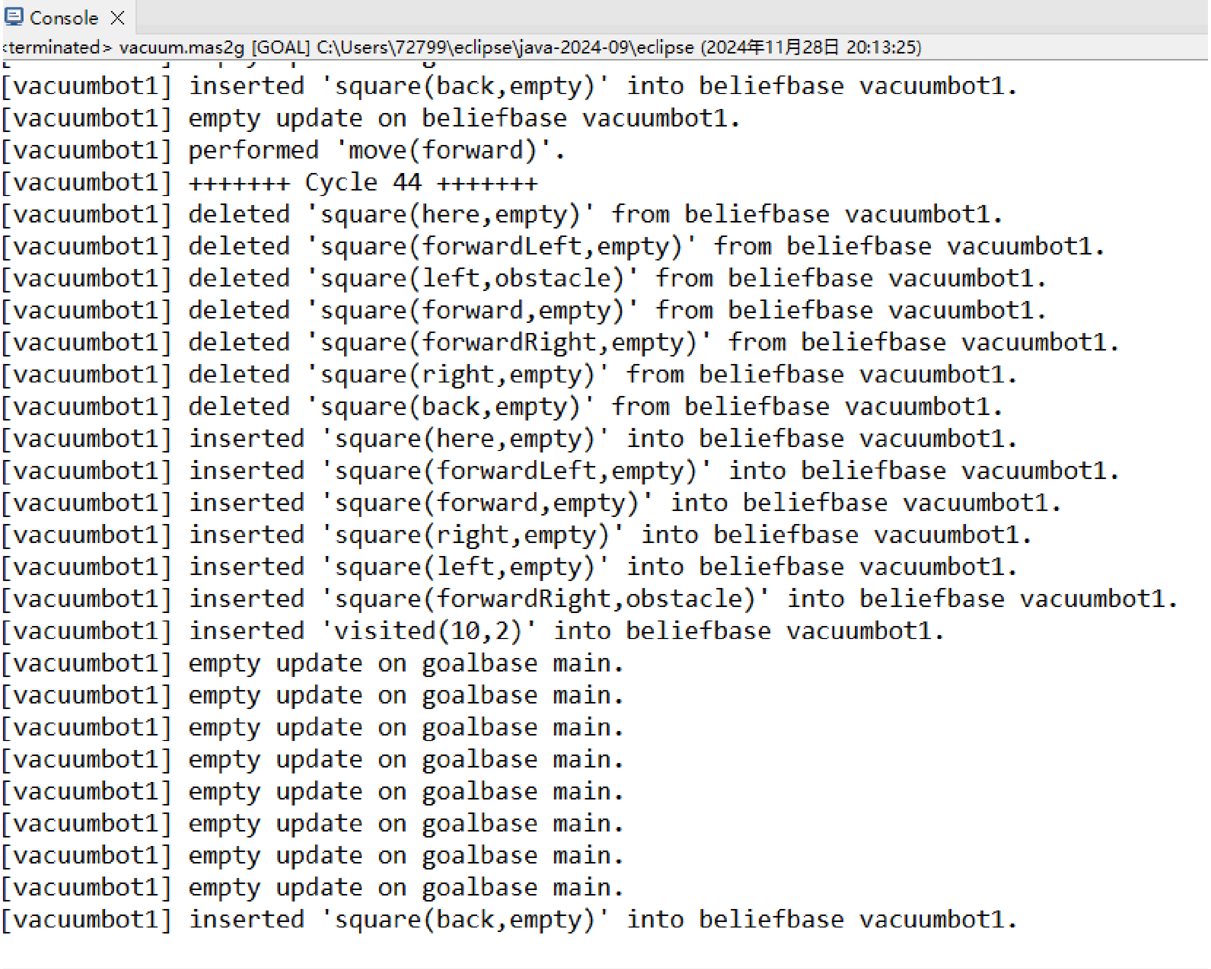
```
use vacuum as knowledge.

module vacuumEvents {
    forall bel(square(Name,Contents)) do delete(square(Name,Contents)).

    forall percept(square(Name,Contents)) do insert(square(Name,Contents)).

    if true then insert(square(back,empty)).
}
```



```
use vacuum as knowledge.

module vacuumInit {
    if true then insert(energyLevel(10)) + adopt(energyLevel(0)).
    if true then insert(utility(0)).
    if true then insert(reward(5)).
    if true then insert(probability(0.5)).
    if true then insert(energyuse(1)).
}
```



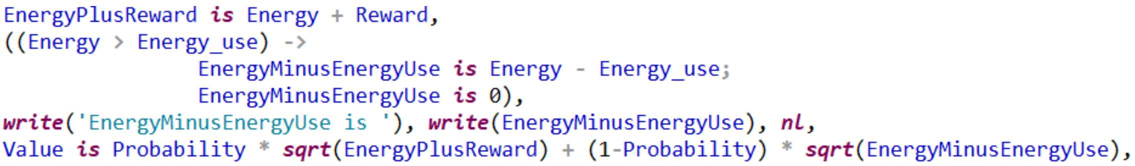
```
use vacuum as knowledge.
use vacuum as actionspec.

%order=linearrandom.
exit=nogoals.


module vacuumBot {
    if bel(energyLevel(Energy), reward(Reward), probability(Probability), energyuse(Energy_use), calculateSqrtUtility(Energy, Reward, Probability, Energy_use, EU)

    if bel(square(here,dust), energyLevel(Value), NewValue is Value + Reward) then delete( energyLevel(Value)) + insert( energyLevel(NewValue) ) + print ("Move wa

    if bel(square(left,dust), energyLevel(Value), NewValue is Value - Energy_use) then delete( energyLevel(Value)) + insert( energyLevel(NewValue) ) + print ("Mov

    if bel(square(right,dust), energyLevel(Value), NewValue is Value - Energy_use) then delete( energyLevel(Value)) + insert( energyLevel(NewValue) ) + print ("Mo

    if bel(square(_,dust), energyLevel(Value), NewValue is Value - Energy_use) then delete( energyLevel(Value)) + insert( energyLevel(NewValue) ) + print ("Move w

    if bel(possiblemove(X), energyLevel(Value), NewValue is Value - Energy_use) then delete(energyLevel(Value)) + insert(energyLevel(NewValue)) + print("Move was

    }
}
```



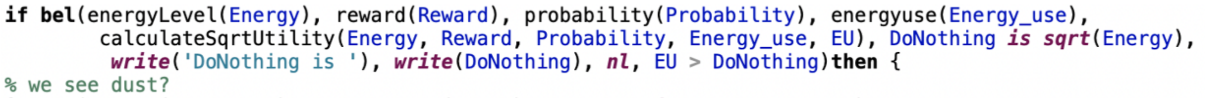
```
<terminated> vacuum.mas2g [GOAL] C:\Users\72799\eclipse\java-2024-09\eclipse (2024年11月28日 20:13:25)
[vacuumbot1] inserted 'square(back,empty)' into beliefbase vacuumbot1.
[vacuumbot1] empty update on beliefbase vacuumbot1.
[vacuumbot1] performed 'move(forward)'.
[vacuumbot1] +++++++ Cycle 44 +++++++
[vacuumbot1] deleted 'square(here,empty)' from beliefbase vacuumbot1.
[vacuumbot1] deleted 'square(forwardLeft,empty)' from beliefbase vacuumbot1.
[vacuumbot1] deleted 'square(left,obstacle)' from beliefbase vacuumbot1.
[vacuumbot1] deleted 'square(forward,empty)' from beliefbase vacuumbot1.
[vacuumbot1] deleted 'square(forwardRight,empty)' from beliefbase vacuumbot1.
[vacuumbot1] deleted 'square(right,empty)' from beliefbase vacuumbot1.
[vacuumbot1] deleted 'square(back,empty)' from beliefbase vacuumbot1.
[vacuumbot1] inserted 'square(here,empty)' into beliefbase vacuumbot1.
[vacuumbot1] inserted 'square(forwardLeft,empty)' into beliefbase vacuumbot1.
[vacuumbot1] inserted 'square(forward,empty)' into beliefbase vacuumbot1.
[vacuumbot1] inserted 'square(right,empty)' into beliefbase vacuumbot1.
[vacuumbot1] inserted 'square(left,empty)' into beliefbase vacuumbot1.
[vacuumbot1] inserted 'square(forwardRight,obstacle)' into beliefbase vacuumbot1.
[vacuumbot1] inserted 'visited(10,2)' into beliefbase vacuumbot1.
[vacuumbot1] empty update on goalbase main.
[vacuumbot1] empty update on goalbase main.
[vacuumbot1] empty update on goalbase main.
[vacuumbot1] empty update on goalbase main.
[vacuumbot1] empty update on goalbase main.
[vacuumbot1] empty update on goalbase main.
[vacuumbot1] empty update on goalbase main.
[vacuumbot1] empty update on goalbase main.
[vacuumbot1] inserted 'square(back,empty)' into beliefbase vacuumbot1.
```

As you can see from the screenshot upon, i have fully modified my code as the ppt shows. I will take a screenshot of console to show it works properly and attach the running video with this file.

2.



```
EnergyPlusReward is Energy + Reward,
((Energy > Energy_use) ->
                EnergyMinusEnergyUse is Energy - Energy_use;
                EnergyMinusEnergyUse is 0),
write('EnergyMinusEnergyUse is '), write(EnergyMinusEnergyUse), nl,
Value is Probability * sqrt(EnergyPlusReward) + (1-Probability) * sqrt(EnergyMinusEnergyUse),
```



```
if bel(energyLevel(Energy), reward(Reward), probability(Probability), energyuse(Energy_use),
        calculateSqrtUtility(Energy, Reward, Probability, Energy_use, EU), DoNothing is sqrt(Energy),
         write('DoNothing is '), write(DoNothing), nl, EU > DoNothing)then {
% we see dust?
```

As shown in the figure above, value is the expected utility of the robot, it’s calculated by the probability of square root of move gain reward plus the rest of the probability of square root of making action gains penalty. then in the bot model we compare it with square root of current energy. If after taking action the energy level is lower than the current, the robot tends not to move. higher reward and probability will encourage the robot to take action, reversly, lower probability will make the robot stay. higher energy use will make the robot stay, and lower energy use encourages the robot to move.

First run, i set the energyuse to 5 which makes the robot stays at the beginning position even there exists dust next to it, this shows when the cost is as big as reward, robot is tending not to move. calculation: EU: 0.5*sqrt(10+5) + 0.5*sqrt(10-5) = 3.0545 DoNothing:sqrt10 = 3.1623.

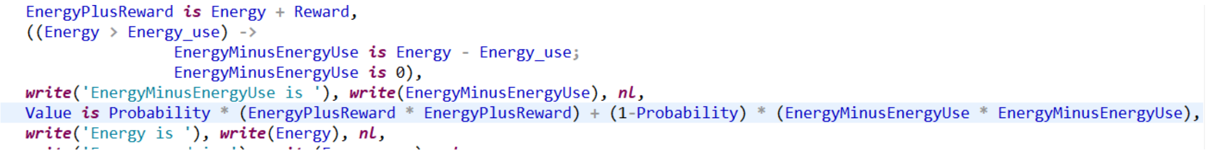
Second run,   reward(10), probability(0.5), energyuse(5), robot movem calculation: 0.5*(sqrt20) + 0.5*sqrt(5) = 3.354 DoNothing stays same. But after 6 cycles, the robot stops movin tog go to sleep because because this time EU is less than Donothing, although there exists dust out there, robot still choose not to move.

Third run: reward(5), probability(0.9), energyuse(5), the robot start to move. calculation: EU: 0.9*sqrt(15) + 0.1*sqrt(5) = 3.7093 DoNothing no change EU > Donothing the robot move. Robot sleep at cycle 4.

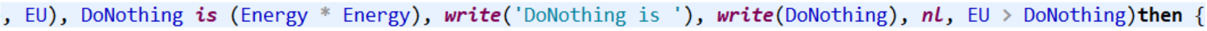
Fourth run: reward(10), probability(0.8), energyuse(5) the robot start to move, calculation: EU: 4.1777, Donothing: 3.1623. Sleep at cycle 31.

In conclusion, with higher reward and probability and lower energy use, the robot would be more likely to run more cycles.

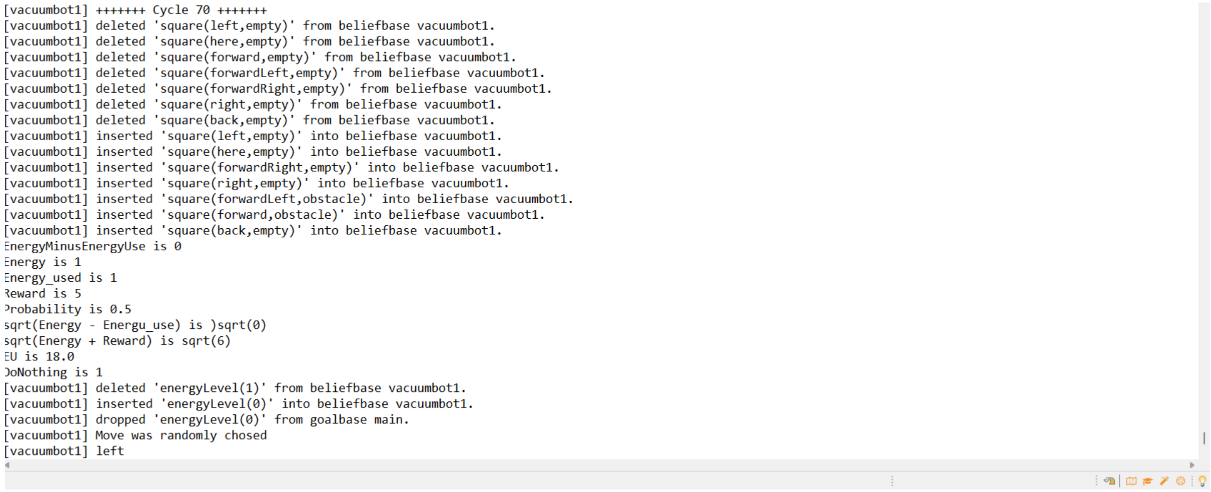
3.



```
EnergyPlusReward is Energy + Reward,
((Energy > Energy_use) ->
                EnergyMinusEnergyUse is Energy - Energy_use;
                EnergyMinusEnergyUse is 0),
write('EnergyMinusEnergyUse is '), write(EnergyMinusEnergyUse), nl,
Value is Probability * (EnergyPlusReward * EnergyPlusReward) + (1-Probability) * (EnergyMinusEnergyUse * EnergyMinusEnergyUse),
write('Energy is '), write(Energy), nl,
```



```
, EU), DoNothing is (Energy * Energy), write('DoNothing is '), write(DoNothing), nl, EU > DoNothing)then {
```



```
[vacuumbot1] +++++++ Cycle 70 +++++++
[vacuumbot1] deleted 'square(left,empty)' from beliefbase vacuumbot1.
[vacuumbot1] deleted 'square(here,empty)' from beliefbase vacuumbot1.
[vacuumbot1] deleted 'square(forward,empty)' from beliefbase vacuumbot1.
[vacuumbot1] deleted 'square(forwardLeft,empty)' from beliefbase vacuumbot1.
[vacuumbot1] deleted 'square(forwardRight,empty)' from beliefbase vacuumbot1.
[vacuumbot1] deleted 'square(right,empty)' from beliefbase vacuumbot1.
[vacuumbot1] deleted 'square(back,empty)' from beliefbase vacuumbot1.
[vacuumbot1] inserted 'square(left,empty)' into beliefbase vacuumbot1.
[vacuumbot1] inserted 'square(here,empty)' into beliefbase vacuumbot1.
[vacuumbot1] inserted 'square(forwardRight,empty)' into beliefbase vacuumbot1.
[vacuumbot1] inserted 'square(right,empty)' into beliefbase vacuumbot1.
[vacuumbot1] inserted 'square(forwardLeft,obstacle)' into beliefbase vacuumbot1.
[vacuumbot1] inserted 'square(forward,obstacle)' into beliefbase vacuumbot1.
[vacuumbot1] inserted 'square(back,empty)' into beliefbase vacuumbot1.
EnergyMinusEnergyUse is 0
Energy is 1
Energy_used is 1
Reward is 5
Probability is 0.5
sqrt(Energy - Energu_use) is )sqrt(0)
sqrt(Energy + Reward) is sqrt(6)
EU is 18.0
DoNothing is 1
[vacuumbot1] deleted 'energyLevel(1)' from beliefbase vacuumbot1.
[vacuumbot1] inserted 'energyLevel(0)' into beliefbase vacuumbot1.
[vacuumbot1] dropped 'energyLevel(0)' from goalbase main.
[vacuumbot1] Move was randomly chosed
[vacuumbot1] left
```

Apparently, with rish seeking function utility, robot are more likely to run more cycles.
1 try: reward(5), probability(0.5), energyuse(1), robot start with moving, stop at cycle 70

2 try: reward(5), probability(0.5), energyuse(5), robot start with moving, stop at cycle 6.

3 try: reward(5), probability(0.3), energyuse(5), robot start with not moving. EU = 85, DoNothing = 100.

same as the conclusion before, for risk seeking function, higher reward and probability, and lower energy use will support robot soing more cycles. This situation becomes more significant by using risk seeking function.