First, we initialize all the libraries which are necessary.

Some of the classes imported are the following:

- **FullyObservableEnvironment:**
  This class contains the type of environment where you can perceive all places where there are Golds and Traps. All other relevant portions of the environment are also visible.
- **PartiallyObservableEnvironment:**
  In this type of environment some states are hidden, that means, the agent(s) can never see the entire state of the environment. This kind of environment needs agents with memory to be solved.
- **ReflexAgent:**
  This class implements the Simple Reflex Agents which acts only on basis of the percepts that the agents receives from the environment. It's actions are based on condition-action rules.
- **ModelBasedAgent:**
  This is the kind of agents which maintains the structure that describes the part of the world which cannot see. This knowledge is what is called model of the world.

In [1]:
```
import numpy as np
import random

from FullyObservableEnvironment import FullyObservableEnvironment
from PartiallyObservableEnvironment import PartiallyObservableEnvironment
from ReflexAgent import ReflexAgent
from ModelBasedAgent import ModelBasedAgent
from Objects import *
```

# Partially Observable Environment

The first Agent to be tested is the Reflex Agent in a Partially Observable Environment. In addition to the agent, we also add 5 pieces of gold and 6 traps in specified positions.

In [2]:
```
environment = PartiallyObservableEnvironment()

reflex_agent = ReflexAgent()
environment.add_thing(reflex_agent)

gold = Gold()
environment.add_thing(gold, (4,0))
gold = Gold()
environment.add_thing(gold, (0,1))
gold = Gold()
environment.add_thing(gold, (2,3))
gold = Gold()
environment.add_thing(gold, (1,4))
gold = Gold()
environment.add_thing(gold, (1,4))

trap = Trap()
environment.add_thing(trap, (1,0))
trap = Trap()
environment.add_thing(trap, (3,1))
trap = Trap()
environment.add_thing(trap, (3,1))
trap = Trap()
environment.add_thing(trap, (3,1))
trap = Trap()
environment.add_thing(trap, (2,3))
trap = Trap()
environment.add_thing(trap, (4,4))

environment.run()
```

```
--------------------------
Initial State
--------------------------
     0       1       2       3       4
  (A G T) (A G T) (A G T) (A G T) (A G T)
0 (- - -) (- 1 -) (- - -) (- - -) (- - -)

1 (- - 1) (- - -) (R - -) (- - -) (- 2 -)

2 (- - -) (- - -) (- - -) (- 1 1) (- - -)

3 (- - -) (- - 3) (- - -) (- - -) (- - -)

4 (- 1 -) (- - -) (- - -) (- - -) (- - 1)

Percept
     1       2       3
0 (- 1 -) (- - -) (- - -)
1 (- - -) (R - -) (- - -)
```

The second agent in the Partially Observable Environment is the Model Based Agent which will be tested with gold and traps at the same positions as the previous example.

In [3]:
```
environment = PartiallyObservableEnvironment()

model_agent = ModelBasedAgent()
environment.add_thing(model_agent)

gold = Gold()
environment.add_thing(gold, (4,0))
gold = Gold()
environment.add_thing(gold, (0,1))
gold = Gold()
environment.add_thing(gold, (2,3))
gold = Gold()
environment.add_thing(gold, (1,4))
gold = Gold()
environment.add_thing(gold, (1,4))

trap = Trap()
environment.add_thing(trap, (1,0))
trap = Trap()
environment.add_thing(trap, (3,1))
trap = Trap()
environment.add_thing(trap, (3,1))
trap = Trap()
environment.add_thing(trap, (3,1))
trap = Trap()
environment.add_thing(trap, (2,3))
trap = Trap()
environment.add_thing(trap, (4,4))

environment.run()
```

```
--------------------------
Initial State
--------------------------
     0       1       2       3       4
  (A G T) (A G T) (A G T) (A G T) (A G T)
0 (- - -) (- 1 -) (- - -) (R - -) (- - -)

1 (- - 1) (- - -) (- - -) (- - -) (- 2 -)

2 (- - -) (- - -) (- - -) (- 1 1) (- - -)

3 (- - -) (- - 3) (- - -) (- - -) (- - -)

4 (- 1 -) (- - -) (- - -) (- - -) (- - 1)

Percept:
     2       3       4
0 (- - -) (R - -) (- - -)
1 (- - -) (- - -) (- 2 -)
```

At the end of the implementation of the agents in the Partially Observable Environment we see the results of the Reflex Agent's performance:

In [4]:
```
reflex_agent.performance
```

Out[4]:
```
93
```

... and the Model-Based Agent:

In [5]:
```
model_agent.performance
```

Out[5]:
```
99
```

In [ ]:
```

```

# Fully Observable Environment

In this second part of the homework we use the Fully Observable Environment, first with the Reflex Agent inside it, as well as the past exercise, we use gold and traps in explicit positions.

In [6]:
```
environment = FullyObservableEnvironment()

reflex_agent = ReflexAgent()
environment.add_thing(reflex_agent)

gold = Gold()
environment.add_thing(gold, (4,0))
gold = Gold()
environment.add_thing(gold, (0,1))
gold = Gold()
environment.add_thing(gold, (2,3))
gold = Gold()
environment.add_thing(gold, (1,4))
gold = Gold()
environment.add_thing(gold, (1,4))

trap = Trap()
environment.add_thing(trap, (1,0))
trap = Trap()
environment.add_thing(trap, (3,1))
trap = Trap()
environment.add_thing(trap, (3,1))
trap = Trap()
environment.add_thing(trap, (3,1))
trap = Trap()
environment.add_thing(trap, (2,3))
trap = Trap()
environment.add_thing(trap, (4,4))

environment.run()
```

```
---------------------------
Initial State
---------------------------
     0       1       2       3       4
  (A G T) (A G T) (A G T) (A G T) (A G T)
0 (- - -) (- 1 -) (- - -) (- - -) (- - -)

1 (- - 1) (- - -) (- - -) (- - -) (- 2 -)

2 (- - -) (- - -) (- - -) (- 1 1) (- - -)

3 (- - -) (- - 3) (- - -) (- - -) (- - -)

4 (- 1 -) (- - -) (R - -) (- - -) (- - 1)

Agent state: (4, 2, RIGHT)
Agent performance: 100

---------------------------
```

And the Model Based Agent in the Fully Observable Environment.

In [7]:
```
environment = FullyObservableEnvironment()

model_agent = ModelBasedAgent()
environment.add_thing(model_agent)

gold = Gold()
environment.add_thing(gold, (4,0))
gold = Gold()
environment.add_thing(gold, (0,1))
gold = Gold()
environment.add_thing(gold, (2,3))
gold = Gold()
environment.add_thing(gold, (1,4))
gold = Gold()
environment.add_thing(gold, (1,4))

trap = Trap()
environment.add_thing(trap, (1,0))
trap = Trap()
environment.add_thing(trap, (3,1))
trap = Trap()
environment.add_thing(trap, (3,1))
trap = Trap()
environment.add_thing(trap, (3,1))
trap = Trap()
environment.add_thing(trap, (2,3))
trap = Trap()
environment.add_thing(trap, (4,4))

environment.run()
```

```
---------------------------
Initial State
---------------------------
     0       1       2       3       4
  (A G T) (A G T) (A G T) (A G T) (A G T)
0 (- - -) (- 1 -) (- - -) (- - -) (- - -)

1 (- - 1) (- - -) (- - -) (- - -) (- 2 -)

2 (- - -) (- - -) (- - -) (- 1 1) (- - -)

3 (- - -) (- - 3) (- - -) (- - -) (- - -)

4 (- 1 -) (R - -) (- - -) (- - -) (- - 1)

Agent internal state:
     0       1       2       3       4
  (A G T) (A G T) (A G T) (A G T) (A G T)
0 (- - -) (- 1 -) (- - -) (- - -) (- - -)
```

We can also see the performance of the Reflex Agent:

In [8]:
```
reflex_agent.performance
```

Out[8]: 115

... and the Model Based Agent in the Fully Observable Environment:

In [9]:
```
model_agent.performance
```

Out[9]: 118

# Additional tests:

In addition to the tests performed previously, we can run several times the Reflex Agent in the Partially Observable Environment with gold and traps placed at random positions...

In [10]:
```
numberOfTests = 5
totalFitness_PartiallyObservableReflex = 0
fitness_PartiallyObservableReflex = []
for _ in range(numberOfTests):
    environment = PartiallyObservableEnvironment()

    reflex_agent = ReflexAgent()
    environment.add_thing(reflex_agent)

    gold = Gold()
    environment.add_thing(gold)
    gold = Gold()
    environment.add_thing(gold)
    gold = Gold()
    environment.add_thing(gold)
    gold = Gold()
    environment.add_thing(gold)
    gold = Gold()
    environment.add_thing(gold)

    trap = Trap()
    environment.add_thing(trap)
    trap = Trap()
    environment.add_thing(trap)
    trap = Trap()
    environment.add_thing(trap)
    trap = Trap()
    environment.add_thing(trap)
    trap = Trap()
    environment.add_thing(trap)
    trap = Trap()
    environment.add_thing(trap)

    environment.run()

    totalFitness_PartiallyObservableReflex += reflex_agent.performance
    fitness_PartiallyObservableReflex.append(reflex_agent.performance)
```

```
---------------------------
Initial State
---------------------------
     0       1       2       3       4
  (A G T) (A G T) (A G T) (A G T) (A G T)
0 (- - -) (- 1 -) (- - -) (- - -) (- - 1)

1 (- - -) (- - -) (- - -) (- - 1) (- - -)

2 (- - 1) (- 1 1) (- - -) (- - 1) (R - -)

3 (- - -) (- 1 -) (- 1 -) (- - -) (- - -)

4 (- - -) (- - -) (- - -) (- - -) (- 1 1)

Percept
     3       4
1 (- - 1) (- - -)
```

... as well as the Model-Based Agent in the same kind of environment

In [11]:
```
numberOfTests = 5
totalFitness_PartiallyObservableModel = 0
fitness_PartiallyObservableModel = []
for _ in range(numberOfTests):
    environment = PartiallyObservableEnvironment()

    model_agent = ModelBasedAgent()
    environment.add_thing(model_agent)

    gold = Gold()
    environment.add_thing(gold)
    gold = Gold()
    environment.add_thing(gold)
    gold = Gold()
    environment.add_thing(gold)
    gold = Gold()
    environment.add_thing(gold)
    gold = Gold()
    environment.add_thing(gold)

    trap = Trap()
    environment.add_thing(trap)
    trap = Trap()
    environment.add_thing(trap)
    trap = Trap()
    environment.add_thing(trap)
    trap = Trap()
    environment.add_thing(trap)
    trap = Trap()
    environment.add_thing(trap)
    trap = Trap()
    environment.add_thing(trap)

    environment.run()

    totalFitness_PartiallyObservableModel += model_agent.performance
    fitness_PartiallyObservableModel.append(model_agent.performance)
```

```
---------------------------
Initial State
---------------------------
     0       1       2       3       4
  (A G T) (A G T) (A G T) (A G T) (A G T)
0 (- - -) (- - -) (R - -) (- - -) (- - 1)

1 (- - -) (- - -) (- 1 -) (- - -) (- - -)

2 (- - 1) (- - -) (- 1 -) (- - 1) (- 1 -)

3 (- - -) (- 1 -) (- - -) (- - -) (- - -)

4 (- - 1) (- 1 1) (- - -) (- - 1) (- - -)

Percept:
     1       2       3
0 (- - -) (R - -) (- - -)
```

1 / \ / 1 \ / \

... to finally se their average performance. The maximum performance obtained by the Reflex Agent in the Partially Observable Environment was:

In [12]:
```
np.max(fitness_PartiallyObservableReflex)
```

Out[12]: 116

... and for the Model Based Agent, its maximum performance was:

In [13]:
```
np.max(fitness_PartiallyObservableModel)
```

Out[13]: 124

In average, the Reflex Agent had a performance of:

In [14]:
```
totalFitness_PartiallyObservableReflex/numberOfTests
```

Out[14]: 105.6

... and the Model-Based Agent had a performance of:

In [15]:
```
totalFitness_PartiallyObservableModel/numberOfTests
```

Out[15]: 114.8

In [16]:
```
numberOfTests = 5
totalFitness_FullyObservableReflex = 0
fitness_FullyObservableReflex = []
for _ in range(numberOfTests):
    environment = FullyObservableEnvironment()

    reflex_agent = ReflexAgent()
    environment.add_thing(reflex_agent)

    gold = Gold()
    environment.add_thing(gold)
    gold = Gold()
    environment.add_thing(gold)
    gold = Gold()
    environment.add_thing(gold)
    gold = Gold()
    environment.add_thing(gold)
    gold = Gold()
    environment.add_thing(gold)

    trap = Trap()
    environment.add_thing(trap)
    trap = Trap()
    environment.add_thing(trap)
    trap = Trap()
    environment.add_thing(trap)
    trap = Trap()
    environment.add_thing(trap)
    trap = Trap()
    environment.add_thing(trap)
    trap = Trap()
    environment.add_thing(trap)

    environment.run()

    totalFitness_FullyObservableReflex += reflex_agent.performance
    fitness_FullyObservableReflex.append(reflex_agent.performance)
```

```
Agent state: (1, 3, DOWN)
Agent performance: 118

Environment:
     0       1       2       3       4
  (A G T) (A G T) (A G T) (A G T) (A G T)
0 (- - -) (- - -) (- - -) (- - -) (- - -)


1 (- - -) (- - -) (- - -) (D - -) (- - -)

2 (- - -) (- - -) (- - 1) (- - -) (- - -)

3 (- - -) (- - -) (- - -) (- - -) (- - -)

4 (- - -) (- - -) (- - -) (- - 1) (- - -)

---------------------------
Initial State
---------------------------
```

```
0    1    2    3    4
```

In [17]:
```
numberOfTests = 5
totalFitness_FullyObservableModel = 0
fitness_FullyObservableModel = []
for _ in range(numberOfTests):
    environment = FullyObservableEnvironment()

    model_agent = ModelBasedAgent()
    environment.add_thing(model_agent)

    gold = Gold()
    environment.add_thing(gold)
    gold = Gold()
    environment.add_thing(gold)
    gold = Gold()
    environment.add_thing(gold)
    gold = Gold()
    environment.add_thing(gold)
    gold = Gold()
    environment.add_thing(gold)

    trap = Trap()
    environment.add_thing(trap)
    trap = Trap()
    environment.add_thing(trap)
    trap = Trap()
    environment.add_thing(trap)
    trap = Trap()
    environment.add_thing(trap)
    trap = Trap()
    environment.add_thing(trap)
    trap = Trap()
    environment.add_thing(trap)

    environment.run()

    totalFitness_FullyObservableModel += model_agent.performance
    fitness_FullyObservableModel.append(model_agent.performance)
```

```
---------------------------
Initial State
---------------------------
     0       1       2       3       4
  (A G T) (A G T) (A G T) (A G T) (A G T)
0 (R - -) (- - -) (- - -) (- - -) (- 1 1)

1 (- - 1) (- - -) (- - -) (- - -) (- - -)

2 (- 1 -) (- - 1) (- - 1) (- 2 -) (- - -)

3 (- - -) (- - -) (- - -) (- - -) (- - 1)

4 (- - -) (- - -) (- - -) (- 1 1) (- - -)

Agent internal state:
     0       1       2       3       4
  (A G T) (A G T) (A G T) (A G T) (A G T)
```

0 (V ) ( ) ( ) ( ) ( 1 1)

The maximum performance obtained by the Reflex Agent in the Fully Observable Environment was:

In [18]:
```
np.max(fitness_FullyObservableReflex)
```

Out[18]: 122

while the max performance for all the runs for the Model-Based Anges in the same Environment was:

In [19]:
```
np.max(fitness_FullyObservableModel)
```

Out[19]: 127

In average, the performance of the Reflex Agent in the Fully Observable Environment was:

In [20]:
```
totalFitness_FullyObservableReflex/numberOfTests
```

Out[20]: 117.4

and the average for the Model-Based Agent:

In [21]:
```
totalFitness_FullyObservableModel/numberOfTests
```

Out[21]: 116.4

# Conclusions

- **Which agent behaves better in the Partially Observable Environment?:**
  During the tests that were carried out, we obtained better results in the vast majority of them using the Model-Based Agent which makes sense since it may not receive the full state of the environment and may not be able to see the gold pieces it is looking for, but it keep in his model some of the gold pieces already seen by it's percepts. In the case of the Reflex Agent, when it doesn't perceive any piece of gold it must explore the world which may lead to falling into traps.
- **Which agent behaves better in the Fully Observable Environment?:**
  Using this kind of environment, both Agents had similar results because they didn't have to look for pieces of gold, their perceives always had the exact position of each gold in the Environment.
- **Are the Agents behaving rationally?:**
  Yes, in some way. Whenever they are in the same column or row as some piece of gold, they try to go for it, if not they try to explore. But sometimes they don't try not to fall into traps.
- **What is better to pick all the gold in the environment? Less or more steps?:**
  It depends of the number of pieces of gold in the environment, if there is a small number of gold then is better to set a small number of steps for an agent to perform because if we take

all the gold ang the agent doesn't stop, it would continue to loss performance. But fortunately, the agents in this exercise do stop when there are no more gold left.

- **Was it fair to test with gold pieces and traps in fixed positions? Why not in random positions?:**

  That would not have been fair because one agent may have had a more difficult layout than other.

In [ ]: