# Assignment 3

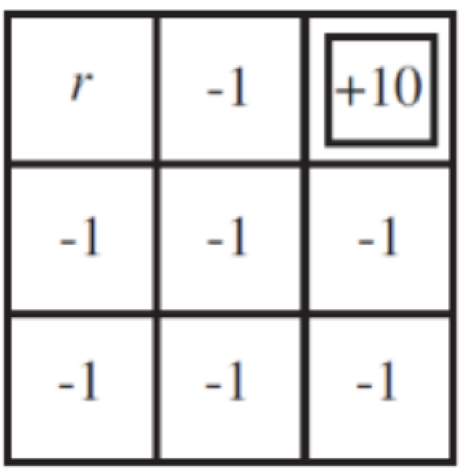
# Markov Decision Processes

## Team members:

| Names | Id |
|---|---|
| حسن علي حسن إبراهيم عبد الواحد | 19015603 |
| عبد الرحمن عادل عبد الفتاح عبد الرؤوف | 17012296 |
| منة الله محمود سعد محمد عمر | 19016713 |
| محمد هارون صالح | 19016520 |

## Description:

Consider the 3x3 world shown in the following figure:




The agent has four actions Up, Down, Right and Left. The transition model is: 80% of the time the agent goes in the direction it selects; the rest of the time it moves at right angles to the intended direction. A collision with a wall results in no movement.

In this project we implement value iteration and policy iteration algorithm to find the optimal policy for this world for each value of r below:

- r = 100
- r = 3
- r = 0

• r = -3

## Algorithms:

Value iteration:

---

**procedure** VALUEITERATION($S, A, T, R, \theta, \gamma$)
  assign $V^0(S)$ arbitrarily, $i \leftarrow 0$
  **repeat**
    $i \leftarrow i + 1$
    **for** each state s **do** ▷ Evaluate policy
      **for** each action a **do**
        $V^i(s) = \sum_{s'} \mathrm{T(s'|s,a)}(R(s,a) + \gamma V^{i-1}(s'))$
      **end for**
    **end for**
    **for** each state s **do** ▷ Improve policy
      $\pi^i(s) = \underset{a}{\operatorname{argmax}} \sum_{s'} \mathrm{T(s'|s,a)}(R(s,a) + \gamma V^i(s'))$
    **end for**
  **until** $\forall s |V^i(s) - V^{i-1}(s)| < \theta$
  **return** $\pi^i, V^i$ ▷ Return optimal policy and its value
**end procedure**

---

Policy Iteration:

**Policy Iteration (using iterative policy evaluation) for estimating $\pi \approx \pi_*$**

1. Initialization
   $V(s) \in \mathbb{R}$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$

2. Policy Evaluation
   Loop:
     $\Delta \leftarrow 0$
     Loop for each $s \in \mathcal{S}$:
       $v \leftarrow V(s)$
       $V(s) \leftarrow \sum_{s',r} p(s', r \,|\, s, \pi(s)) \big[r + \gamma V(s')\big]$
       $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
   until $\Delta < \theta$ (a small positive number determining the accuracy of estimation)

3. Policy Improvement
   *policy-stable* $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
     *old-action* $\leftarrow \pi(s)$
     $\pi(s) \leftarrow \operatorname{argmax}_a \sum_{s',r} p(s', r \,|\, s, a) \big[r + \gamma V(s')\big]$
     If *old-action* $\neq \pi(s)$, then *policy-stable* $\leftarrow$ *false*
   If *policy-stable*, then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

## Policy for each r:

## r = 0:

## Value Iteration:

```
Values:
0.0000  9.5575  0.0000
6.1447  8.1952  9.5575
5.5978  6.8627  8.0455
Best policy:
   _    right      _
right   right     up
right   right     up
-------------------------------------------
Execution Time: 66 Millis
```

## Policy iteration:

```
Values:
0.0000  9.5575  0.0000
6.1447  8.1952  9.5575
5.5978  6.8627  8.0455

policy:
   _    right      _
right   right     up
right   right     up
-------------------------------------------
Execution Time: 15 Millis

Process finished with exit code 0
```

## For r = 3

## Value iteration:

```
Values:
0.0000  9.5575  0.0000
6.4480  8.1952  9.5575
5.6311  6.8627  8.0455
Best policy:
   _    right     _
right   right     up
right   right     up
-------------------------------------------
Execution Time: 85 Millis

Process finished with exit code 0
```

## Policy Iteration:

```
Values:
0.0000  9.5575  0.0000
6.4480  8.1952  9.5575
5.6311  6.8627  8.0455

policy:
   _    right     _
right   right     up
right   right     up
-------------------------------------------
Execution Time: 22 Millis
```

## For r = -3:

## Value iteration:

```
Values:
0.0000  9.5575  0.0000
5.8414  8.1952  9.5575
5.5645  6.8627  8.0455
Best policy:
   _    right     _
right   right     up
right   right     up
-------------------------------------------
Execution Time: 73 Millis

Process finished with exit code 0
```

## Policy iteration:

```
Values:
0.0000  9.5575  0.0000
5.8414  8.1952  9.5575
5.5645  6.8627  8.0455

policy:
   _    right     _
right   right     up
right   right     up
-------------------------------------------
Execution Time: 31 Millis
```

## For r = 100:

## Value iteration:

```
Values:
0.0000  99.1955 0.0000
99.1955 96.7185 90.1053
96.4463 94.2959 91.6790
Best policy:
   _    left       _
  up    left    down
  up    left    left
-------------------------------------------
Execution Time: 86 Millis
```

## Policy iteration:

```
Values:
0.0000  99.1955 0.0000
99.1955 96.7185 90.1053
96.4463 94.2959 91.6790

policy:
   _    left       _
  up    left    down
  up    left    left
-------------------------------------------
Execution Time: 31 Millis
```

## Explain intuitively why the value of r leads to each policy:

In r = 0, r = 3, r = -3

The reward of 10 is greater than 0 or 3 or  -3

So it starts by moving to r but at the end it moves to 10

In r = 100

The reward of 100 is greater than 10 so the policy starts by moving to 10 but at the end it moves to 100

**Programming language used:** java

**Data structure used:** just arrays