Reinforcement learning project report

1- custom map 1 hole reward = -0.1, goal reward = 1, move reward = -0.1, theta = 0.0001, is slippery = false, gamma = [1, 0.9, 0.5, 0.1]

gamma = 1:

```
rewards: 0.6  - moves: 5  - final state: 7
rewards: 0.6  - moves: 5  - final state: 7
rewards: 0.6  - moves: 5  - final state: 7
rewards: 0.6  - moves: 5  - final state: 7
rewards: 0.6  - moves: 5  - final state: 7
  (Right)
HFSFFFFG


1
[['Lt' 'Rt' 'Rt' 'Rt' 'Rt' 'Rt' 'Rt' 'Lt']]
```

gamma 0.9:

```
rewards: 0.6  - moves: 5  - final state: 7
rewards: 0.6  - moves: 5  - final state: 7
rewards: 0.6  - moves: 5  - final state: 7
rewards: 0.6  - moves: 5  - final state: 7
rewards: 0.6  - moves: 5  - final state: 7
  (Right)
```

```
HFSFFFFG

1
[['Lt' 'Rt' 'Rt' 'Rt' 'Rt' 'Rt' 'Rt' 'Lt']]
```

gamma 0.5:

```
rewards: 0.6  - moves: 5  - final state: 7
rewards: 0.6  - moves: 5  - final state: 7
rewards: 0.6  - moves: 5  - final state: 7
rewards: 0.6  - moves: 5  - final state: 7
rewards: 0.6  - moves: 5  - final state: 7
  (Right)
HFSFFFFG
```

gamma 0.1:

```
rewards: 0.6  - moves: 5  - final state: 7
rewards: 0.6  - moves: 5  - final state: 7
rewards: 0.6  - moves: 5  - final state: 7
rewards: 0.6  - moves: 5  - final state: 7
rewards: 0.6  - moves: 5  - final state: 7
  (Right)
HFSFFFFG

1
```

```
[['Lt' 'Rt' 'Rt' 'Rt' 'Rt' 'Rt' 'Rt' 'Lt']]
```

gamma determines how much weight to give to future rewards,

so by lowering it you would choose the closer reward (short term reward).

high gamma → longterm reward is better

low gamma → shortterm reward is better

2- custom map 2 ,policy iteration → hole reward = -4, goal reward = 10, move reward = -0.9

```
rewards: 3.6999999999999993  - moves: 8  - final state: 24
rewards: 3.6999999999999993  - moves: 8  - final state: 24
rewards: 3.6999999999999993  - moves: 8  - final state: 24
rewards: 3.6999999999999993  - moves: 8  - final state: 24
rewards: 3.6999999999999993  - moves: 8  - final state: 24
```

```
[['Rt' 'Rt' 'Rt' 'Dn' 'Dn']
 ['Lt' 'Lt' 'Lt' 'Dn' 'Lt']
 ['Dn' 'Dn' 'Dn' 'Dn' 'Lt']
 ['Dn' 'Dn' 'Dn' 'Dn' 'Dn']
 ['Rt' 'Rt' 'Rt' 'Rt' 'Lt']]
```

3- custom map 3 , hole reward = -5, goal reward = 5, move reward = -0.5, gamma = 0.9, theta = 0.0001

is slippery = false:

```
rewards: 0.5  - moves: 6  - final state: 20
rewards: 0.5  - moves: 6  - final state: 20
rewards: 0.5  - moves: 6  - final state: 20
rewards: 0.5  - moves: 6  - final state: 20
rewards: 0.5  - moves: 6  - final state: 20
```

```
[['Rt' 'Dn' 'Dn' 'Dn' 'Dn']
 ['Lt' 'Dn' 'Dn' 'Dn' 'Dn']
 ['Lt' 'Dn' 'Dn' 'Dn' 'Dn']
 ['Lt' 'Dn' 'Dn' 'Dn' 'Dn']
 ['Lt' 'Lt' 'Lt' 'Lt' 'Lt']]
```

is slippery = true:

```
rewards: -12.100000000000001  - moves: 20  - final state: 20
rewards: -54.399999999999935  - moves: 67  - final state: 20
rewards: -31.89999999999997  - moves: 42  - final state: 20
rewards: -25.59999999999998  - moves: 35  - final state: 20
rewards: -35.499999999999964  - moves: 46  - final state: 20
```

```
[['UP' 'Rt' 'Lt' 'Lt' 'Lt']
 ['Lt' 'Rt' 'Dn' 'Dn' 'Dn']
```

```
['Lt' 'Rt' 'Dn' 'Dn' 'Dn']
['Lt' 'Rt' 'Dn' 'Dn' 'Dn']
['Lt' 'Dn' 'Lt' 'Lt' 'Dn']]
```

if slippery is false the path to goal is clear and player will gain more points and will get to goal quickly, and if slippery is true the path is unclear and player should have more tries to get to goal.

4- custom map 4:

is slippery = false:

```
rewards: -1.3000000000000007  - moves: 8  - final state: 29
rewards: -1.3000000000000007  - moves: 8  - final state: 29
rewards: -1.3000000000000007  - moves: 8  - final state: 29
rewards: -1.3000000000000007  - moves: 8  - final state: 29
rewards: -1.3000000000000007  - moves: 8  - final state: 29
```

```
[['Dn' 'Lt' 'Lt' 'Lt' 'Rt' 'Dn' 'Dn']
 ['Dn' 'Lt' 'Lt' 'Lt' 'Lt' 'Dn' 'Dn']
 ['Rt' 'Dn' 'Dn' 'Dn' 'Dn' 'Dn' 'Dn']
 ['Lt' 'Dn' 'Dn' 'Dn' 'Dn' 'Dn' 'Dn']
 ['Rt' 'Lt' 'Lt' 'Lt' 'Lt' 'Lt' 'Lt']]
```

is slippery = true:

```
rewards: -47.199999999999946  - moves: 59  - final state: 29
rewards: -26.49999999999998  - moves: 36  - final state: 29
```

```
rewards: -13.899999999999999  - moves: 22  - final state: 29
rewards: -39.99999999999996  - moves: 51  - final state: 29
rewards: -29.199999999999974  - moves: 39  - final state: 29
```

```
[['Lt' 'UP' 'UP' 'UP' 'UP' 'Rt' 'Lt']
 ['Lt' 'Lt' 'Lt' 'Lt' 'Lt' 'Rt' 'Dn']
 ['UP' 'Dn' 'Dn' 'Dn' 'Dn' 'Dn' 'Dn']
 ['Lt' 'Rt' 'Dn' 'Lt' 'Lt' 'Lt' 'Dn']
 ['Dn' 'Lt' 'Lt' 'Lt' 'Lt' 'Lt' 'Dn']]
```

5- custom map 5, hole reward = -3, goal reward = 7, theta = 0.0001, gamma = 0.9, is slippery = false, move reward = [-4, -2, 0, 2]

move reward = -4:

```
rewards: -9  - moves: 5  - final state: 7
rewards: -9  - moves: 5  - final state: 7
rewards: -9  - moves: 5  - final state: 7
rewards: -9  - moves: 5  - final state: 7
rewards: -9  - moves: 5  - final state: 7
  (Right)
HFSFFFFG


7
[['Lt' 'Rt' 'Rt' 'Rt' 'Rt' 'Rt' 'Rt' 'Lt']]
```

move reward = -2:

```
rewards: -1  - moves: 5  - final state: 7
rewards: -1  - moves: 5  - final state: 7
rewards: -1  - moves: 5  - final state: 7
rewards: -1  - moves: 5  - final state: 7
rewards: -1  - moves: 5  - final state: 7
  (Right)
HFSFFFFG


7
[['Lt' 'Rt' 'Rt' 'Rt' 'Rt' 'Rt' 'Rt' 'Lt']]
```

move reward = 0:

```
rewards: 7  - moves: 5  - final state: 7
rewards: 7  - moves: 5  - final state: 7
rewards: 7  - moves: 5  - final state: 7
rewards: 7  - moves: 5  - final state: 7
rewards: 7  - moves: 5  - final state: 7
  (Right)
HFSFFFFG


7
[['Lt' 'Rt' 'Rt' 'Rt' 'Rt' 'Rt' 'Rt' 'Lt']]
```

move reward = 2:

```
rewards: 15  - moves: 5  - final state: 7
rewards: 15  - moves: 5  - final state: 7
rewards: 15  - moves: 5  - final state: 7
rewards: 15  - moves: 5  - final state: 7
rewards: 15  - moves: 5  - final state: 7
  (Right)
HFSFFFFG


7
[['Lt' 'Rt' 'Rt' 'Rt' 'Rt' 'Rt' 'Rt' 'Lt']]
```

the higher move reward the higher reward is. moves are the same.

6- custom map 6:

```
rewards: 15  - moves: 5  - final state: 7
rewards: 15  - moves: 5  - final state: 7
rewards: 15  - moves: 5  - final state: 7
rewards: 15  - moves: 5  - final state: 7
rewards: 15  - moves: 5  - final state: 7
```

```
[['Lt' 'Rt' 'Rt' 'Rt' 'Rt' 'Rt' 'Rt' 'Lt']
 ['Lt' 'Rt' 'Rt' 'Rt' 'Rt' 'Rt' 'Rt' 'UP']
 ['Lt' 'Rt' 'Rt' 'Rt' 'Rt' 'Rt' 'Rt' 'UP']]
```

7- custom map 7: hole reward = -2, goal reward = 50, move reward = -1, theta=0.0001, gamma = 0.9, is slippery = true

```
rewards: -53  - moves: 104  - final state: 24
rewards: 34  - moves: 17  - final state: 24
rewards: 35  - moves: 16  - final state: 24
rewards: -281  - moves: 332  - final state: 24
rewards: -7  - moves: 58  - final state: 24
  (Down)
SFFFF
FFFFH
HHFFF
HFFFH
FFFFG

50
[['Dn' 'Rt' 'Rt' 'Lt' 'UP']
 ['UP' 'UP' 'Rt' 'Lt' 'Lt']
 ['Lt' 'Lt' 'Rt' 'Lt' 'Dn']
 ['Lt' 'Rt' 'Rt' 'Lt' 'Lt']
 ['Dn' 'Dn' 'Rt' 'Dn' 'Lt']]
```

first visit & every visit:

num episode = 500:

```
first_monto [[-7.35602776 -6.47830592 -5.28132892 -5.21368809  0.        ]
 [-7.0921606  -6.30715114 -3.93463552 -3.06356233  0.        ]
 [ 0.          0.         -0.5122671   0.44499195  0.        ]
 [ 0.          0.          5.3438807   8.12061181  0.        ]
 [ 0.          0.         13.54810064 28.26786294  0.        ]]
every_monto [[-5.37176274 -2.80048077 -1.61437734 -1.47807681  0.        ]
```

```
 [-7.13172338 -2.96582734  2.74711568  2.84203297  0.        ]
 [ 0.          0.          6.71684435  8.65227377  0.        ]
 [ 0.          0.         13.5090411  17.62316716  0.        ]
 [ 0.          0.         21.83669725 35.09784736  0.        ]]
```

num episode = 5000

```
first_monto [[-7.13664506 -6.25843815 -5.10340812 -5.06388734  0.        ]
 [-6.91246838 -5.89031669 -3.57426632 -3.42524124  0.        ]
 [ 0.          0.          0.06023844  0.80476573  0.        ]
 [ 0.          0.          5.85267321  9.43738601  0.        ]
 [ 0.          0.         12.9778138  27.99600457  0.        ]]
every monto [[-4.68818911 -0.17319311  2.08257691  2.45622483  0.        ]
 [-5.01217469 -0.23410646  4.80733053  5.16904663  0.        ]
 [ 0.          0.         10.49632868 11.05022103  0.        ]
 [ 0.          0.         18.73752613 21.34281189  0.        ]
 [ 0.          0.         26.4346116  37.67693249  0.        ]]
```

higher episode number in monte carlo algorithm results in better an accurate perdictions but on the other hand it takes more time.

different results for monte carlo:

because the monte carlo has a little randomness in it's algorithm and this algorithm simulations are stochastic simulations.

8- custom map 8:

```
first_monto [[ 0.         8.96470516  6.68490102  5.0943786   5.21009403  5.61757615
   0.        ]
 [18.65252268 15.68353486 10.70319323  8.34703246 10.66865684 14.19205345
  18.67270356]
 [32.35337106 22.99381058 13.98406689  9.16134059 13.94511878 22.80238638
  32.31565049]
 [ 0.         32.70961832 18.14832214  0.         18.65394349 32.91103732
   0.        ]]
every monto [[ 0.         7.2821299   5.64130936  5.17932684  6.14774898  7.23989073
   0.        ]
 [16.00344799 13.55717559  9.29697867  7.8241588  10.49686431 16.2282934
  18.95870631]
 [32.16635959 22.50928143 14.29500503  9.96742094 14.3221162  23.60430423
```

```
 33.29223774]
[ 0.         32.58113772 18.58101754  0.         18.77666993 34.92210867
  0.        ]]
```

policy left for example:

```
policy left:
(0, {0: 0.0, 1: 0.0, 2: 0.0, 3: 0.0})
(1, {0: 1.0, 1: 0.0, 2: 0.0, 3: 0.0})
(2, {0: 1.0, 1: 0.0, 2: 0.0, 3: 0.0})
(3, {0: 1.0, 1: 0.0, 2: 0.0, 3: 0.0})
(4, {0: 1.0, 1: 0.0, 2: 0.0, 3: 0.0})
(5, {0: 1.0, 1: 0.0, 2: 0.0, 3: 0.0})
(6, {0: 0.0, 1: 0.0, 2: 0.0, 3: 0.0})
(7, {0: 1.0, 1: 0.0, 2: 0.0, 3: 0.0})
(8, {0: 1.0, 1: 0.0, 2: 0.0, 3: 0.0})
(9, {0: 1.0, 1: 0.0, 2: 0.0, 3: 0.0})
(10, {0: 1.0, 1: 0.0, 2: 0.0, 3: 0.0})
```