

UWAGA: Wczytaj do Colab plik **frozen_lake.py** (instrukcja w pliku **COLAB_instrukcja.pdf**)

FrozenLake 1

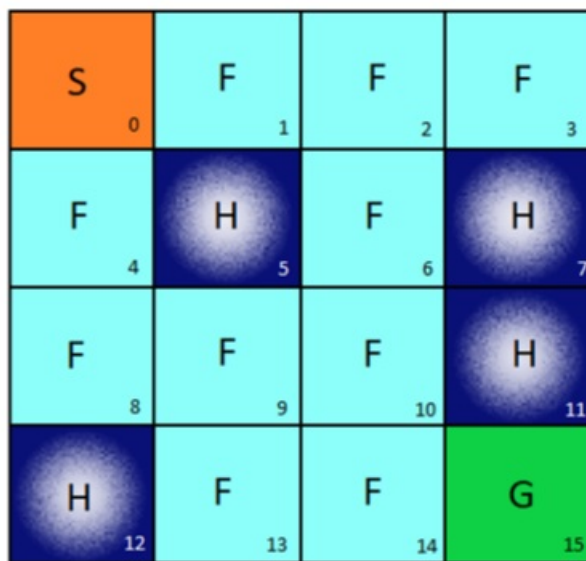
Wprowadzenie

Agent porusza się w świecie złożonym z **16 pól (stanów)**. Stany są ponumerowane od 0 do 15.

Niektóre pola siatki są dostępne do chodzenia (**F**-frozen), a inne są przerębiami (**H**-hole).

Możliwe są 4 akcje: **0 - LEFT**, **1 - DOWN**, **2 - RIGHT**, **3 - UP**

Agent jest nagradzany (**R=1**) za dotarcie do pola **G**. W pozostałych przypadkach **R=0**.



Łaadowanie biblioteki (wcześniej konieczne załadowanie pliku **frozen_lake.py** do Colaba - instrukcja w pliku **PDF**).

In [1]:

```
from frozen_lake import FrozenLakeEnv
```

Wczytanie środowiska:

In [2]:

```
env = FrozenLakeEnv()
```

Sprawdzamy ilość możliwych stanów (16) i akcji (4)

In [3]:

```
print(env.nS)  
print(env.nA)
```

16
4

Dynamika

Dynamika opisana jest za pomocą: `env.P[s][a]`

gdzie: **s** to **stan** (0,1,2,...,15), **a** to **akcja** (0,1,2,3).

Rozważmy przykład: w stanie 0 agent wykonuje akcję 1 (porusza się w dół):

In [4]:

```
env.P[0][1]
```

Out[4]:

```
[(1.0, 4, 0.0, False)]
```

Powyższą czwórkę interpretujemy jako: (**prawdopodobieństwo, nowy stan, nagroda, czy koniec?**).

Czyli w powyższym przykładzie: po wykonaniu w stanie 0 akcji 1 prawdopodobieństwo przejścia do stanu 4 wynosi 1, nagroda 0, agent nie wpadł do przerębli ani nie dotarł do pola G.

Polecenie 1 (do uzupełnienia)

Sprawdź dynamikę dla następujących przypadków:

W **stanie 1** agent **przechodzi w dół**:

In [5]:

```
env.P[1][1]
```

Out[5]:

```
[(1.0, 5, 0.0, True)]
```

W **stanie 10** agent **przechodzi w lewo**:

In [6]:

```
env.P[10][1]
```

Out[6]:

```
[(1.0, 14, 0.0, False)]
```

W **stanie 14** agent **przechodzi w prawo**:

In [7]:

```
env.P[14][1]
```

Out[7]:

```
[(1.0, 14, 0.0, False)]
```

Poruszanie i wizualizacja

W świecie FrozenLake możemy się poruszać wykonując 4 akcje (omówione powyżej). Podgląd położenia uzyskujemy za pomocą `env.render()` (wcześniej resetujemy położenie agenta).

In [8]:

```
env.reset()
env.render()
```

```
SFFF
FHHH
FFFF
HFFG
```

Wykonajmy dwa ruchy w prawo i jeden w dół:

In [9]:

```
env.reset()
env.step(2)
env.step(2)
env.step(1)
env.render()
```

```
(Down)
SFFF
FHHH
FFFF
HFFG
```

Metoda `step` zwraca krotkę (**nowy stan**, **nagroda**, **czy koniec ruchu**,`_`). Koniec następuje wtedy gdy agent wpadł do przerębli lub dotarł do pola 15 - GOAL). Sprawdźmy to.

Z pola początkowego 0 agent rusza się w prawo (akcja - 2) na pole 1 i zdobywa nagrodę 0:

In [10]:

```
env.reset()
env.step(2)
```

Out[10]:

```
(1, 0.0, False, {'prob': 1.0})
```

Agent kontynuuje ruch: rusza się w prawo (akcja - 2) na pole 2 i zdobywa nagrodę 0:

In [11]:

```
env.step(2)
env.render()
```

```
(Right)
SFFF
FHHH
FFFF
HFFG
```

Polecenie 2 (do uzupełnienia)

Przeprowadź agenta dowolną drogą z pola 0 do pola 15 (GOAL). Sprawdź czy nagroda po wejściu na to pole wynosi 1.

In [12]:

```
env.reset()
env.step(2)
env.render()
env.step(2)
env.render()
env.step(1)
env.render()
env.step(1)
env.render()
env.step(1)
env.render()
```

```
env.render()
env.step(2)
```

```
      (Right)
SFFF
FHHF
FFFH
HFFG
      (Right)
SFFF
FHHF
FFFH
HFFG
      (Down)
SFFF
FHHF
FFFH
HFFG
      (Down)
SFFF
FHHF
FFFH
HFFG
      (Down)
SFFF
FHHF
FFFH
HFFG
```

Out[12]:

```
(15, 1.0, True, {'prob': 1.0})
```

Ruch agenta w pętli

Ruch agenta można zapętlić. Na razie akcja w każdym stanie generowana jest losowo (wykorzystujemy metodę:

```
env.action_space.sample()
```

). Agent wykona 10 akcji.

UWAGA: kiedy agent jest na polu oznaczonym **H** (stany 5,7,11,12) **dowolna akcja pozostawia go na tym polu** (agent nie może uciec z przerębli).

In [13]:

```
env.reset()
for i in range(10):
    action = env.action_space.sample()
    obs, rew, fin, _ = env.step(action)
    print("Action=", action, "State =", obs, "Reward =", rew, "End =", fin)
```

```
Action= 3 State = 0 Reward = 0.0 End = False
Action= 1 State = 4 Reward = 0.0 End = False
Action= 0 State = 4 Reward = 0.0 End = False
Action= 0 State = 4 Reward = 0.0 End = False
Action= 2 State = 5 Reward = 0.0 End = True
Action= 0 State = 5 Reward = 0 End = True
Action= 0 State = 5 Reward = 0 End = True
Action= 2 State = 5 Reward = 0 End = True
Action= 2 State = 5 Reward = 0 End = True
Action= 3 State = 5 Reward = 0 End = True
```

Polecenie 3 (do uzupełnienia)

Sprawdź czy **możliwe jest dotarcie agenta do pola G** w przypadku gdy **akcje są generowane losowo**. Przeprowadź dużą liczbę testów (zbuduj odpowiednią pętlę). Zawsze gdy agent wpadnie do przerębli przerwij pętlę.

Poniżej wpisz kod:

In [14]:

```
env.reset()
for _ in range(100):
    #print(env.action_space.sample())
    print("=====\n")
    fin = False
    env.reset()
    while fin == False:
        action = env.action_space.sample()
        obs, rew, fin, is_end = env.step(action)
        print("Action=",action,"State =",obs,"Reward =",rew,"End =",fin)
        #if rew == 1.0:
            #print("Action=",action,"State =",obs,"Reward =",rew,"End =",fin)
```

```
=====
Action= 2 State = 1 Reward = 0.0 End = False
Action= 2 State = 2 Reward = 0.0 End = False
Action= 2 State = 3 Reward = 0.0 End = False
Action= 1 State = 7 Reward = 0.0 End = True
=====
```

```
Action= 3 State = 0 Reward = 0.0 End = False
Action= 2 State = 1 Reward = 0.0 End = False
Action= 3 State = 1 Reward = 0.0 End = False
Action= 1 State = 5 Reward = 0.0 End = True
=====
```

```
Action= 2 State = 1 Reward = 0.0 End = False
Action= 0 State = 0 Reward = 0.0 End = False
Action= 3 State = 0 Reward = 0.0 End = False
Action= 3 State = 0 Reward = 0.0 End = False
Action= 0 State = 0 Reward = 0.0 End = False
Action= 3 State = 0 Reward = 0.0 End = False
Action= 2 State = 1 Reward = 0.0 End = False
Action= 1 State = 5 Reward = 0.0 End = True
=====
```

```
Action= 1 State = 4 Reward = 0.0 End = False
Action= 2 State = 5 Reward = 0.0 End = True
=====
```

```
Action= 3 State = 0 Reward = 0.0 End = False
Action= 3 State = 0 Reward = 0.0 End = False
Action= 1 State = 4 Reward = 0.0 End = False
Action= 3 State = 0 Reward = 0.0 End = False
Action= 1 State = 4 Reward = 0.0 End = False
Action= 1 State = 8 Reward = 0.0 End = False
Action= 1 State = 12 Reward = 0.0 End = True
=====
```

```
Action= 3 State = 0 Reward = 0.0 End = False
Action= 1 State = 4 Reward = 0.0 End = False
Action= 2 State = 5 Reward = 0.0 End = True
=====
```

```
Action= 1 State = 4 Reward = 0.0 End = False
Action= 2 State = 5 Reward = 0.0 End = True
=====
```

```
Action= 0 State = 0 Reward = 0.0 End = False
Action= 3 State = 0 Reward = 0.0 End = False
Action= 0 State = 0 Reward = 0.0 End = False
Action= 1 State = 4 Reward = 0.0 End = False
Action= 1 State = 8 Reward = 0.0 End = False
Action= 2 State = 9 Reward = 0.0 End = False
Action= 2 State = 10 Reward = 0.0 End = False
Action= 1 State = 14 Reward = 0.0 End = False
Action= 3 State = 10 Reward = 0.0 End = False
Action= 0 State = 9 Reward = 0.0 End = False
Action= 0 State = 8 Reward = 0.0 End = False
Action= 2 State = 9 Reward = 0.0 End = False
Action= 2 State = 10 Reward = 0.0 End = False
Action= 2 State = 11 Reward = 0.0 End = True
=====
```

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]


```
Action= 0 State = 0 Reward = 0.0 End = False
Action= 0 State = 0 Reward = 0.0 End = False
Action= 2 State = 1 Reward = 0.0 End = False
Action= 2 State = 2 Reward = 0.0 End = False
Action= 3 State = 2 Reward = 0.0 End = False
Action= 0 State = 1 Reward = 0.0 End = False
Action= 3 State = 1 Reward = 0.0 End = False
Action= 2 State = 2 Reward = 0.0 End = False
Action= 0 State = 1 Reward = 0.0 End = False
Action= 1 State = 5 Reward = 0.0 End = True
=====
```

```
Action= 0 State = 0 Reward = 0.0 End = False
Action= 3 State = 0 Reward = 0.0 End = False
Action= 0 State = 0 Reward = 0.0 End = False
Action= 1 State = 4 Reward = 0.0 End = False
Action= 3 State = 0 Reward = 0.0 End = False
Action= 2 State = 1 Reward = 0.0 End = False
Action= 3 State = 1 Reward = 0.0 End = False
Action= 1 State = 5 Reward = 0.0 End = True
=====
```

```
Action= 3 State = 0 Reward = 0.0 End = False
Action= 0 State = 0 Reward = 0.0 End = False
Action= 1 State = 4 Reward = 0.0 End = False
Action= 1 State = 8 Reward = 0.0 End = False
Action= 0 State = 8 Reward = 0.0 End = False
Action= 2 State = 9 Reward = 0.0 End = False
Action= 3 State = 5 Reward = 0.0 End = True
=====
```

```
Action= 2 State = 1 Reward = 0.0 End = False
Action= 2 State = 2 Reward = 0.0 End = False
Action= 2 State = 3 Reward = 0.0 End = False
Action= 0 State = 2 Reward = 0.0 End = False
Action= 3 State = 2 Reward = 0.0 End = False
Action= 1 State = 6 Reward = 0.0 End = False
Action= 0 State = 5 Reward = 0.0 End = True
=====
```

```
Action= 3 State = 0 Reward = 0.0 End = False
Action= 3 State = 0 Reward = 0.0 End = False
Action= 1 State = 4 Reward = 0.0 End = False
Action= 3 State = 0 Reward = 0.0 End = False
Action= 3 State = 0 Reward = 0.0 End = False
Action= 2 State = 1 Reward = 0.0 End = False
Action= 3 State = 1 Reward = 0.0 End = False
Action= 3 State = 1 Reward = 0.0 End = False
Action= 0 State = 0 Reward = 0.0 End = False
Action= 3 State = 0 Reward = 0.0 End = False
Action= 0 State = 0 Reward = 0.0 End = False
Action= 0 State = 0 Reward = 0.0 End = False
Action= 2 State = 1 Reward = 0.0 End = False
Action= 0 State = 0 Reward = 0.0 End = False
Action= 2 State = 1 Reward = 0.0 End = False
Action= 1 State = 5 Reward = 0.0 End = True
=====
```

```
Action= 0 State = 0 Reward = 0.0 End = False
Action= 2 State = 1 Reward = 0.0 End = False
Action= 0 State = 0 Reward = 0.0 End = False
Action= 0 State = 0 Reward = 0.0 End = False
Action= 1 State = 4 Reward = 0.0 End = False
Action= 2 State = 5 Reward = 0.0 End = True
```

TWOJE PODSUMOWANIE TESTÓW:

In [15]:

```
#Jest mozliwe dotarcie do stanu 15 przy losowym generowaniu akcji
```

```
Action= 0 State = 0 Reward = 0.0 End = False
Action= 0 State = 0 Reward = 0.0 End = False
Action= 1 State = 4 Reward = 0.0 End = False
Action= 3 State = 0 Reward = 0.0 End = False
```

```
Action= 3 State = 0 Reward = 0.0 End = False
Action= 1 State = 4 Reward = 0.0 End = False
Action= 1 State = 8 Reward = 0.0 End = False
Action= 2 State = 9 Reward = 0.0 End = False
Action= 1 State = 13 Reward = 0.0 End = False
Action= 3 State = 9 Reward = 0.0 End = False
Action= 2 State = 10 Reward = 0.0 End = False
Action= 1 State = 14 Reward = 0.0 End = False
Action= 0 State = 13 Reward = 0.0 End = False
Action= 3 State = 9 Reward = 0.0 End = False
Action= 2 State = 10 Reward = 0.0 End = False
Action= 1 State = 14 Reward = 0.0 End = False
Action= 2 State = 15 Reward = 1.0 End = True
```

File "<ipython-input-15-e18059bf9586>", line 3

```
Action= 0 State = 0 Reward = 0.0 End = False
```

^

SyntaxError: invalid syntax