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

FrozenLake 3

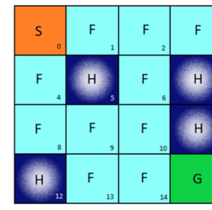
In [1]:
```
from frozen_lake import FrozenLakeEnv
#from frozen_lake_slippery import FrozenLakeEnv
import numpy as np

env = FrozenLakeEnv()
```

Chcemy napisać funkcję, która korzystając z określonych wartości zwrótów ***V(s)*** (dla wszystkich stanów **s**) zwróci wartości ***Q(s,a)*** dla konkretnego stanu **s** i dla wszystkich akcji **a** możliwych do wykonania w stanie **s**.

Załóżmy, że mamy dane ***V(s)*** takie jak na rysunku poniżej:

V(s)

0.16807	0.2401	0.343	0.2401
0.2401	0.	0.49	0.
0.343	0.49	0.7	0.
0.	0.7	1.	0.



Wartości zwrotów ***V(s)*** dla każdego stanu zapiszemy w tablicy:

In [2]:
```
V = np.array([0.16807,0.2401,0.343,0.2401,0.2401,0.,0.49,0.,0.343,0.49,
0.7,0.,0.,0.7,1.,0.])
print(V)
```

```
[0.16807 0.2401  0.343   0.2401  0.2401  0.      0.49    0.      0.343
 0.49    0.7     0.      0.      0.7     1.      0.      ]
```

Funkcję zdefiniujemy korzystając z formuły:

$$q_\pi(s,a) \quad = \quad \sum_{s',r} p(s',r|s,a)\Big[r + \gamma v_\pi(s')\Big]$$

Polecenie 1 (do uzupełnienia)

Funkcja dla danego **s** i znanego ***V*** ma zwracać wartości zwrotów **dla czterech akcji** możliwych do wykonanania w stanie **s**. Czyli może wyglądać tak (**UZUPEŁNIJ DEFINICJĘ FUNKCJI**):

In [6]:
```
def Q_from_V(env, V, s, gamma=0.99):
    Q = np.zeros(env.nA)
    for action in range(env.nA):
        for next_state in range(len(env.P[s][action])):
            prob, next_state, reward, done = env.P[s][action][next_stat
e]
            Q[action] += prob * (reward + gamma * V[next_state])


    return Q
```

OBJAŚNIENIE: Argumenty funkcji (oprócz ***V*** i **s**) to zmienna ***env*** związana ze środowiskiem ***FrozenLake*** i wartość ***gamma***, która występuje w powyższym wzorze. ***Q*** zdefiniowane w pierwszej linijce definicji to **4 elementowa tablica złożona z zer** (**env.nA** to ilość akcji, które można wykonać w środowisku określonym przez **env**). W pętli **for** mają być wyliczone **wartości zwrotów dla każdej z czterech akcji 0,1,2,3**. Wartości te mają być zapisane w tablicy ***Q***. Funkcja zwróci tę tablicę.

Polecenie 2 (do uzupełnienia)

Przetestuj działanie funkcji ***Q_from_V*** dla domyślnej wartości **gamma=0,99**

Wartości zwrotów dla 4 akcji w stanie ***s=0***:

In [9]:
```
#zwrot dla akcji 0
Q_from_V(env,V,0)
#zwrot dla akcji 1
Q_from_V(env,V,0)
#zwrot dla akcji 2
Q_from_V(env,V,0)
#zwrot dla akcji 3
Q_from_V(env,V,0)
```

Out[9]:
```
array([0.1663893, 0.237699 , 0.237699 , 0.1663893])
```

Wartości zwrotów dla 4 akcji w stanie ***s=8***:

In [10]:
```
#zwrot dla akcji 0
Q_from_V(env,V,8)
#zwrot dla akcji 1
Q_from_V(env,V,8)
#zwrot dla akcji 2
Q_from_V(env,V,8)
#zwrot dla akcji 3
Q_from_V(env,V,8)
```

Out[10]:
```
array([0.33957 , 0.      , 0.4851  , 0.237699])
```

Wartości zwrotów dla 4 akcji w stanie ***s=15***:

In [13]:
```
#zwrot dla akcji 0
Q_from_V(env,V,15)
#zwrot dla akcji 1
Q_from_V(env,V,15)
#zwrot dla akcji 2
Q_from_V(env,V,15)
#zwrot dla akcji 3
Q_from_V(env,V,15)
```

Out[13]:
```
array([0., 0., 0., 0.])
```

Przetestuj działanie funkcji ***Q_from_V*** dla mniejszej wartości **gamma=0.1**

Wartości zwrotów dla 4 akcji w stanie **s=0**:

In [14]:
```
#zwrot dla akcji 0
Q_from_V(env,V,0,0.1)
#zwrot dla akcji 1
Q_from_V(env,V,0,0.1)
#zwrot dla akcji 2
Q_from_V(env,V,0,0.1)
#zwrot dla akcji 3
Q_from_V(env,V,0,0.1)
```

Out[14]:
```
array([0.016807, 0.02401 , 0.02401 , 0.016807])
```

Wartości zwrotów dla 4 akcji w stanie ***s=8***:

In [15]:
```
#zwrot dla akcji 0
Q_from_V(env,V,8,0.1)
#zwrot dla akcji 1
Q_from_V(env,V,8,0.1)
#zwrot dla akcji 2
Q_from_V(env,V,8,0.1)
#zwrot dla akcji 3
Q_from_V(env,V,8,0.1)
```

Out[15]:
```
array([0.0343 , 0.     , 0.049  , 0.02401])
```

Wartości zwrotów dla 4 akcji w stanie ***s=15***:

In [16]:
```
#zwrot dla akcji 0
Q_from_V(env,V,15,0.1)
#zwrot dla akcji 1
Q_from_V(env,V,15,0.1)
#zwrot dla akcji 2
Q_from_V(env,V,15,0.1)
#zwrot dla akcji 3
Q_from_V(env,V,15,0.1)
```

Out[16]:
```
array([0., 0., 0., 0.])
```

Polecenie 3 (do uzupełnienia)

Jaki wpływ na wyniki miała zmiana wartości parametru **gamma** i dlaczego taki?

WPISZ ODPOWIEDŹ:jesli ustamimy mala gamme to agent patrzy tylko na pierwsza nagrode, a jesli ustawimy ja na wieksza to bedzie dazyc do tego by na koniec miec najwieksza nagrode. Ogolnie chodzi o to ze czym mniejsza gamma tym agent mniej patrzy w przyszla rozgrywke.