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

FrozenLake 3

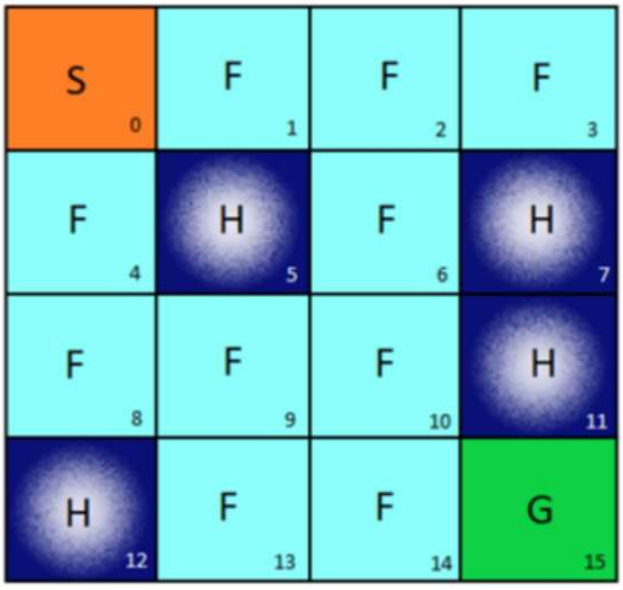
```
import gym
import numpy as np

env = gym.make("FrozenLake-v0")
```

Chcemy napisać funkcję **Q**, 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:

```
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:

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

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**):

```
def Q_from_V(env, V, s, gamma=0.99):
    Q = np.zeros(env.nA)

    for action in range(env.nA):
      action_value = 0
      for i in range(len(env.P[s][action])):
        prob, next_state, r, _ = env.P[s][action][i]
        action_value += prob * (r + gamma * V[next_state])
     # Q.append(action_value)
      Q[action]= action_value
   # Q = np.argmax(np.asarray(action_values))



      #DO UZUPEŁNIENIA

    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 tę 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***:

```
X= Q_from_V(env, V, 0)
```

```
#zwrot dla akcji 0
#zwrot dla akcji 1
#zwrot dla akcji 2
#zwrot dla akcji 3


print(X)
```

```
[0.1901592 0.2139291 0.2139291 0.1901592]
```

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

```
X= Q_from_V(env, V, 8)
print(X)
#zwrot dla akcji 0
#zwrot dla akcji 1
#zwrot dla akcji 2
#zwrot dla akcji 3
```

```
[0.192423 0.27489  0.240933 0.354123]
```

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

```
X= Q_from_V(env, V, 15)
print(X)
#zwrot dla akcji 0
#zwrot dla akcji 1
#zwrot dla akcji 2
#zwrot dla akcji 3
```

```
[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***:

```
X= Q_from_V(env, V, 0, 0.1)
print(X)
#zwrot dla akcji 0
#zwrot dla akcji 1
#zwrot dla akcji 2
#zwrot dla akcji 3
```

```
[0.019208 0.021609 0.021609 0.019208]
```

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

```
X= Q_from_V(env, V, 8,0.1)
```

```
print(X)
#zwrot dla akcji 0
#zwrot dla akcji 1
#zwrot dla akcji 2
#zwrot dla akcji 3
```

```
[0.01943667 0.02776667 0.02433667 0.03577   ]
```

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

```
X= Q_from_V(env, V, 15)
print(X)
#zwrot dla akcji 0
#zwrot dla akcji 1
#zwrot dla akcji 2
#zwrot dla akcji 3
```

```
[0. 0. 0. 0.]
```

Polecenie 3 (do uzupełnienia)

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

WPISZ ODPOWIEDŹ:

Zmiejsza nam wynikowe Q od V na wielokrotnie mniejsze