Задание 3.

Task_1.
Я провел ряд эксперементов с изменением gamma параметра при разных значениях iter_n и _eval_iten_n параметров. По графику видно, что корреляция gamma параметра и mean_total_rewards не особо заметна. Однако, при gamma=0.999 Алгоритм работает подругому, улучшая свою среднюю метрику.

Результаты:

[{'gamma': 0.03, 'mean_total_rewards': 0.751},
{'gamma': 0.06, 'mean_total_rewards': 0.733},
{'gamma': 0.09, 'mean_total_rewards': 0.732},
{'gamma': 0.12, 'mean_total_rewards': 0.716},
{'gamma': 0.15, 'mean_total_rewards': 0.724},
{'gamma': 0.18, 'mean_total_rewards': 0.725},
{'gamma': 0.21, 'mean_total_rewards': 0.718},
{'gamma': 0.24, 'mean_total_rewards': 0.736},
{'gamma': 0.27, 'mean_total_rewards': 0.752},
{'gamma': 0.30000000000000004, 'mean_total_rewards': 0.741},
{'gamma': 0.32999999999999996, 'mean_total_rewards': 0.723},
{'gamma': 0.36, 'mean_total_rewards': 0.724},
{'gamma': 0.39, 'mean_total_rewards': 0.732},
{'gamma': 0.42000000000000004, 'mean_total_rewards': 0.75},
{'gamma': 0.44999999999999996, 'mean_total_rewards': 0.739},
{'gamma': 0.48, 'mean_total_rewards': 0.767},
{'gamma': 0.51, 'mean_total_rewards': 0.771},
{'gamma': 0.54, 'mean_total_rewards': 0.753},
{'gamma': 0.5700000000000001, 'mean_total_rewards': 0.744},
{'gamma': 0.6, 'mean_total_rewards': 0.743},
{'gamma': 0.63, 'mean_total_rewards': 0.702},
{'gamma': 0.66, 'mean_total_rewards': 0.723},
{'gamma': 0.69, 'mean_total_rewards': 0.735},
{'gamma': 0.72, 'mean_total_rewards': 0.742},
{'gamma': 0.75, 'mean_total_rewards': 0.726},
{'gamma': 0.78, 'mean_total_rewards': 0.729},
{'gamma': 0.81, 'mean_total_rewards': 0.725},
{'gamma': 0.84, 'mean_total_rewards': 0.753},
{'gamma': 0.87, 'mean_total_rewards': 0.739},
{'gamma': 0.9, 'mean_total_rewards': 0.75},
{'gamma': 0.9299999999999999, 'mean_total_rewards': 0.741},
{'gamma': 0.96, 'mean_total_rewards': 0.735},
{'gamma': 0.99, 'mean_total_rewards': 0.866}
]

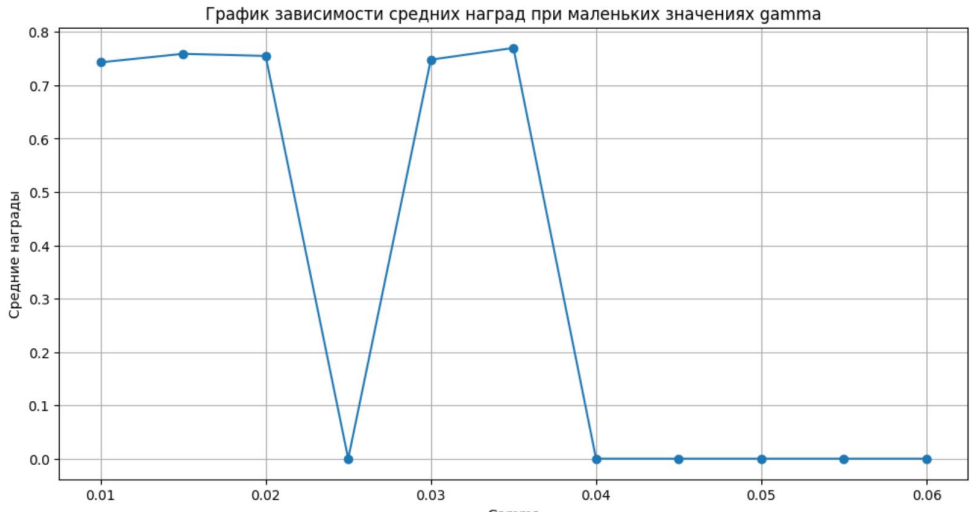
Графически:

Task 2.
Я переписал политику обновления на каждом шаге с учетом значений, получених на прошлой итерации.

```
def policy_evaluation_step(env, v_values, policy, gamma):
    """

    """
    q_values = get_q_values(env, v_values, gamma)

    for state in env.get_all_states():

        for action in env.get_possible_actions(state):
            v_values[state] += policy[state][action] * q_values[state][action]
    # print(v_values)
    return v_values
```

После этого алгоритм перестал сходится, и стал показывать среднюю награду 0. При gamma > 0.03. Но при маленьких значениях gamma алгоритм сходится «через раз» тоже

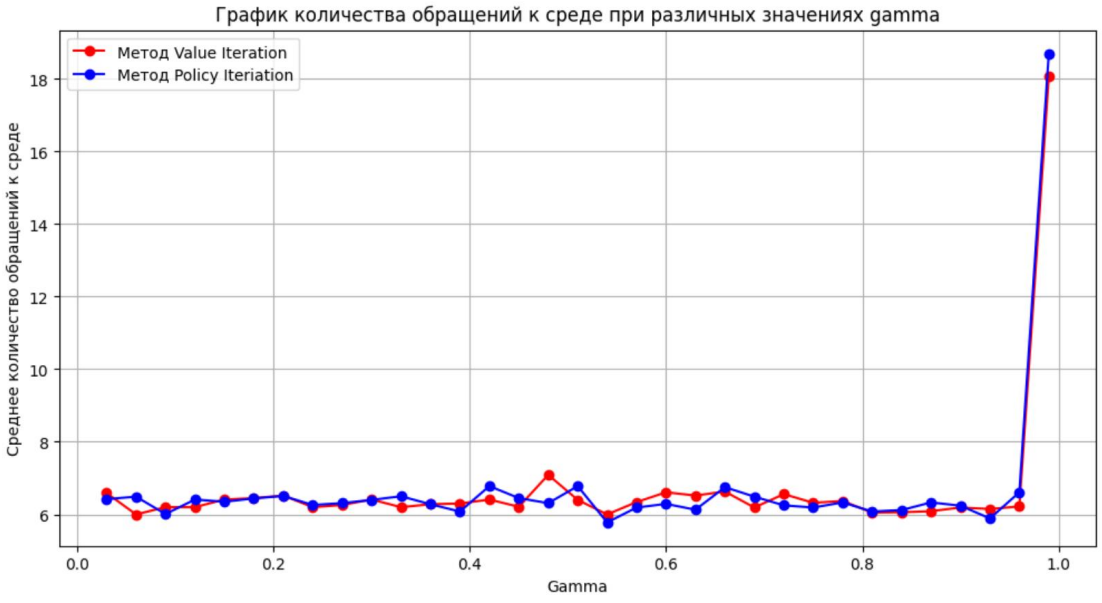
Если я правильно понял, инициализируя значение value 0ми, мы их делаем независимыми для каждого отдельного состояния. В случае, если я использую старую, то я ищу зависимости между состояниями (значение валью в состояние Sn зависит от состояния S(n-1), что не корректно.



Task 3.
Я переписал методы police на новые.

```
1 usage
def value_iteration_step(env, v_values, gamma):
    env_step_count = 0
    new_v_values = init_v_values(env)
    q_values = get_q_values(env, v_values, gamma)
    for state in env.get_all_states():
        if env.get_possible_actions(state):
            max_q_value = max(q_values[state].values())
            new_v_values[state] = max_q_value
            env_step_count += 1
    return new_v_values, env_step_count


1 usage
def value_iteration(env, gamma, iter_n=100):
    v_values = init_v_values(env)
    env_iter_count = 0
    for _ in range(iter_n):
        v_values, env_step_count = value_iteration_step(env, v_values, gamma)
        env_iter_count += env_step_count
    return v_values, env_iter_count
```

Далее я добавил счетчик обращений к среде на инференсе, и сравнил результаты для двух моделей. (базовой модели, и моей модели, использующей максимальные значения по итерациям).



По графику сложно оценить результаты моделей поэтому проанализируем статически.

```
import pandas as pd
import numpy as np
```

```
data = pd.DataFrame({'value_iter': lst_v,
                     "policy_iter":lst_q})
```

```
data['value_is_better'] = np.where(
    data.value_iter - data.policy_iter < 0,
    "Value_iter - better!",
    "Policy_iter - better!"
    )
data['value_is_better'].value_counts()
```

```
Policy_iter - better!    17
Value_iter - better!     16
```

Получилось, что при разных значениях гамма, различные методы являются лучшими для изучения среды.