iniciar\_Q(); //Iniciar o valor da estimativa Q(s,a) para todo os estados **s** e ações **a**

Estado s = observa\_estado() //Inicializa s com o valor do estado atual

cond\_paragem = avaliar\_estado( **s** );

while( !cond\_paragem ){

for( cada episodio de decisão ){

// Escolhe a ação com maior recompensa, considerando todas as ações possíveis de

// executar naquele instante, para o estado **s,** e segundo a politica **π** que utiliza

// as estimativas da função Q para maximizar a soma total de recompensas

Ação ***escolha*** = seleciona\_Acao\_Q(**s**);

executa( ***escolha*** ); // Executa a ação no ambiente;

Recompensa ***r*** = recebe\_Recompensa() // Recebe a recompensa por executar a sua ação;

atualiza\_Q(**s**, ***escolha***, ***r***); // Atualiza o valor que Q(s,a) com o valor da

// recompensa efetivamente recebida;

**s** = observa\_estado(); // Observa o novo estado para o qual o ambiente

// transitou após executar a ação;

cond\_paragem = avaliar\_estado(**s**); // Verificar se **s** é um estado terminal ou se

// foi atingido o número de iterações máximo;

}

}

iniciar\_Q(); //Iniciar o valor da estimativa Q(s,a) para todo os estados **s** e ações **a**

Estado s = observa\_estado() //Inicializa s com o valor do estado atual

Cond\_paragem = avaliar\_estado( **s** );

while( !cond\_paragem ){

// Escolhe a ação com maior recompensa, considerando todas as ações possíveis de

// executar naquele instante, para o estado **s,** e segundo a politica **π** que utiliza

// as estimativas da função Q para maximizar a soma total de recompensas

Ação ***escolha*** = seleciona\_Acao\_Q(**s**);

for( cada episodio de decisão ){

executa( ***escolha*** ); // Executa a ação no ambiente;

Recompensa ***r*** = recebe\_Recompensa() // Recebe a recompensa por executar a sua ação; ´

**s’** = observa\_estado(); // Observa o novo estado para o qual o ambiente

// transitou após executar a ação;

Ação ***next*** = seleciona\_Acao\_Q(**s’**); // Para o novo estado, determina qual a melhor

// ação seguinte segundo a politica atual;

// Atualiza o valor que Q(s,a) com o valor da recompensa efetivamente recebida

// assim como o próximo estado e decisão ***next*** a tomar.

atualiza\_Q(**s**, ***escolha***, ***r***, **s’**, ***next***);

**s** = **s’**; **escolha** = **next**; // Define que a próxima escolha a tomar será

// a que foi determinada para o novo estado;

cond\_paragem = avaliar\_estado(**s**); // Verificar se **s** é um estado terminal ou se

// foi atingido o número de iterações máximo;

}

}