1) A heurística H1 mede a distância de Manhattan entre o nó atual é o nó objetivo. A dintância de Manhattan é dada pela fórmula $|x_1 - x_2| + |y_1 - y_2|$, dadas duas cordenadas (x_1, x_2) e (y_1, y_2).

```
public Double heuristc_one(ArrayList<Estado> list){
    if(list == null){
        return (double) 0;
    }

    double cost = (Math.abs(list.get(0).getCol() - list.get(1).getCol()) +
            Math.abs(list.get(0).getLin() - list.get(1).getLin()));

    return cost;
}
```

A função recebe uma lista que contém dois estados e calcular a distância de Manhattan entre eles. Essa função é chamada pelo método *search* que monta a árvore de busca.
Esta heurística é admissível pois retorna exatamente a distância em “blocos” que o nó atual está do nó objetivo. Ela também é consistente pois o custo dos filho de um nó é sempre maior que o custo dele próprio.

2) A heurística dois calcula a distância euclidiana entre os dois estados que foram passados por parâmetro. Ela é dada pela fórmula: $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$, dadas duas cordenadas (x_1, x_2) e (y_1, y_2).

```
public Double heuristic_two(ArrayList<Estado> list){
    if(list == null){
        return (double) 0;
    }

    int x1 = list.get(0).getCol();
    int y1 = list.get(0).getLin();
    int x2 = list.get(1).getCol();
    int y2 = list.get(1).getLin();

    double cost = Math.sqrt( Math.pow((x1 - x2), 2) + Math.pow((y1 - y2), 2) );
    return cost;
}
```

A função recebe uma lista que contém dois estados e calcular a distância euclidiana entre eles. Essa função é chamada pelo método *search* que monta a árvore de busca.
Esta heurítica é admissível pois a distância euclidiana é sempre menor ou igual que a distância real. Ela também é consistente pois a g(n) de um dado nó nunca é menor que o g(n) de seu pai.

3) Dominancia significa que: $\forall n, h_2(n) \geq h_1(n)$, ou seja, que determinada função heurística é sempre melhor (mais eficiente) qualquer seja o tamanho do problema. Neste caso, a heurística dominante é h_2.

4)

Estratégia	ct_ja_explorado	ct_descartado	Total	N nós memória	Solução	Custo
Uniforme	149	-	199	9	N, N, N, NE, L, L, L, L, NE, NE, L	12,5
H1	73	-	112	14	N, N, N, NE, L, L, L, L, NE, NE, L	12,5
H2	61	-	94	15	N, N, N, NE, L, L, L, L, NE, NE, L	12,5

5) Todos os métodos de busca acham a solução ótima. Eles também são completos, ou seja, caso haja a solução, ela sempre será encontrada. Os resultados são compatíveis com o que se esperava de cada heurística. A primeira tem que explorar e criar muito mais nós para achar a solução ótima. Já as última duas, que utilizaram heurística, conseguem achar a solução com mais rapidez por se basearem em uma função que as ajuda a encontrar a solução ótima da forma mais rápida.

6)

a) O algoritmo avisa que nenhuma solução foi encontrada
b) Sim, pois não entra em loop caso a solução não seja encontrada
c) 54