Projeto Inteligência Artificial – Prof. Murilo Naldi

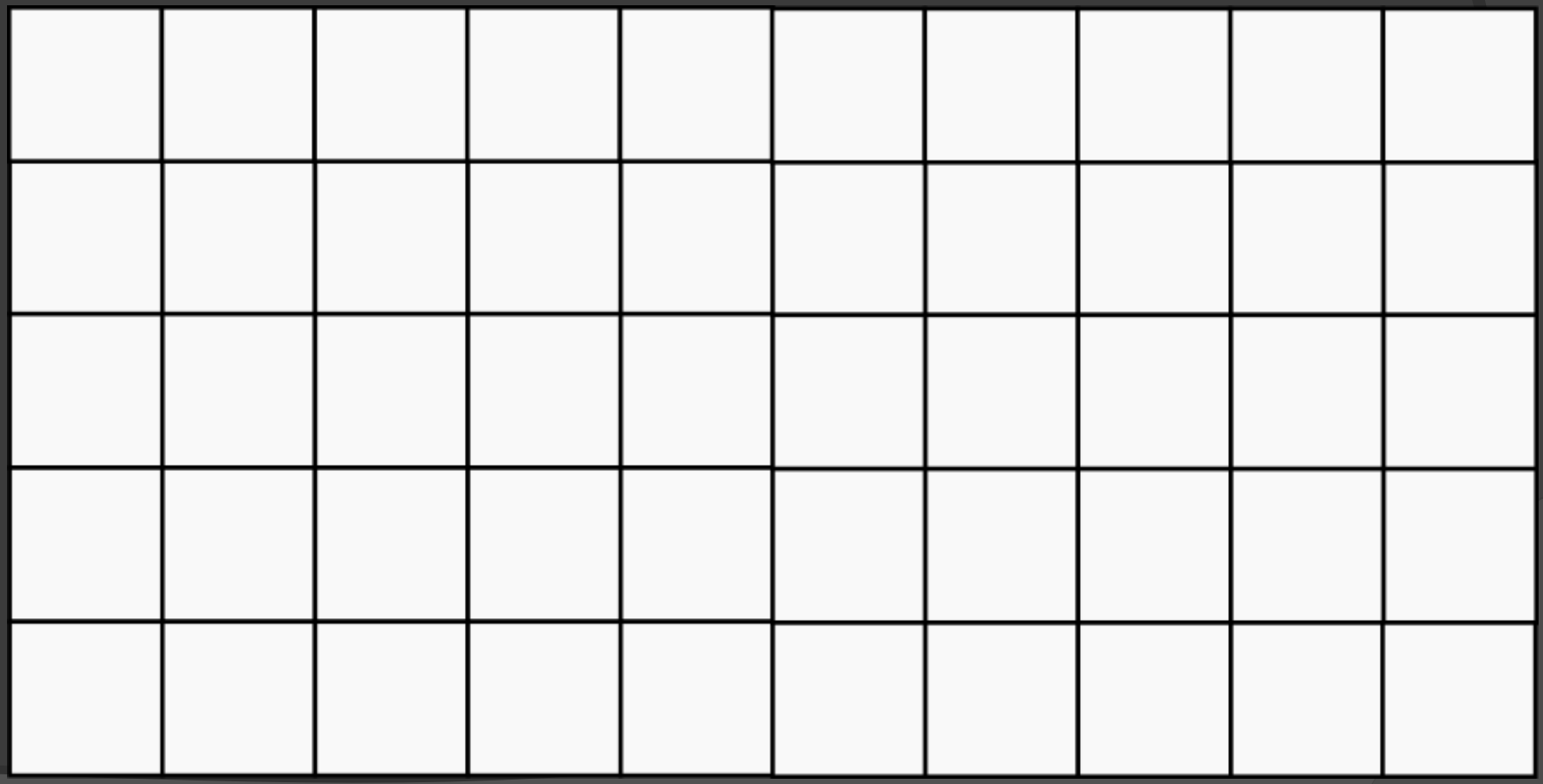
Agente Aspirador de pó

Motivação

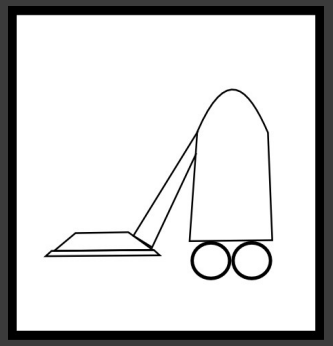
- Fazer um programa que seja capaz de gerar o caminho a ser percorrido por um agente aspirador de pó
- O caminho deve conter todas as sujeiras do cenário, passar pela lixeira e terminar no *dock station*

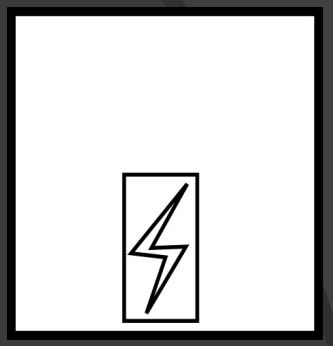
Ambiente

- Modelagem do ambiente em forma de um galpão: 5 andares X 10 espaços



Agente

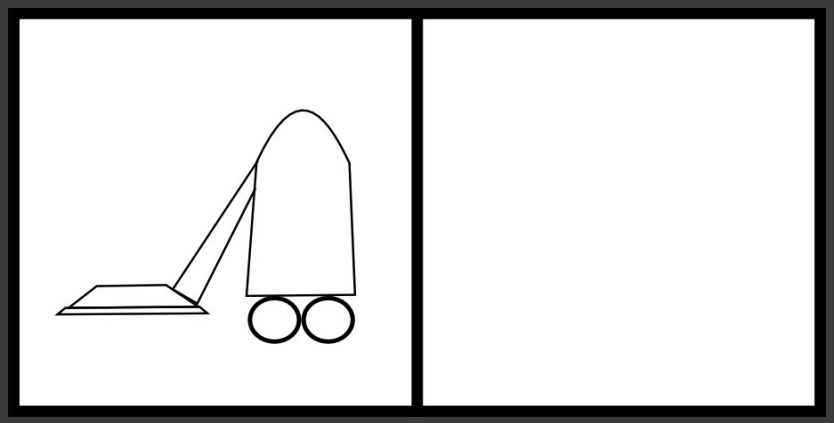


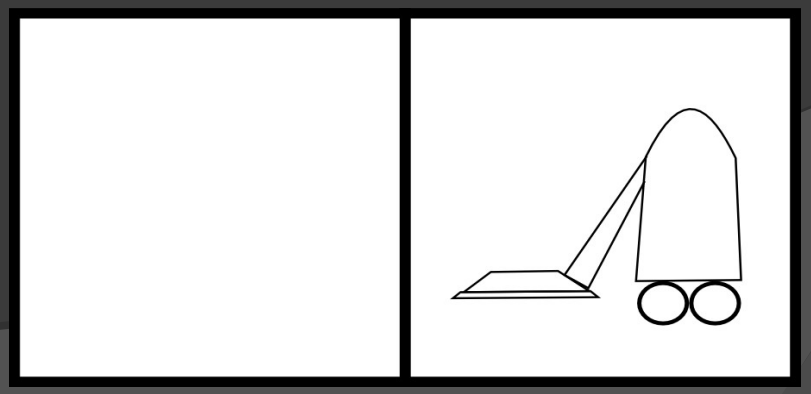


- AADP
  - Define a posição inicial do problema de busca
  - Qualquer posição do ambiente
- *Dock station*
  - Define a posição final do problema de busca, onde o aadp vai recarregar
  - Qualquer posição do ambiente

Movimentação

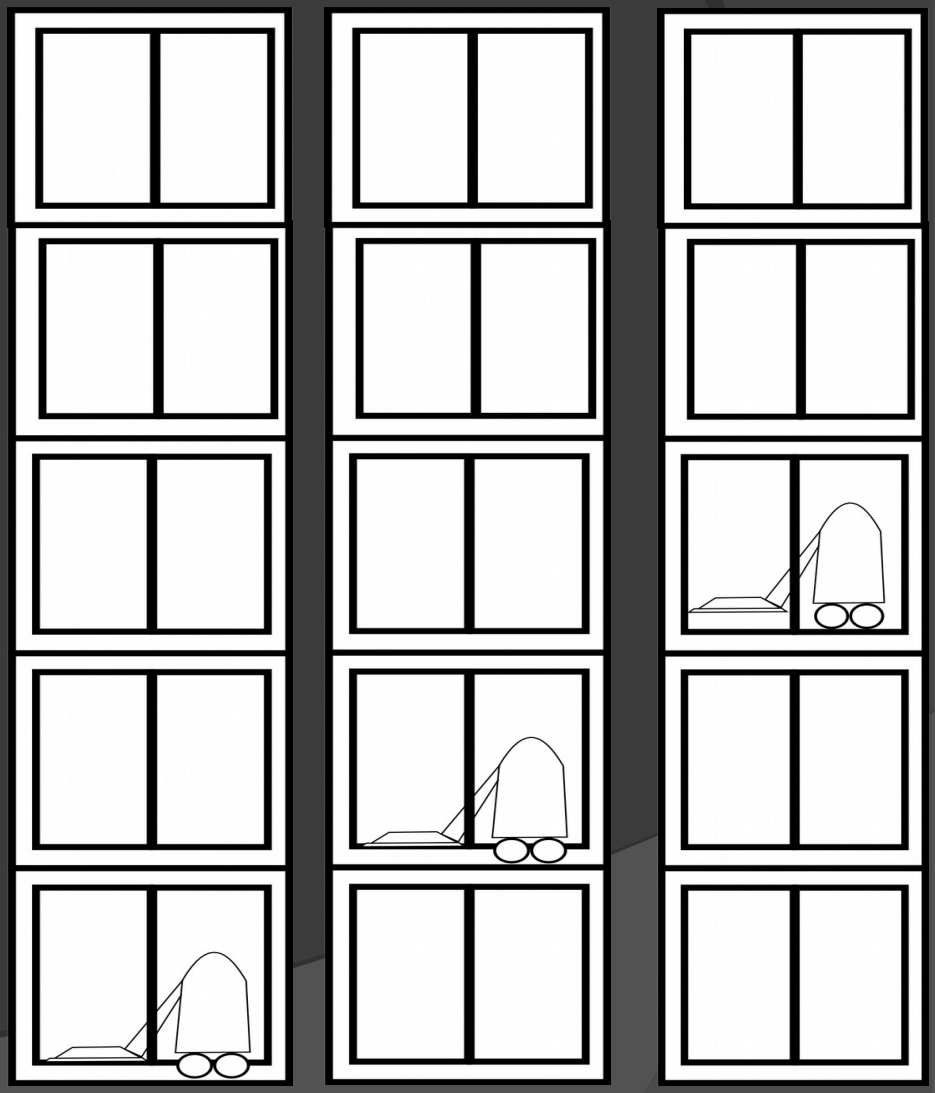
- Movimentação é livre em um mesmo andar (horizontal)





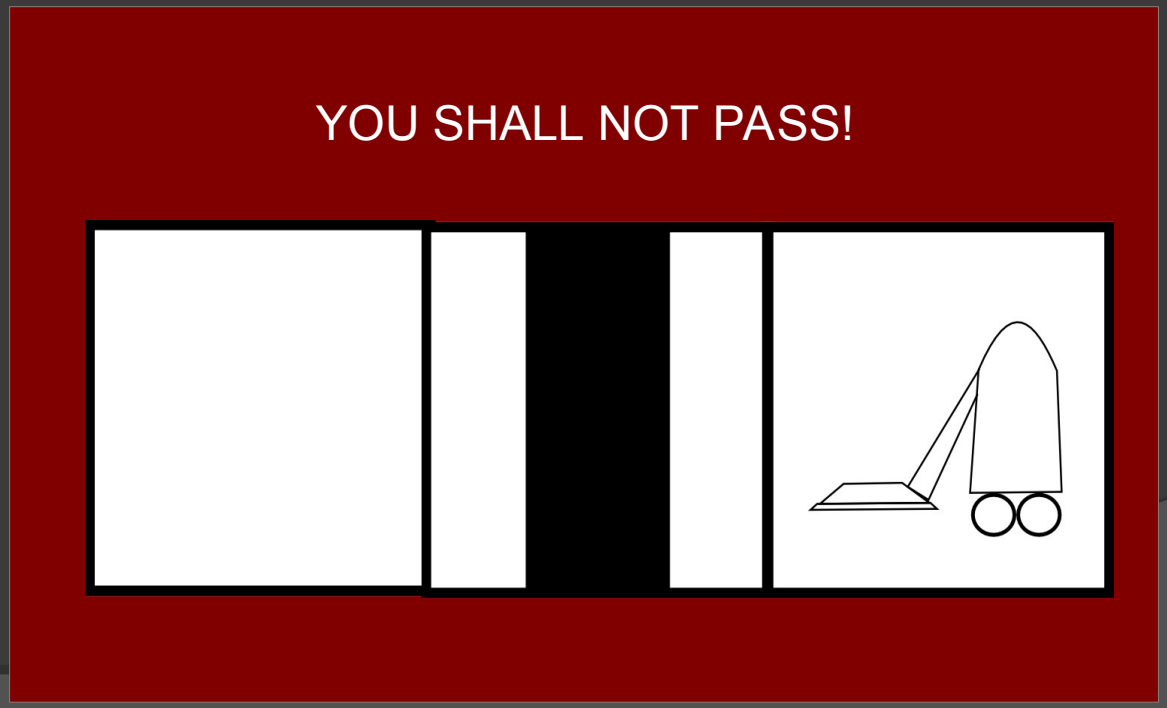
Movimentação

- Movimentação entre andares só é possível se houver um elevador (vertical), podendo se movimentar entre todos os andares.



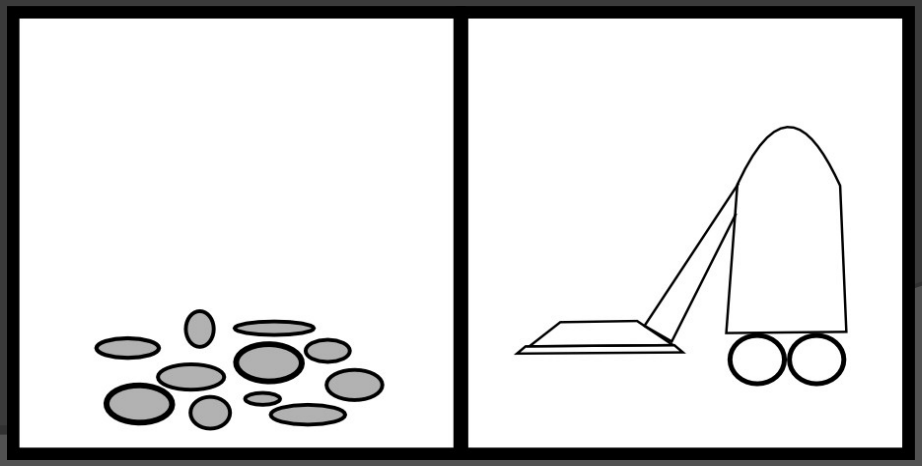
Bloqueio

- O espaços podem conter paredes, que impossibilitam a passagem do aadp



Sujeira

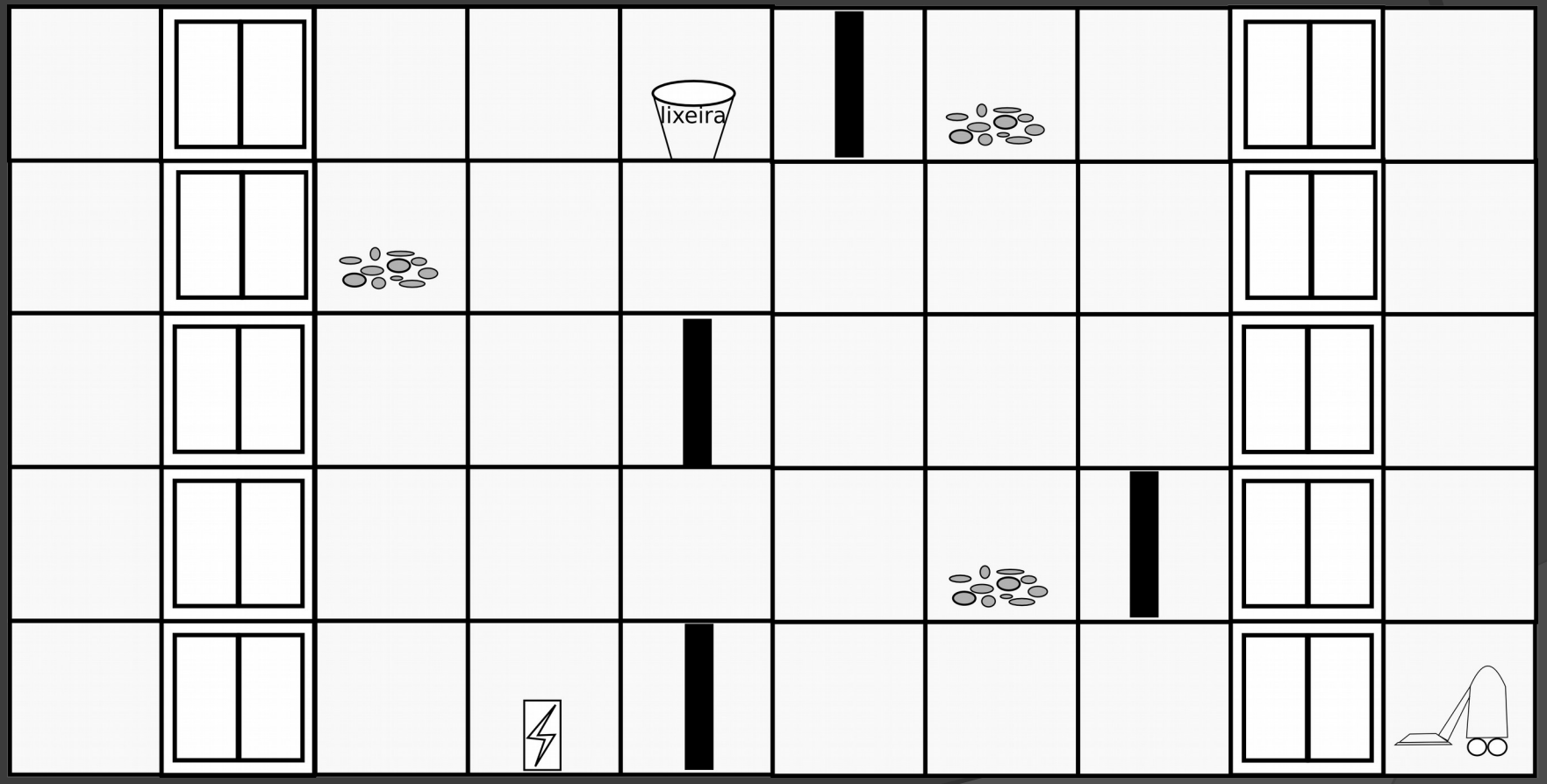
- Durante o caminho, o aadp deve passar por todas as sujeiras do cenário.
- Pode ser **uma** ou **mais!!!!**
- Ou seja, para que o caminho seja válido, é necessário que **todas** as sujeiras estejam em alguma posição do caminho.



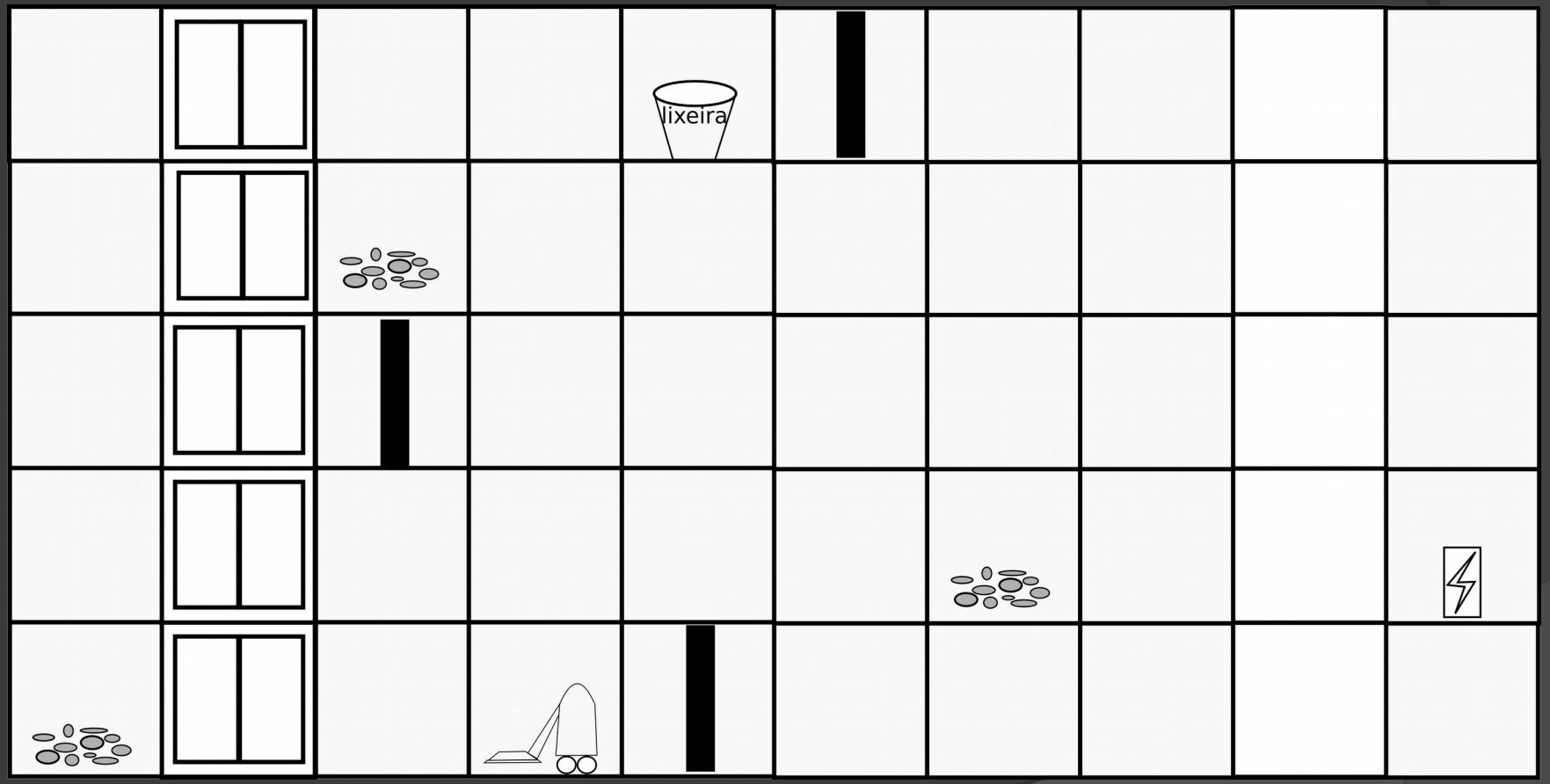
Lixeira

- Todas as vezes que o aadp passar por **duas sujeiras**, ele deve passar pela lixeira para descarregar toda a sujeira adquirida no caminho!
- Depois ele poderá seguir até o *dock station* e completar o caminho!

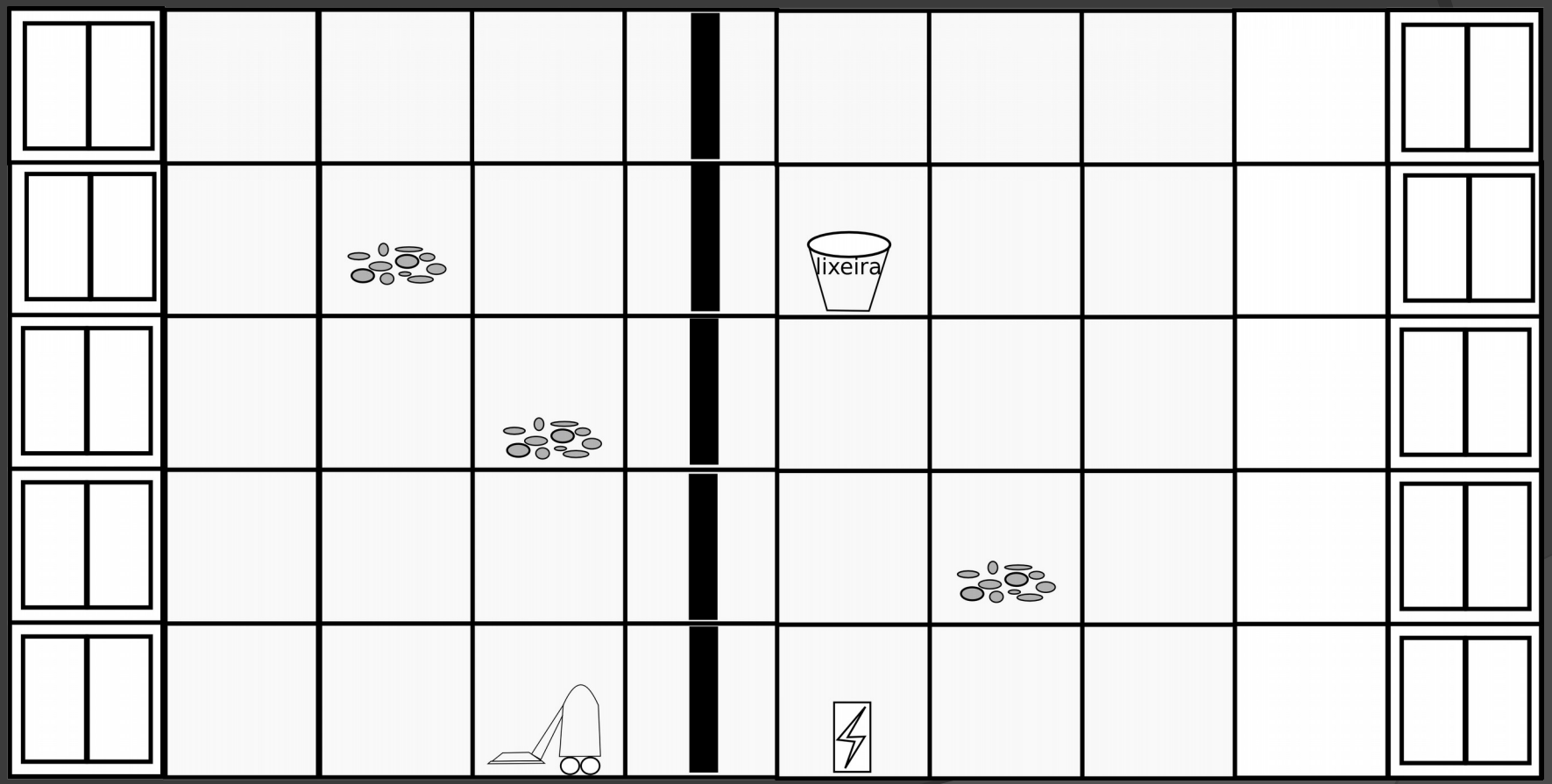
Ambiente Completo 1



Ambiente Completo 2



Ambiente Completo 3



Objetivo 1

- Implementar em Prolog o ambiente de forma adequada (em regras e fatos), considerando os objetos:
  - aadp, lixeira, sujeiras, *dock station*
  - elevadores e paredes

- Deve se considerar que qualquer objeto pode estar em qualquer posição da tabela (programável pelas regras ou consulta)
- Valor: até 2/10 dos pontos!

Objetivo 2

- Dado um estado inicial (aadp) e um estado final (*dock station*), obter o caminho entre os dois, de forma que:
  - O estado inicial seja escolhido pelo usuário na chamada (?-)
  - O caminho deve conter todas as sujeiras e, a cada duas sujeiras, a lixeira.
  - Ser capaz de rodar em diferentes cenários.
- Valor: até 8/10 dos pontos!

Grupos

- Grupos de até 3 pessoas
- Notas individuais de apresentação.

Documentação do projeto

- Tudo que foi desenvolvido (incluindo os desafios) deve ser explicado em um documento .pdf (somente digital) entregue no AVA até dia 17 de maio. Outros formatos de arquivos não serão aceitos.
- Esta documentação deve conter a base conhecimento utilizada pelo Prolog.
- Também deve conter exemplos de utilização do conjunto de regras, mostrando todas as funcionalidades implementadas.
- Deve ser completo e didática, de forma que um aluno possa entender.

Apresentação do projeto

- Cada grupo deverá apresentar o seu projeto de maneira didática em 10 minutos no dia 17 de maio.
- Deverá ser feito o upload da apresentação junto com o pdf no AVA!
- A apresentação deve explicar como os fatos e as regras foram gerados, bem como a lógica que faz com que o Prolog consiga chegar no objetivo
- Deve conter exemplo(s) de uso
- Todos os membros do grupo devem apresentar e ganharão notas individuais nessa etapa
- Serão feitas perguntas (até 5 minutos.
- Excesso de tempo penaliza a nota.

Calculo da nota final

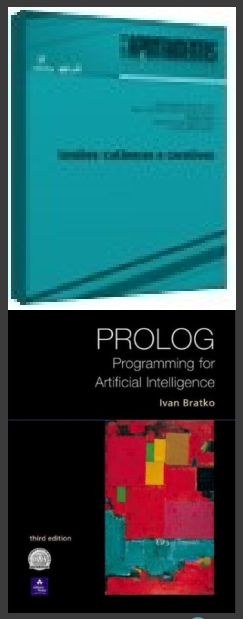
- Nota Trabalho (NT) → 0-10
- Nota Apresentação (NA) → 0-1

NF = NT * NA

Não faça!

- Se quiser perder nota:
  - Demore para introduzir o problema em sua apresentação, explicando tudo que já expliquei. Quero ver o que vocês fizeram, não o enunciado do trabalho.
  - Copie material e não cite a fonte, especialmente se forem as minhas figuras e slides.
  - Faça várias regras de sucessor para cada dupla de estados e para todos os estados separadamente.
  - Não use regras gerais que possam ser facilmente aplicadas a qualquer estado.
  - Não mostre que sua solução é capaz de resolver cenários interessantes/desafiadores.

Bibliografia



NICOLETTI, M. C. A Cartilha Prolog. EDUFSCAR. 2005. ISBN 8576000113

Bratko – Prolog: Programming for Artificial Intelligence 2001

- Material de apoio no PVAnet
- Prolog para download: http://www.swi-prolog.org/ (ou repositorio Ubuntu)