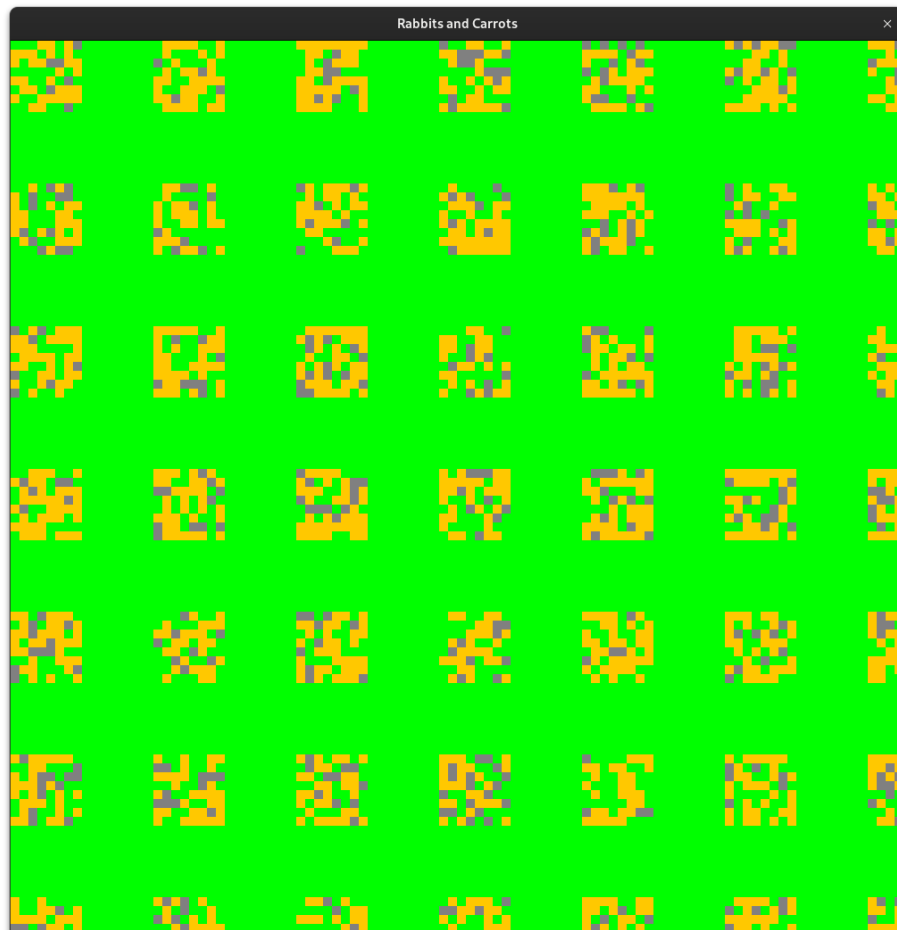


## Relatório de Programação II

David Marinho  
54560

Axel Amoroso Carapinha  
55248



## Procedimentos

1. Na execução do programa podem ser passados 3 argumentos pela linha de comandos:
  - $i \rightarrow$  O comprimento do prado.
  - $j \rightarrow$  A largura do prado.
  - $starveTime \rightarrow$  Número de timesteps a que os coelhos sobrevivem sem comer qualquer cenoura. Por defeito, se não passados quaisquer argumentos, temos  $i = 100$ ,  $j = 100$  e  $starveTime = 5$ .
2. Um prado de comprimento  $i \times j$  é preenchido com objetos do tipo 'Grass' e são gerados aleatoriamente outros objetos, objetos de tipo 'Rabbit' e 'Carrot'.
3. A cada timestep é gerado um novo prado respeitando as regras a que a geração anterior esteve sujeita.

## Design Patterns

Recorremos aos três pilares da Programação Orientada a Objetos, encapsulamento herança e polimorfismo, e criamos quatro classes para representar entidades específicas:

- LifeBeing  $\rightarrow$  Classe mãe e abstrata das classes 'Rabbit', 'Carrot' e 'Grass'.
- Rabbit  $\rightarrow$  Classe filha que herda 'LifeBeing'. Tem como função organizar todos os dados a respeito dos coelhos.
- Carrot  $\rightarrow$  Classe filha que herda 'LifeBeing'. Organiza toda a estrutura de dados a respeito das cenouras.
- Grass  $\rightarrow$  Classe filha que herda 'LifeBeing' e organiza toda a estrutura de dados a respeito das ervas.
- Grassland  $\rightarrow$  Classe responsável por organizar todos os dados a respeito do campo, maioritariamente a posição dos objetos no campo e o tempo de longevidade dos mesmos.
- Simulation  $\rightarrow$  Classe principal do programa. Responsável pela renderização e execução do programa.

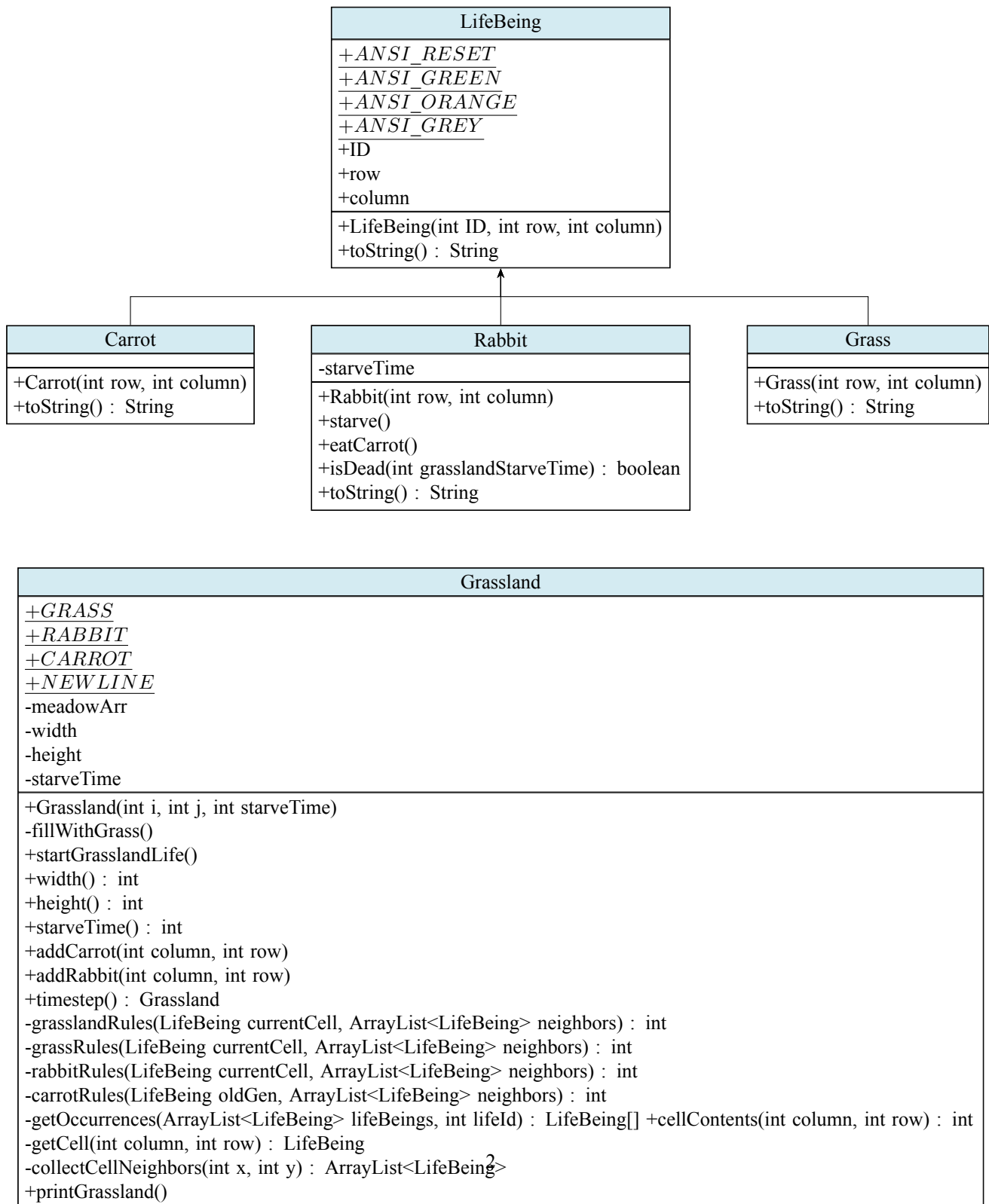


Figure 1: Diagrama de classes