PPA: Coursework 3 (Life)
Authors: Rayan Popat (K21056367) & James Coward (K22004743)

CELLS
BASIC UNITS OF LIFE

—

Introduction

The purpose of this coursework is to create a simulation that emulates the human gut biome. We wanted the simulation to demonstrate the complex interaction between distinct cells prominently found in gut biomes such as Mycoplasma, Staphylococcus Aureus (S. Aureus), Lactobacillus and Cancer.

Each cell is modelled in a class that inherits from a common superclass named Cell. This was done so as to reduce code duplication, improve maintainability and increase code extendability.

The simulation is, by default, a 100x80 field of locations. Where, at runtime, the *populate()* method in the Simulator class randomly populates the field with cells, according to the probabilities specified in the static CELL_TYPE_PROBABILITIES Hashmap.

Once populated, the cells interact with one another based on a set of rules defined in each of their respective subclasses. These rulesets are described in more detail below.

Description of Life Forms

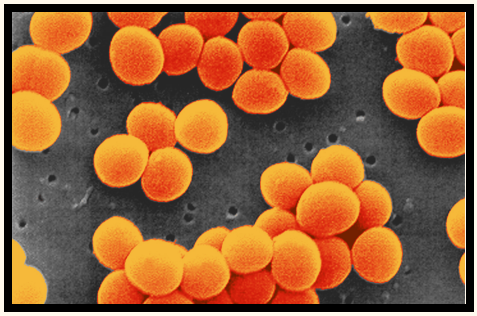
Mycoplasma

Mycoplasma is a bacterium that causes infections in different areas of your body, such as the gut. They also, interestingly, lack a cell wall around their cell membranes, meaning they are more susceptible to incoming chemicals produced by other cells. In our simulation, it is represented by a green cell.

Rule Set

- If the cell has fewer than two live Mycoplasma neighbours it will die (isolation).
- If the cell has two or three live Mycoplasma neighbours it will live on to the next generation.
- If the cell has more than three Mycoplasma neighbours it will die (overcrowding).
- If the cell has more than three S. Aureus neighbours it will die (toxic environment).

Staphylococcus Aureus



Staphylococcus Aureus is a spherically shaped bacterium that can produce toxins that cause a range of illnesses. One interesting toxin that we chose to implement in our simulation is the Staphylococcal Enterotoxin B (SEB). SEB is a superantigen that is harmful to other cells, which is why each cell in the environment (apart from cancer) will die if it has more than three S.Aureus neighbours.

Rule Set

- If the cell has fewer than one live S. Aureus neighbour it will die (isolation).
- If the cell has two or three live S. Aureus neighbours it will live on to the next generation.
- If the cell has more than five S. Aureus neighbours it will die (overcrowding).
- If the cell has more than six Lactobacillus neighbours it will die (pH Imbalance).

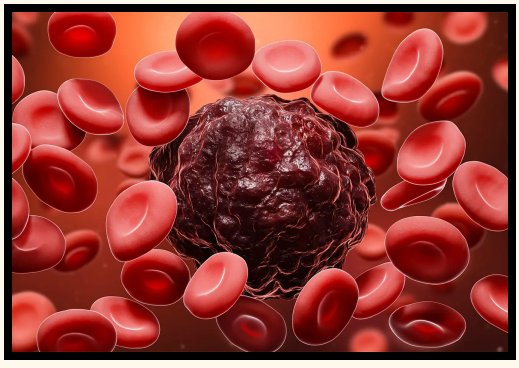
Lactobacillus Acidophilus

Lactobacillus is a genus of rod-shaped bacteria and is a type of probiotic found primarily. in the human gut. L. acidophilus can help break down food, absorb nutrients, and fight off "bad" organisms that might cause diseases. The bacteria interestingly also produces Lactic Acid, which lowers the pH of the surrounding environment, and kills off harmful organisms such as S. Aureus.

Rule Set

- If the cell has more than five Lactobacillus neighbours it will die (overcrowding).
- If the cell has more than three S. Aureus neighbours it will die (toxic environment).

Cancer



Cancer is a condition where cells in a specific part of the body grow and reproduce uncontrollably. The cancerous cells can invade and destroy surrounding healthy tissue, including organs. Cancer can be benign (grow and spread slowly) or malignant (grow and spread rapidly). **In our simulation, cancer spawns benign, and can only infect other cells if it is malignant.**

Rule Set

- A dead cell has a small chance to become cancerous (benign).
- If the cell is benign, it has a small chance to become malignant.
- If the cell is malignant, it infects other cells around it to become benign cancer.

Tasks Completed

Base Tasks

Mycoplasma Ruleset

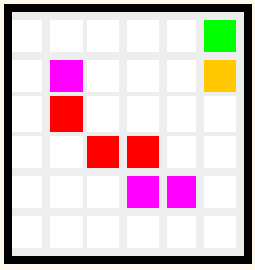
To implement the Mycoplasma rules, we used conditional statements to check the number of Mycoplasma neighbours of each cell while iterating through the cells in the simulation. CellTypes were kept track of using an enum.

```
protected HashMap<CellType, Integer> getNeighbourTypes() {
  HashMap<CellType, Integer> neighbourTypes = new HashMap<>();
  List<Cell> neighbours = getNeighbours();
  for (Cell neighbour : neighbours) {
    neighbourTypes.merge(neighbour.getType(), 1, Integer::sum);
  }
  return neighbourTypes;
}
```

The number of types of each cell in a cell's list of neighbours was kept track of using a HashMap returned by the function *getNeighbourTypes()*.

Colour Changing

For implementing colour change, we decided to use colour to visually represent the different states of the cancer cells, with benign cells appearing pink and malignant cells appearing red. As the simulation progresses, the state of the cancer cell changes, along with the colour.



Different Behaviour over Time

As time progresses, a cancer cell changes states from benign to malignant, changing from not spreading (benign) to actively infecting and transforming other cells (malignant). If a cell is malignant, it acts differently to if it is benign.

Challenge Tasks

Non-deterministic Cells

The non-deterministic cell in our simulation was Cancer, which had 3 stages: Dead, Benign and Malignant, with the changes between these states all having a probability to occur. Changes only occur from left to right, i.e. Dead to Benign, and Benign to Malignant.

```
if (getAge() % 5 == 0 && getAge() > 25) {
    if (Math.random() <= cancerChance) {
        updateState(CellType.CANCER);
        return;
    }
}
```

Firstly, any given DeadCell which has been 'alive' (present in the simulation) for more than 25 generations will have a (very small) chance to turn into a benign cancer cell every 5th generation after (i.e. its 30th generation alive, its 35th generation, etc.).

```
if (!malignant) {
    if (Math.random() <= malignantChance) {
        setColor(Color.RED);
        this.malignant = true;
    }
}
```

Once a DeadCell becomes a benign cancer cell, that benign cancer cell then has a probability to turn into a **malignant** cancer cell.

Disease

Once a cancer cell becomes malignant, all neighbouring cells will have a chance to turn into malignant cancer cells. This is done independently of the cancer cell itself - each neighbouring cell checks if it has a malignant cancer cell adjacent, and if there is, the cell itself has a chance to transform into a benign cancer cell (i.e. become 'diseased'). This emulates the spread of a cancer, as living cells become infected and transform into benign cancer cells, which then transform into malignant ones, causing other neighbouring cells to transform.

```
if (neighbourTypeCount.get(CellType.CANCER) != null && checkMalignant()) {
    if (Math.random() <= spreadChance) {
        updateState(CellType.CANCER);
        return;
    }
}
```

Symbiotic / Parasitic Cell Relationships

The cancer cell effectively acts as a parasite on all living cells in the simulation, with the ability to 'infect' any given living cell that has a neighbouring malignant cancer cell. In this sense, a parasitic relationship is formed, as the cancer actively benefits from the spread of living cells, as it gives the cancer itself more opportunity to spread, whereas living cells are actively harmed.