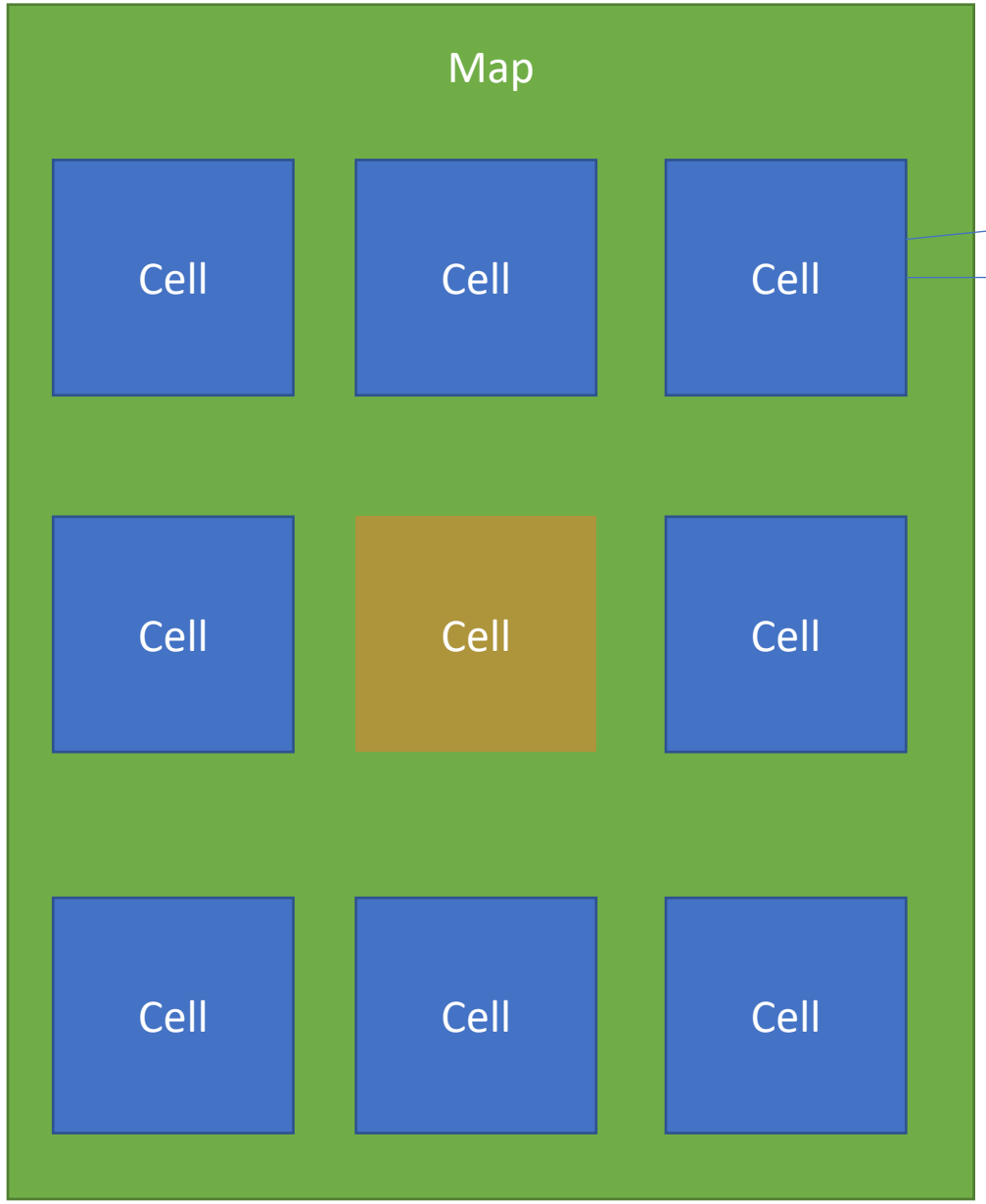
Gruppe 17

INF200

Planlegging og struktur

- Planlegging
- Design av tester
- Testbasert programmering
- Utforsking av egne ideer
- Implementere flere tester
- Utarbeide dokumentasjon

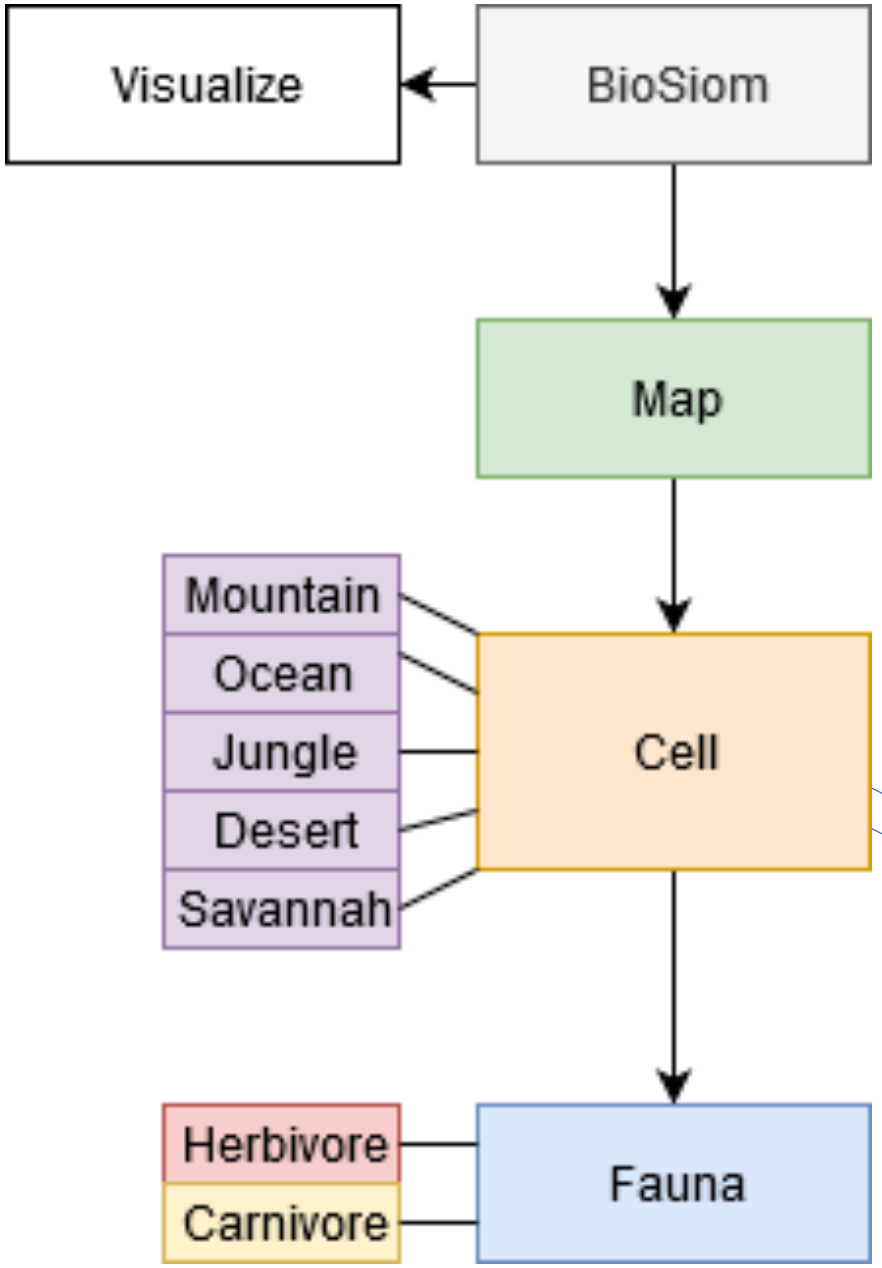


List of herbivore objects
List of carnivore objects

- The map is an object containing cell_map.
- Cell_map is a matrix containing cell-objects.
- Each cell-object has an attribute called population_herbivores and population_carnivores.
- In these attributes we can add creatures.

- Visualization is a separate object, which takes information from the map-class to create a visual understanding of what's going on on the Island.

- BioSim class contains necessary functions and attributes to run the project.



- BioSim class contains necessary functions and attributes to run the project.
- The map is an object containing cell_map.
- Cell_map is a matrix containing cell-objects.
- Each cell-object has an attribute called population_herbivores and population_carnivores.
- In these attributes we can add creatures.

- Visualization is a separate object, which takes information from the map-class to create a visual understanding of what's going on on the Island.

List of herbivore objects
List of carnivore objects

Troverdighet

- Enhetstester
  - Forskjellige lister
  - Parameterisering
  - Implementering
  - If/else

test_biosim_interface....	97% lines covered
test_cell.py	100% lines covered
test_fauna.py	82% lines covered
test_map.py	88% lines covered

Noen kodesnutter som vi ønsker å vise frem:

- Lagre til csv hvert år:

```python
def save_mid_simulation_result(self, herbivores, carnivores, total):
    """ Saves the mid simulation results to a CSV-file each year. """
    with open('save mid simulation result', 'a', newline='') as file:
        writer = csv.writer(file)
        writer.writerow([self._year, herbivores, carnivores, total])
```

```python
    #mock.patch("biosim.map.choice", return_value=1, autospec=True)
    @mock.patch("biosim.map.choice", return_value=1, autospec=True)
    def test_select_index_to_move(self, mock_choice, map=map):
        probabilities_index_test = [0, 0.25, 0.75, 0.25]
        index = map.select_index_to_move(probabilities_index_test)
        assert index == 1
        mock_choice.assert_called_once_with(1)
```

Problem med å visualisere med andre frekvenser enn en, her var en liten hotfix.

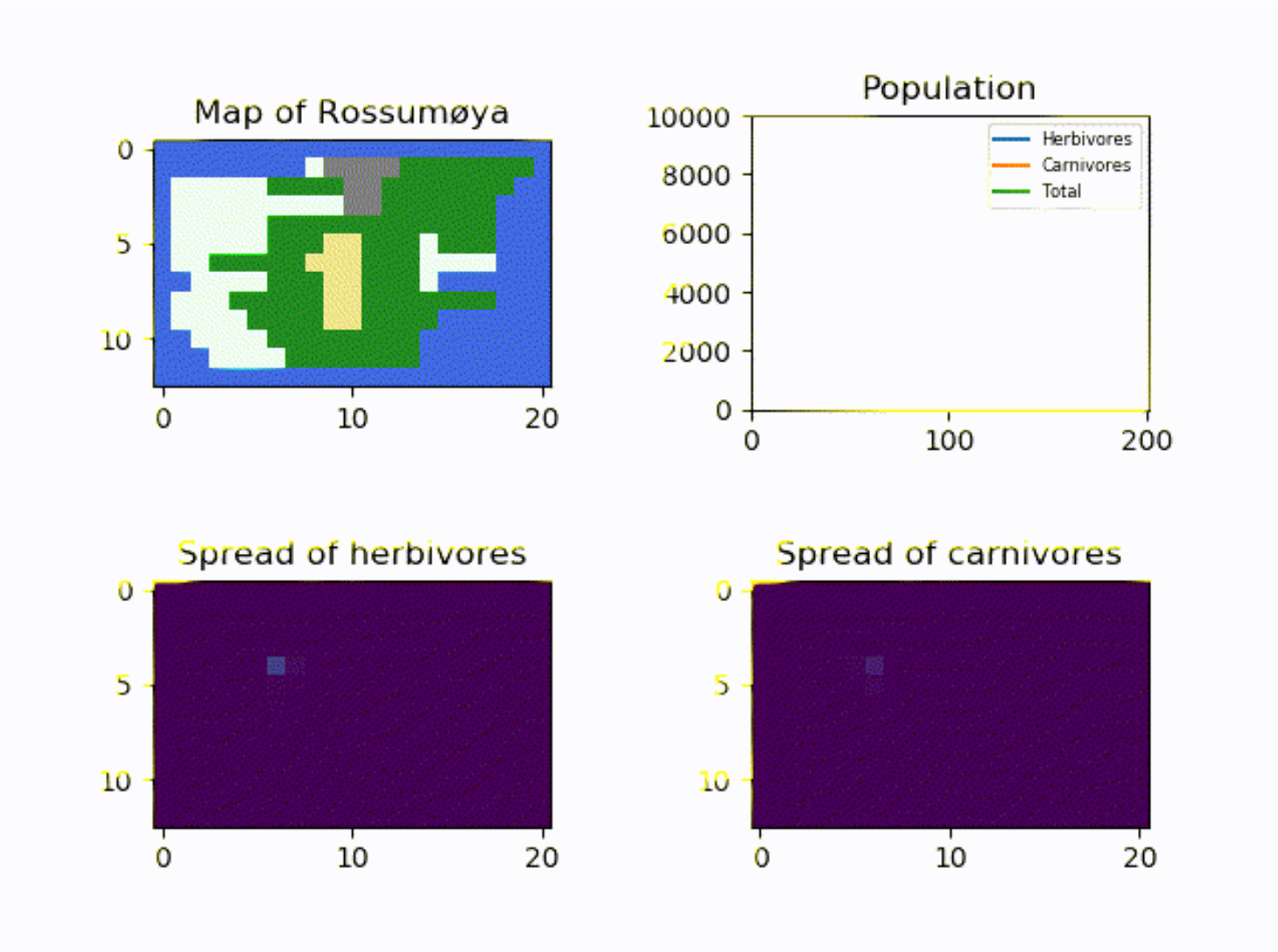
```
# Converts nan values
added_herbivores = 0
added_carnivores = 0
for i in range(current_year):
    if math.isnan(herbivore_data[i]):
    # if herbivore_data[i] is NaN:
        if added_herbivores == 0:
            herbivore_step = added_herbivores / self.frequency
            carnivore_step = added_carnivores / self.frequency
        if i != 0:
            herbivore_data[i] = herbivore_data[i-1] + herbivore_step
            carnivore_data[i] = carnivore_data[i - 1] + carnivore_step
            total_data[i] = herbivore_data[i] + carnivore_data[i]
```

Kunne ha forenklet med comprehensions.

```
221
222     def add_age(self):
223         """
224         Increases the age for all the creatures in the population list.
225         :return:
226         """
227         for creature in self.population_herbivores:
228             creature.age += 1
229         for creature in self.population_carnivores:
230             creature.age += 1
```

```
221
222     def add_age(self):
223         """
224         Increases the age for all the creatures in the population list.
225         :return:
226         """
227         [herbivore.age+1 for herbivore in self.population_herbivores]
228         [carnivore.age + 1 for carnivore in self.population_carnivores]
229
```

Simulering


Map of Rossumøya
0
5
10
0
10
20
Population
10000
8000
6000
4000
2000
0
0
100
200
Herbivores
Carnivores
Total
Spread of herbivores
0
5
10
0
10
20
Spread of carnivores
0
5
10
0
10
20