ID2209
Assignment 3- Coordination and Utility

Group 26
Edvin Walleborn
Patrik Zhong

Species

Queens

The queen is what constitutes the absolute majority of task 1 and is constructed with only a few methods. The queen has access to a reflex which acts depending on the inform received from another queen or the starting inform. The queen can either be told to locate her position in the grid, and she does so by examining the predecessor's index relative to the current queen and deciding whether it's safe or not to position somewhere, or she is told to reposition by her successor. When a queen receives a repositioning message from the successor, she simply does the same act as when she originally located her valid position, except that this time she adds her previous position to the illegal positions. When a queen has positioned herself, she messages the next queen in line to find her position. This queen tries to find her position, and if none is available she tells the previous one to reposition herself. This repositioning message can cascade up the queue back to the first queen.

Guests

This species was the main species. The guest represented each visitor at the festival and what type of preferences it had. It has the ability to communicate with all stages using fipa and request their utility values. The main way it gets the values from stages is asking them on a timer that repeats itself. After having asked, it simply calculates its total utility gained from each stage and walks towards it.

Stages

The stage is a specie that is spawned at the beginning of the program. There are 4 in total that are spawned at set coordinates. They each generate a total of 6 values, such as lighting, music, band and moshpits. These values are changed at set interval. The guests interact with the stage by asking them and the stages tell them their values.

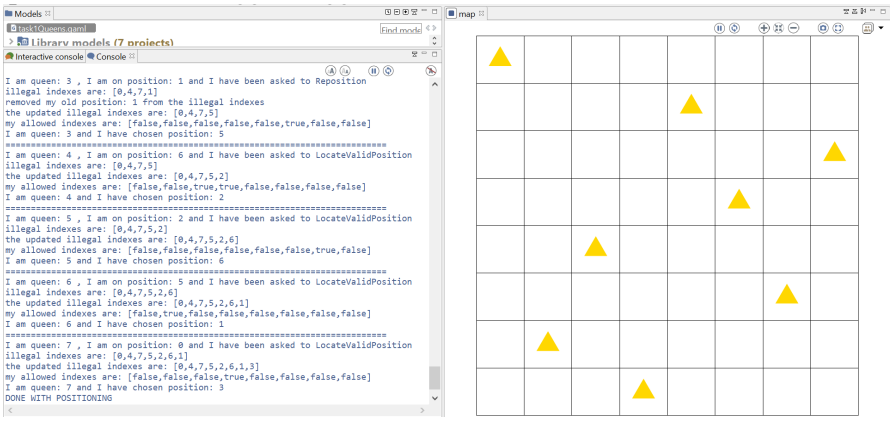
Task 1

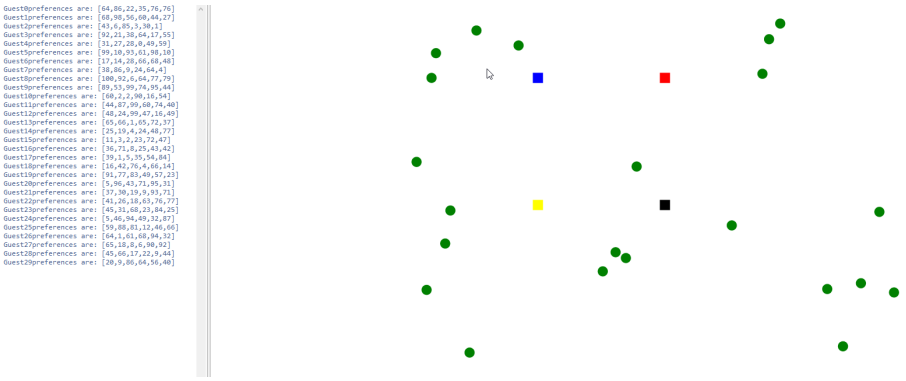
In this task we are to implement the 'N Queens problem', in which N amount of queens must be positioned on a NxN grid, in such a way that they do not 'check' each other. This is implemented via a grid that has basically no functions and only acts as a method for positioning the queens, and the queens themselves. The queens are capable of communicating with each other and thus position themselves depending on their positions according to the method previously described.

Task 2

In this task we implemented a very simple utility-based protocol. The stages create their own values and the guests ask for them. Depending on which stage that has the highest utility, they will then go to that coordinate. After a set amount of time, the stages reset their values and generate new ones. These are then once again processed by each guest. This occurs every set time frame.

Results





For task 1, we can see that the queens are properly placed and we can see in the console that the algorithm is implemented correctly.

For task2, we can see that the coloured squares that represent the stages are set in the middle, and that the guests move towards them depending on their values which are shown in the console.

Discussion/Conclusion

This assignment was very interesting when it came down to implementing an algorithm through FIPA. It was interesting to see how little info that had to be passed between the queens for them to figure out their placement. For task 2, we utilized previous learnings and reinforced how the FIPA protocol worked.