# Presentation of the data structure

The data structure is divided into two big blocks: data (gegevens) and solution (oplossing). The data exists of all the input data witch was read from the .csv file. The solution is a data structure which is continuously changed by our searching algorithm. Below you can find a detailed overview of our data classes.

## Data (gegevens)

Reservatie:

|  |  |
| --- | --- |
| id | The id of the reservation |
| zoneId | The id of the zone of the reservation |
| dag | The day of the reservation |
| startTijd | The starting time in minutes from the beginning of the day |
| duur | The length of the reservation in minutes |
| voertuigID | This is a list of all possible vehicles |
| penalty1 | Penalty which is added to the total cost when the reservation is not assigned to a car |
| penalty2 | Penalty which is added to the total cost when the reservation is assigned to a car that is not located in the same zone. |

Voertuig:

|  |  |
| --- | --- |
| id | The id of the vehicle |

Zone:

|  |  |
| --- | --- |
| id | The id of the zone |
| aanliggendId | This is a list of all neighbor zones |

## Solution (oplossing)

AReservatie:

|  |  |
| --- | --- |
| resId | The id of the reservation |
| voertuigId | This is the id of the vehicle which got assigned to this reservation. When the value is NULL, it means that there is no vehicle assigned to the reservation. |
| penalty1 | This is a copy of penalty1 from the data. We make a copy of this value for faster computation in the searching algorithm. |
| penalty2 | This is a copy of penalty1 from the data. We make a copy of this value for faster computation in the searching algorithm. |
| AVoertuig | This is a pointer to the vehicle object. |

AVoertuig:

|  |  |
| --- | --- |
| voertuigId | The id of the vehicle. |
| zoneId | The id of the zone which the vehicle got assigned to. |
| reservaties | This is a list of all reservation which this vehicle got assigned to. |

# Algorithm

We made two different algorithms. They both use the same data structure.

## Algorithm 1

First the algorithm generates a initial solution. The initial solution is feasible but is has a very high cost. Al cars will be placed in zone 0 and there are no cars linked to a reservation.

Then the algorithm picks a random car and place it in a random zone. The car is more likely not yet linked to a reservation. The car can only be moved to a neighbor zone.

Then we check if the car was already linked to a reservation. If the car was already linked to a reservation we first have to check if the link between the reservation and de car is still possible. When the link is not possible anymore, the car must be unlinked from the reservation.

Now we can try to attach the car to a different reservation. First the algorithm searches a reservation which is located in the same zone. When there is no reservation possible in the same zone, the algorithm tries to find a reservation in a neighbor zone.

Each iteration, a cost is calculated. When the new cost is lower than the previous cost, the solution is saved.