Design Documentation

List of semaphores

```
    Queue<Customer> customersWaitingForANumber; // Customers waiting in line at Information Desk to get a number.
    Queue<Customer> customersInWaitingArea; // Customers in the waiting area until number is called.
    Queue<Customer> customersWaitingForAnAgent; // Customers in line for an agent.
    Semaphore customersWaitingForANumberMutex = 1; // Controls access to the customersWaitingForANumber queue.
    Semaphore customersInWaitingAreaMutex = 1; // Controls access to the customersInWaitingArea queue.
    Semaphore customersWaitingForAnAgentMutex = 1; // Controls access to the customersWaitingForAnAgent queue.
    Semaphore informationDesk = 1; // Shows if the information desk is available for the customer to approach
    Semaphore customersReadyForANumber =  0; // Shows if a customer is in line ready for a number
    Semaphore customerReadyForAnAgent = 0; // Shows if a customer is in the waiting room ready for an agent
    Semaphore agent = 2; // Shows how many agents are available for the customer to approach
    Semaphore agentLine = 4; // Shows how many spots are open in the agent line
    Semaphore customerReadyForService = 0; // Show how many customers are in the agent line
```

Pseudocode

```
void main(){
  create DMV();
  create InformationDesk();
  create Announcer();
  create Agent(2);
  create Customer(Capacity);
  while(customersJoined < capacity)
    run();
  print("Done");
  exit;
}
```

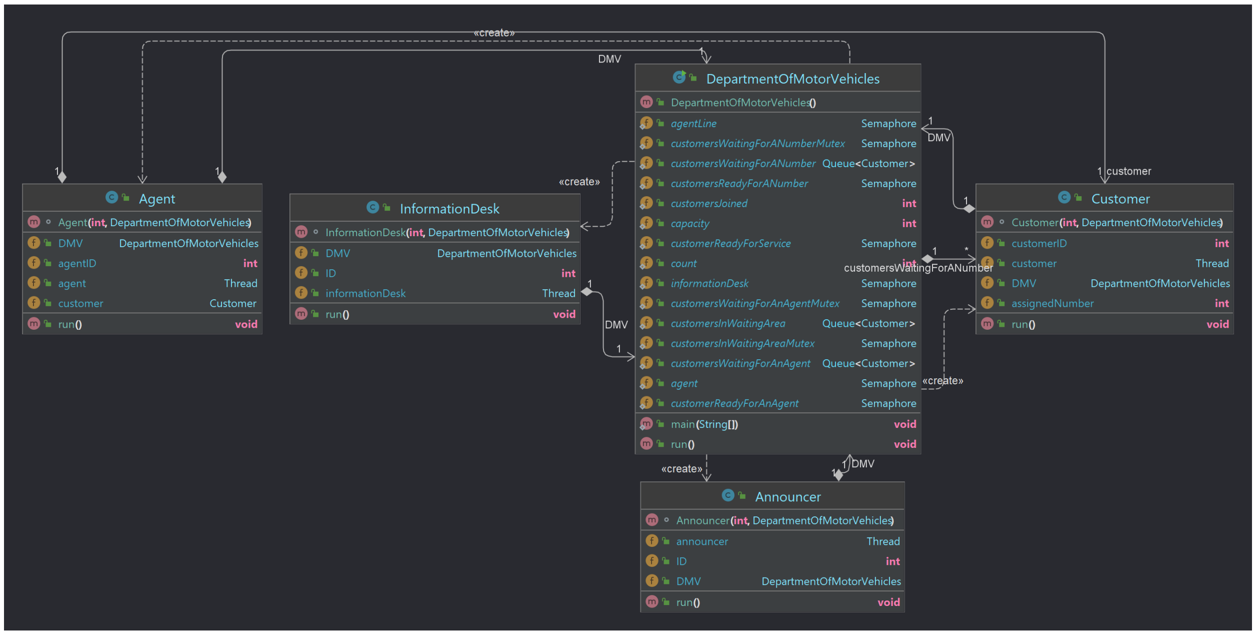
```
void Customer(){
  wait(customersWaitingForANumberMutex);
  customersWaitingForANumber.add(customer);
  signal(customersWaitingForANumberMutex);
  wait(informationDesk);
  print("Customer created, enters DMV.");
  signal(customersReadyForANumber);
  signal(informationDesk);
  singal(customerReadyForAnAgent);
}
```

```
void InformationDesk(){
  wait(customersReadyForANumber);
  wait(customersInWaitingAreaMutex);
  wait(customersWaitingForANumberMutex);
  print("Customer gets number, enters waiting room");
  customersInWaitingArea.add(customer);
  signal(customersWaitingForANumberMutex);
  signal(customersInWaitingAreaMutex);
  signal(customersReadyForANumber);
}
```

```
void Announcer(){
  wait(agentLine);
  wait(customerReadyForAnAgent);
  wait(customersInWaitingAreaMutex);
  wait(customersWaitingForAnAgentMutex);
  print("Announcer calls number");
  print("Customer moves to agent line" );
  customersWaitingForAnAgent.add(customer);
  signal(customersWaitingForAnAgentMutex);
  signal(customersInWaitingAreaMutex);
  wait(agent);
  signal(agentLine);
  signal(customerReadyForService);
  signal(agent);
}
```

```
void Agent(){
  wait(customerReadyForService);
  wait(customersWaitingForAnAgentMutex);
  customer = customersWaitingForAnAgent.remove();
  print("Agent is serving customer");
  signal(customersWaitingForAnAgentMutex);
  print("Agent asks customer to take photo and eye exam");
  print("Customer completes photo and eye exam for agent");
  print("Agent gives license to customer");
  print("Customer gets license and departs");
  customer.join();
  print("Customer was joined");
}
```

UML



«create»
DMV
1
DepartmentOfMotorVehicles
DepartmentOfMotorVehicles()
agentLine Semaphore
customersWaitingForANumberMutex Semaphore
customersWaitingForANumber Queue<Customer>
customersReadyForANumber Semaphore
customersJoined int
capacity int
customerReadyForService Semaphore
count int
informationDesk Semaphore
customersWaitingForAnAgentMutex Semaphore
customersInWaitingArea Queue<Customer>
customersInWaitingAreaMutex Semaphore
customersWaitingForAnAgent Queue<Customer>
agent Semaphore
customerReadyForAnAgent Semaphore
main(String[]) void
run() void
Agent
Agent(int, DepartmentOfMotorVehicles)
DMV DepartmentOfMotorVehicles
agentID int
agent Thread
customer Customer
run() void
«create»
InformationDesk
InformationDesk(int, DepartmentOfMotorVehicles)
DMV DepartmentOfMotorVehicles
ID int
informationDesk Thread
run() void
DMV
1
customersWaitingForANumber
*
customer
Customer
Customer(int, DepartmentOfMotorVehicles)
customerID int
customer Thread
DMV DepartmentOfMotorVehicles
assignedNumber int
run() void
«create»
«create»
DMV
Announcer
Announcer(int, DepartmentOfMotorVehicles)
announcer Thread
ID int
DMV DepartmentOfMotorVehicles
run() void

End of document.