Invariantes

1. Todos os registos de um parque são relativos a viaturas registadas no sistema.

**Contexto GesPark**

**INV** registosViaturasExistemMesmo :: registos->forAll(r | identificadores -> one(i|i.registos.includes(r)))

1. Construa o **Diagrama de Sequência** para a operação da classe GesPark que, dado um identificador de uma viatura e um intervalo de datas, determina o montante a pagar nesse intervalo de datas.

A operação deverá ter como pré-condição a existência do identificador dado. Acrescente-a ao Diagrama de classes.

**Contexto GesPark** montanteAPagar(id : String, início : LocalDateTime, fim : LocalDateTime): float

**PRE**: identificadores->exists(i|i.id=id

1. Construa o **Diagrama de Sequência** para a operação da classe GesPark que, dado um identificador, determina o total de parques visitados pela viatura a que este está associado.

A operação deverá ter como pré-condição que o identificador existe e como pós-condição que o resultado é não negativo. Acrescente-as ao Diagrama de classes.

**Contexto GesPark** totalParquesVisitados(id: String): int

**PRE**: identificadores->exists(i|i.id=id)

**Pos**: result >=0

1. Considere agora que o sub-sistema possui informação sobre os utentes que se registaram nos parques. Acrescente essa informação à arquitectura e construa o **Diagrama de Sequência** da operação que permitirá calcular, para todas as viaturas de um utente, identificado pelo seu NIF, qual o montante global a pagar num determinado intervalo de datas.

A operação deverá ter como pré-condição que o utente está registado no sub-sistema. Acrescente-a ao Diagrama de Classes.

**Contexto GesPark** montanteAPagar(nif: String, início : LocalDateTime, fim : LocalDateTime): float

**PRE**: identificadores->exists(i|i.viatura.utente.nif = nif