Relatório de Implementação

Projeto da cadeira de Programação II – dronyDeliv

- Grupo 82
- José Miguel Vilhena de Almeida – nº 55373
- Miguel Lourenço Lages – nº 54975

Creditação:

José:

- organize.py
- timeTools.py
- Combo.py
- DronyD.py
- AutoDronyD.py
- FileNames.py

Miguel:

- reafFiles.py
- writeFiles.py
- Drone.py
- Parcel.py
- Header.py

E ambos tratamos do dronyD.py e fomos sempre conversando por videochamada à medida que avançávamos na criação e resolução do programa, explicando tudo o que fizemos um ao outro de forma a que cada um de nós não só entendesse o que o outro fez, mas também o código do programa em si.

Funções implementadas:

- dronyD.py – permite executar o programa chamando os dois ficheiros (o ficheiro de drones e o das encomendas) na linha de comandos (consola), seguindo a seguinte sintaxe: python dronyD.py <ficheiro de drone> <ficheiro de listagem de encomendas>

- autoDronyD.py – uma função adicional que permite executar o programa sem ser necessário chamar os dois ficheiros na linha de comandos (consola). Apenas é necessário que os dois ficheiros (o ficheiro de drones e o das encomendas) se encontrem no mesmo diretório que a função autoDrony.py.

- readFiles.py – recebe ficheiros tanto como os drones, como com as encomendas e coloca as informações necessárias nos respetivos objetos. Ou seja, recebe a informação dos ficheiros com os drones e coloca num objeto Drone; recebe a informação dos ficheiros das encomendas e coloca num objeto Parcel; e recebe a informação do header e coloco num objeto Header.

- Drone.py – objeto que representa um drone, onde se encontram todos os atributos recebidos no input file que foi submetido, anteriormente, no readFiles.py. Contém também o método de acesso (getter) e o método de mudança de estado (setter).

- Parcel.py – objeto que representa uma encomenda, onde se encontram todos os atributos recebidos no input file que foi submetido, anteriormente, no readFiles.py. Contém também o método de acesso (getter) e o método de mudança de estado (setter).

- Header.py - objeto que representa uma encomenda, onde se encontram todos os atributos recebidos no input file que foi submetido, anteriormente, no readFiles.py. Contém também o método de acesso (getter).

- Combo.py – classe implementada para conectar os drones com as respetivas parcels, seguindo os vários critérios necessários. Define também o estado das encomendas.

- organize.py – define qual o melhor drone para a encomenda em questão, mais uma vez, seguindo os vários critérios necessários.

- writeFiles.py – recebe dois ficheiros, o primeiro consiste numa lista das encomendas com os respetivos drones atribuídos no organize.py e conectados no Combo.py, e outra com os drones atualizados, tendo em conta as viagens que os mesmo foram sujeitos.

- timeTools.py – tem como função obter a data e hora de um determinado pedido ou encomenda e, dependendo da situação, fazer trocas de formato ou adicionar/subtrair tempo a uma determinada hora.

- FileNames.py – uma classe que permite obter o nome dos ficheiros, tanto de drones, como também de encomendas.

Funções por implementar:

- Da nossa perspetiva não ficou nenhuma função por implementar.

Erros conhecidos:

- Não foram detetados nenhuns erros durante realização do projeto, nem durante a execução do programa.