Relatório de Implementação

Programação 2 (LTI)

1. Grupo 40
2. Guilherme Miguel Barros de Almeida; nº 52052
   Rui Afonso Dias Escudeiro Pereira; nº 51623
3. Trabalho feito por cada membro (%):
   Guilherme: 60%
   Rui: 40%

O trabalho de grupo foi feito por ambos os membros na percentagem apresentada acima.

Começámos por criar as classes "Client" e "Expert", os seus respetivos getters e setters e o método __str__. De seguida, implementámos o método "getNewTime" em ambas as classes assim como os seus métodos auxiliares para, mais tarde, os podermos utilizar nas várias ocasiões em que me viria ser necessário.

Depois disto, começámos a desenvolver as classes "clientCollection" e "expertCollection" que têm como objetivo ler os ficheiros input dados e criar tantos objetos respetivos (Client/Expert) quantos aqueles presentes no ficheiro. Ambas as classes funcionam como subclasse da classe built-in "UserList".

Dando as classes por terminadas passámos então para a classe Schedule, classe que tem como objetivo tanto a atribuição do par client-expert e a atualização dos experts com a sua nova disponibilidade e dinheiro acumulado, como também a escrita dos novos ficheiros. Ao começar a fazer a atribuição das matches chegámos à conclusão que nos seria útil implementar um método __lt__ na classe Expert para poder verificar qual o expert mais adequado ao cliente em questão. Terminada a atribuição das matches, com a exceção "declined" no caso de não existir nenhum expert adequado ao cliente em questão, procedemos para a atualização do experts requisitados.

De seguida, focámo-nos na escrita dos ficheiros que das matches e atualizações viriam a resultar.

Apercebemo-nos que a forma como tínhamos pensado em fazer a leitura dos headers não era a ideal (os métodos para esta circunstância estavam inseridos juntamente com a classe Expert) porque nos limitava e decidimos então criar a classe Header.

Depois de termos acabado esta nova classe, voltámos à escrita dos ficheiros, começando pela escrita do "Schedule" seguido da escrita dos "Experts" atualizados, tendo sido este o passo final para terminar a classe Schedule.

Passámos então à escrita da função "Main", cyberConc, onde implementámos a função update que tem como objetivo a assemblagem de todas as classes criadas, bem como as exceções e, finalmente, o argv que permite correr o projeto através da linha de comandos.

Não existem funcionalidades por implementar nem erros conhecidos.