O'Connor Airports

TURMA 3 – GRUPO 7

Davide Castro - up201806512
Diogo Rosário - up201806582
Henrique Ribeiro - up201806529

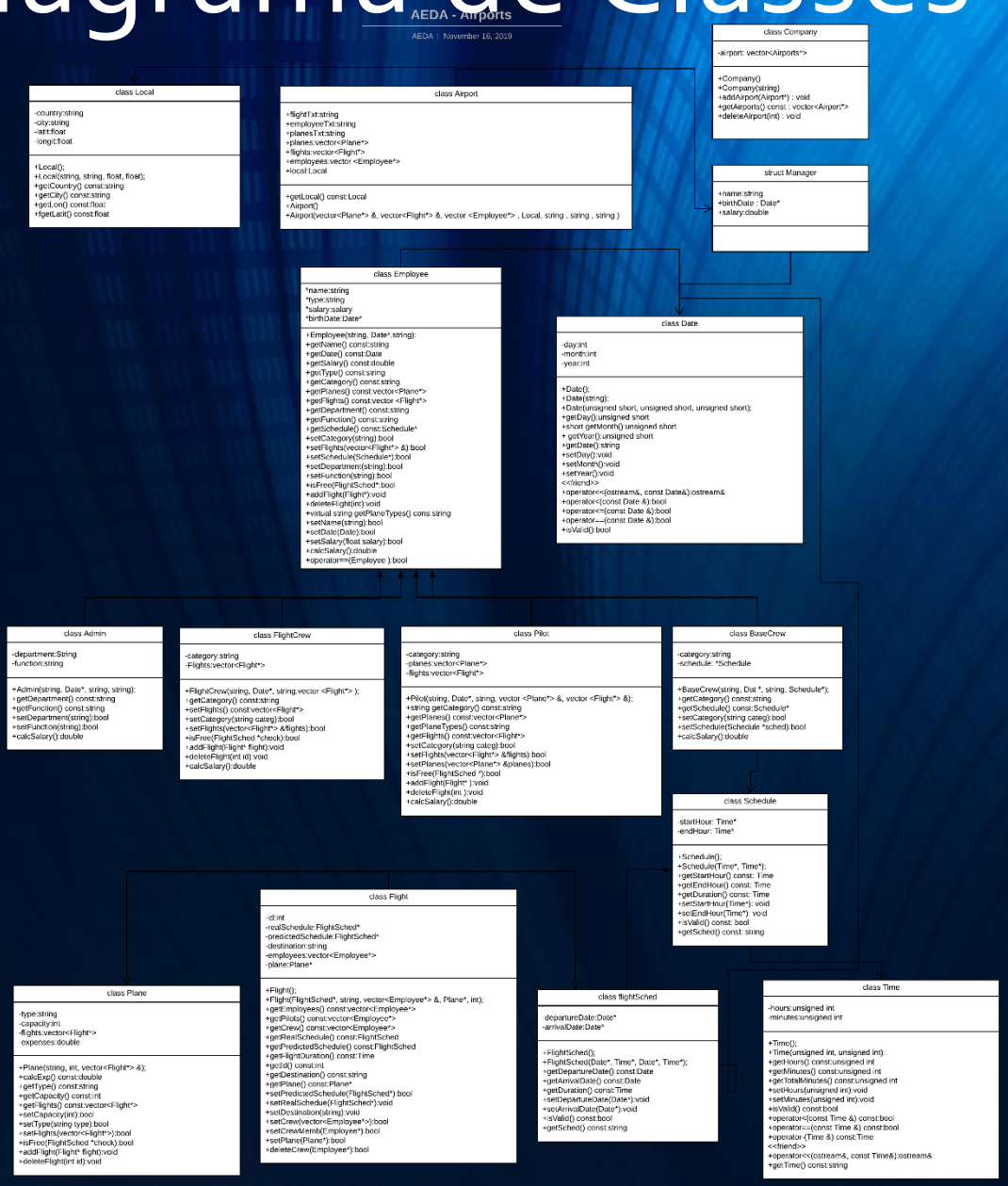
Problema

Efetuar a gerência de uma companhia aérea, incluindo os seus aeroportos, respetivos empregados, aviões e voos, sendo possível consultar todos os dados relevantes e alterá-los convenientemente.

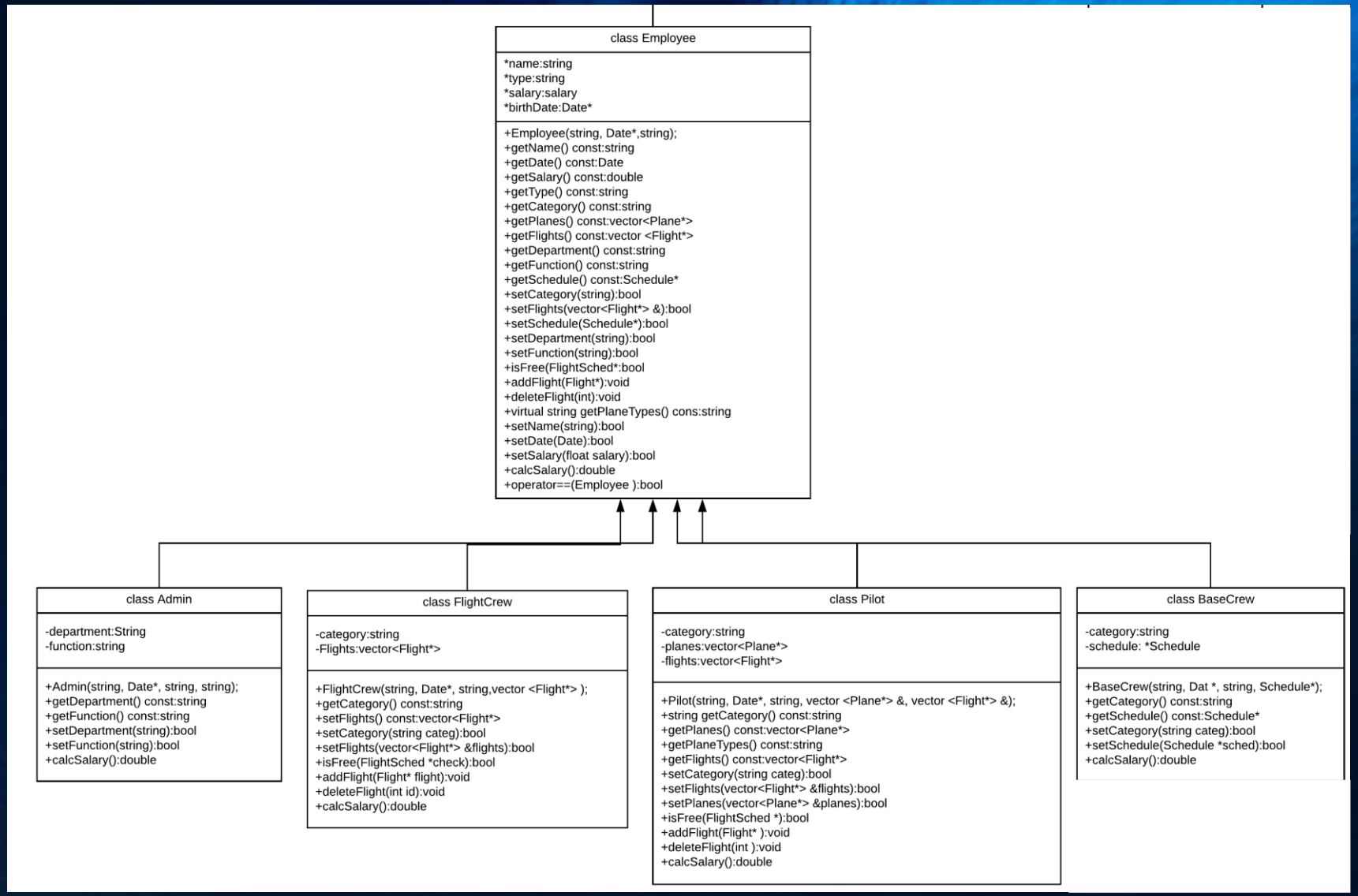
Solução

Desenvolvimento de uma aplicação, em C++, que permite a leitura dos dados sobre a companhia e a sua cadeia de aeroportos a partir de ficheiros pré-existentes, e, posteriormente, tratá-los com vista à possibilidade de criação ou eliminação de dados ou alteração dos já existentes. O programa utiliza várias estruturas de dados, em destaque, as classes e vetores. Utilizam-se os conceitos de herança e polimorfismo de forma a tornar o desevolvimento da aplicação mais eficiente.

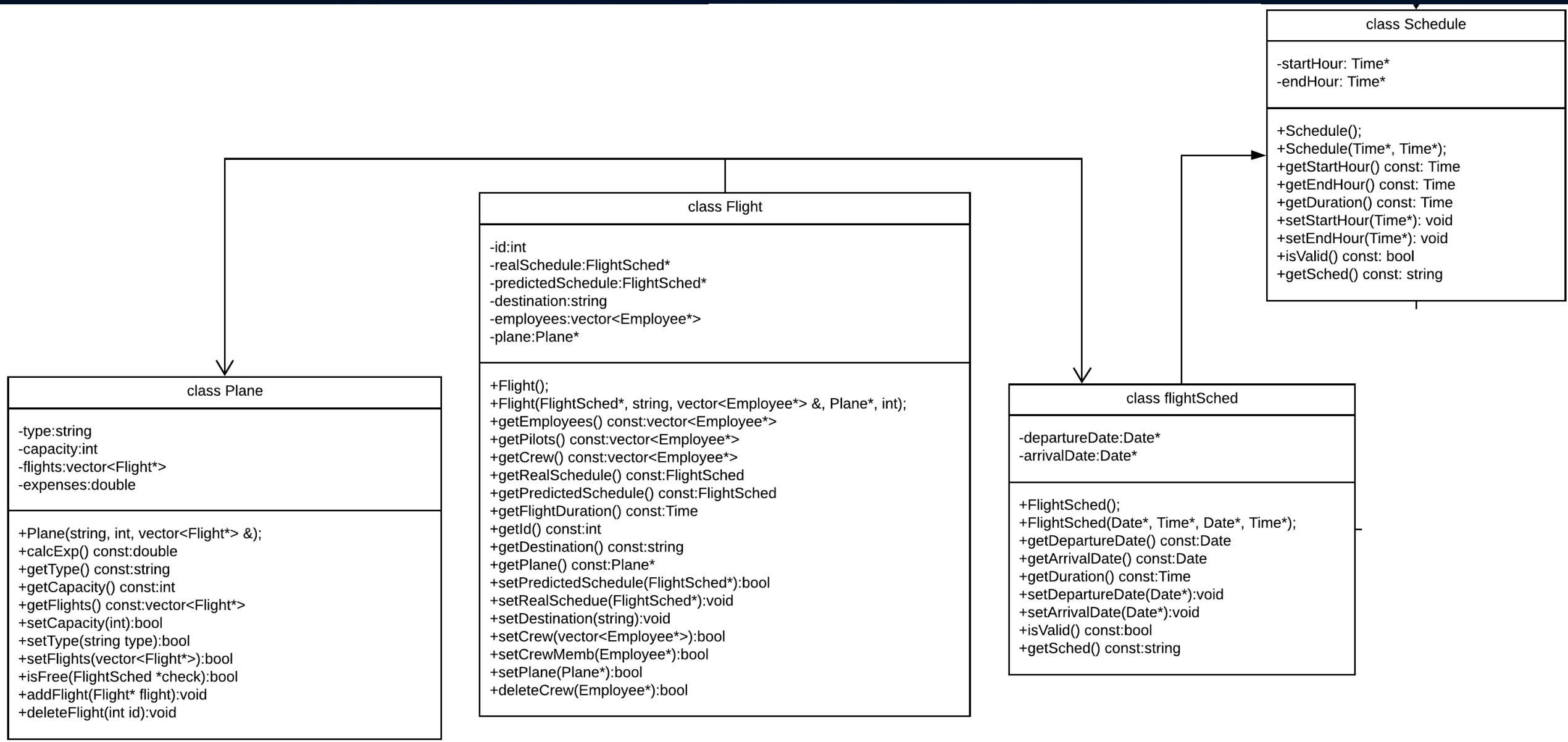
Diagrama de Classes



Classe Employee



Classe Flight e dependências



Estrutura de Ficheiros

Decidimos usar uma estrutura em que a companhia possui um ficheiro .txt com as informações de todos os aeroportos, que por sua vez têm o nome dos ficheiros dos respetivos empregados, voos e aviões.

Airports

```
Franca                      Cidade
Paris                       País
3.45                        Latitude
2.512                       Longitude
Paris_Flights.txt
Paris_Planes.txt
Paris_Employees.txt
Arnaldo                     Manager
6/9/1996
5000
::::::::::
Reino Unido
Londres
12
12
Londres_Flights.txt
Londres_Planes.txt
```

Employees

```
Pilot                       Type
Sum                         Name
12/4/1970                   Birth Date
A                           Category
A, B                        Plane types
1, 2                        Flights ID's
::::::::::
Pilot
Ting
25/3/1978
A
A, B
1
::::::::::
Flight Crew
Wong
```

Flights

```
1                           ID
9/11/2019                   Departure
10/11/2019                  Arrival
22                          Departure Hour
0                           Departure Minutes
0                           Arrival Hour
30                          Arrival Minutes
Porto                       Destination
::::::::::
2
12/11/2019
12/11/2019
21
0
23
0
London
```

Planes

```
A                           Category
120                         Capacity
1, 2                        Flights
::::::::::
B
200
2
```

Tratamento de exceções

- InvalidName
- InvalidCategory
- InvalidFlights
- InvalidBirthDate
- capacityError
- Exit (lançada quando o utilizador sai do pedido de input com CTRL+Z)

Exemplo: exceção InvalidCategory em readCategory e o seu tratamento

```
string readCategory()
{
    string category;
    cin >> category;
    if (cin.eof()) {
        cin.clear();
        throw Exit();
    }
    cin.ignore(100, '\n');
    if (cin.fail() || ((category != "A") && (category != "B") && (category != "C")))
    {
        cin.clear();
        throw InvalidCategory(category);
    }
    return category;
}
```

```
cout << "-------------------------------------------------------------------------------\n";
cout << "Category: \n";
do
{
    badInput = false;
    try
    {
        category = readCategory();
    }
    catch (Exit ex)
    {
        cout << ex.getMsg() << endl;
    }
    catch (InvalidCategory cat)
    {
        cout << "Invalid category " << cat.getCategory() << "! Please insert pilot's category again (A, B or C)\n";
        badInput = true;
    }
} while (badInput);
```

Funcionalidades

Create, update, read e delete de empregados, aviões, voos, aeroportos e manager.

- Pesquisar voos entre duas datas indicadas pelo utilizador e ver se estão prontos (avião associado e tripulação completa)
- Adicionar membros de tripulação, pilotos e aviões a voos existentes
- Pesquisar empregados por nome ou categoria, ou também aviões por categoria
- Consultar despesas do aeroporto tanto total como por secções (empregados e derivadas, aviões e voos)
- Mudar e eliminar qualquer tipo de dados tendo em conta as implicações nos dados relacionados.
- Criar, ver, mudar, eliminar e alternar entre diferentes aeroportos para os consultar mais detalhadamente e geri-los

Funcionalidades - **Read**

Exemplo: pesquisar empregado por nome

Input: Sum

Output:

```
|-----------------------------------------------------------------------------------------------|
|-----------------------------------------------------------------------------------------------|
|              Name | Sum                                                                       |
|-----------------------------------------------------------------------------------------------|
|        Birth Date | 12/4/1970                                                                 |
|-----------------------------------------------------------------------------------------------|
|          Category | A                                                                         |
|-----------------------------------------------------------------------------------------------|
|            Salary | 1200                                                                      |
|-----------------------------------------------------------------------------------------------|
|       Plane types | A, B                                                                      |
|-----------------------------------------------------------------------------------------------|
|           Flights | 1, 2                                                                      |
-------------------------------------------------------------------------------------------------
```

Pesquisa todos os empregados com o nome ‘Sum’ e mostra-os na consola. Avisa se não houverem empregados com o nome desejado ou se ainda não houverem empregados no aeroporto, e pede outro nome.

Funcionalidades - **Create**

Exemplo: criar membro de tripulação

```
------------------------------------------------------------------------------------------------------------
Name:
Jose Silva
------------------------------------------------------------------------------------------------------------
BirthDate (dd/mm/yyyy):
12/12/1975
------------------------------------------------------------------------------------------------------------
Category:
B
------------------------------------------------------------------------------------------------------------
Flights:
5
One of more of those flights ID's don't exist!
Do you want to create this employee with no flights?
y
------------------------------------------------------------------------------------------------------------
New Flight Crew member successfuly created!
------------------------------------------------------------------------------------------------------------
1)Create another person.
0)Return to the last menu.
```

Pede todas as informações sobre o novo membro, incluindo voos onde participar. O programa verifica quais voos têm tripulação incompleta e só aceita esses no input do utilizador. Caso contrário é dada a opção de criar o empregado sem voos associados por enquanto

Funcionalidade em destaque

Delete Pilot

- Eliminar piloto

Verificar se ele está associado a algum voo

- Se sim, aceder aos voos correspondentes e tirar o piloto

Verificar quais pilotos estão livres para o substituir

- Pedir ao utilizador um piloto substituto se houverem disponíveis naquela data e hora

Guardar as alterações nos voos e no novo piloto que foi associado

Devido às implicações que eliminar um piloto tem no resto dos dados (voos, pilotos, aeroporto) esta função é um exemplo de uma daquelas que foi mais trabalhosa de implementar

Principais impasses no projeto

- Planear a estrutura dos ficheiros, de modo a que não existissem dependências entre eles. Por exemplo: Os voos e aviões estão associados entre eles, visto que os voos contêm um avião e os aviões contêm os voos que vão realizar. Por isso, não seria possível ler um ficheiro de aviões sem ler os voos e vice-versa. Solução: criação de ID's para os voos e cada avião teria os ID's dos voos associados no ficheiro.

- Verificação de inputs, destacando-se o input de ID's de flights (flight1, flight2, …) em que tivemos de usar funções para decompor inputs após os verificar.

- Criação e mudança de dados, devido à quantidade de verificações necessárias e as numerosas implicações no resto dos dados.