Departamento de Matemática

FCTUC

Relatório

Métodos de Programação II

Professor: José Carlos Teixeira ([teixeira@mat.uc.pt](mailto:teixeira@mat.uc.pt))

Diário de viagens

**Autores:**

Ana Marques

anaritamnmarques@hotmail.com

Luís Domingues

l\_evora\_d@hotmail.com

**Data:**

06-01-2017

ÍNDICE

[1. Sumário 3](#_Toc462994391)

[2. Desenvolvimento 4](#_Toc462994392)

[2.1 Requisitos da solução 4](#_Toc462994393)

[2.2 Arquitetura da solução 4](#_Toc462994394)

[2.3 Interface com o utilizador 4](#_Toc462994395)

[2.4 Estruturas de dados 4](#_Toc462994396)

[2.5 Principais funções 4](#_Toc462994397)

[3. Apresentação de uma sessão de utilização 5](#_Toc462994398)

[3.1 Entrada 5](#_Toc462994399)

[3.2 XXX 5](#_Toc462994400)

[4. Conclusões 6](#_Toc462994401)

[5. Bibliografia 7](#_Toc462994402)

# Sumário

Este trabalho foi solicitado pelo professor José Carlos Teixeira, no âmbito da disciplina de Métodos de Programação II, assim como os seus principais objetivos.

Este trabalho consiste, sumariamente na realização de uma aplicação capaz de facilitar a gestão de informação relacionada com viagens.

Esta aplicação permite consultar e analisar informação referente aos países e cidades visitadas, ao número de quilómetros, às despesas e aos dias de viagem. Para todas estas informações é possíveis consultar o valor total, por viagem, e por ano (sendo também possível obter o valor de quilómetros percorridos por meio de transporte). Assim como os seus valores percentuais. Para uma melhor análise dos resultados está disponível um relatório, quer no ecrã, quer num ficheiro de texto, com os países onde visitou mais cidades e onde gastou mais dinheiro ordenados alfabeticamente.

Como funcionalidades extra a aplicação permite

< o conteúdo geral do trabalho [breve resumo dos pontos/secções do relatório] e as principais conclusões/resultados.>

O relatório foi dividido de modo a se explicar primeiro a declaração de dados e variáveis e depois como era feita a leitura de dados introduzidos pelo utilizador. De seguida, apresenta-se a função *insereord* responsável pela inserção ordenada dos produtos numa lista e apresentam-se, de seguida, os relatórios que o programa permite elaborar. Ainda há pontos para explicar as funções *maiorestres*, responsável pela apresentação dos três produtos mais valiosos, *maiorativosarea*, responsável pela apresentação da área de actividade com maior valor de activos e *eliminaproduto*, responsável pela eliminação de um produto escolhido pelo utilizador. Por último, há um tópico onde se apresentam os processos de escrita de dados e, finalmente, os menus, onde também se explica as funções *alterarpreco* e *alteraquant.*

# Desenvolvimento

## Requisitos da solução

Neste programa têm de ser considerados, pelo menos: dois modos de transporte (carro e avião), datas de início e de fim de viagem, países e cidades visitadas durante a viagem, número de quilómetros percorridos em cada viagem e custos associados a cada viagem.

O carregamento da informação pode ser feito de duas formas: carregamento direto no sistema e através de um ficheiro. A informação adicionada ao programa de gestão não deve ser perdida.

Utilizando uma forma dinâmica de organizar e gerir a informação, desenvolva um programa modular e eficiente que lhe permita consultar e analisar, pelo menos, a seguinte informação:

1. Países e cidades visitadas: número e nomes, total e por ano;

2. Número de km percorridos: total, por ano e por meio de transporte;

3. Despesas: total, por viagem e por ano;

4. Dias de viagem: total, por viagem e por ano;

5. Valores percentuais das consultas anteriores;

6. Relatórios, em ficheiros e no ecrã e por ordem alfabética, dos países onde visitou mais cidades e onde gastou mais dinheiro.

## Arquitetura da solução

Este programa começa pela função *criaCalendario* que cria a base para a lista principal (struct Ano), de seguida com a função *carregaFicheiro* lê o ficheiro com a informação necessária que é guardada na lista, ou se ao tentar ler o ficheiro este nao existir cria um ficheiro vazio, procede para a função *menu* onde, como o nome indica, permite ao escolher uma das seguintes opçôes: Adicionar viagem (função *criaViagem*); Consultar informação (função *consultaInformacao*); Consultar informação percentual (função *consultaPercentagens*); Criar relatório no ecrã (função *criaRelatorioE*); Criar relatório num ficheiro (função *criaRelatorioF*); Sair.

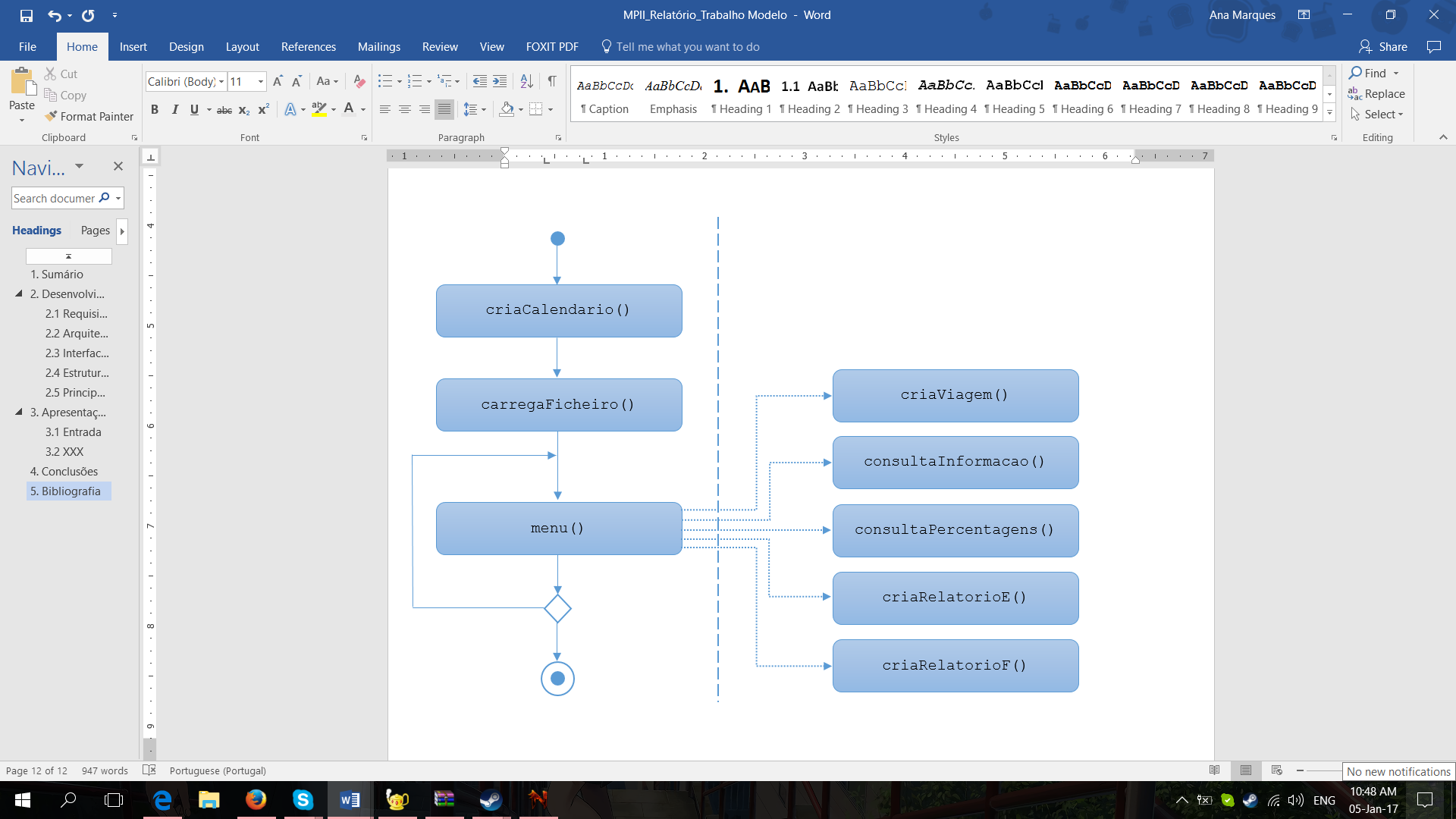


Figura 1 - Arquitetura da solução

## Interface com o utilizador

<Apresentação da conceção da interface com o utilizador, podendo ser apresentados esboços (*mock-ups*) que suportem esta apresentação.>

## Estruturas de dados

São utilizadas várias estruturas de dados nesta aplicação, a principal é uma lista simplesmente ligada chamada *struct Ano*, esta serve de base a duas listas também simplesmente ligadas, *struct Viagem* e *struct Cidade*. Para além destas existe uma estrutura de dados denominada *struct Data* que não é uma lista porém é utilizada no *struct Viagem*.

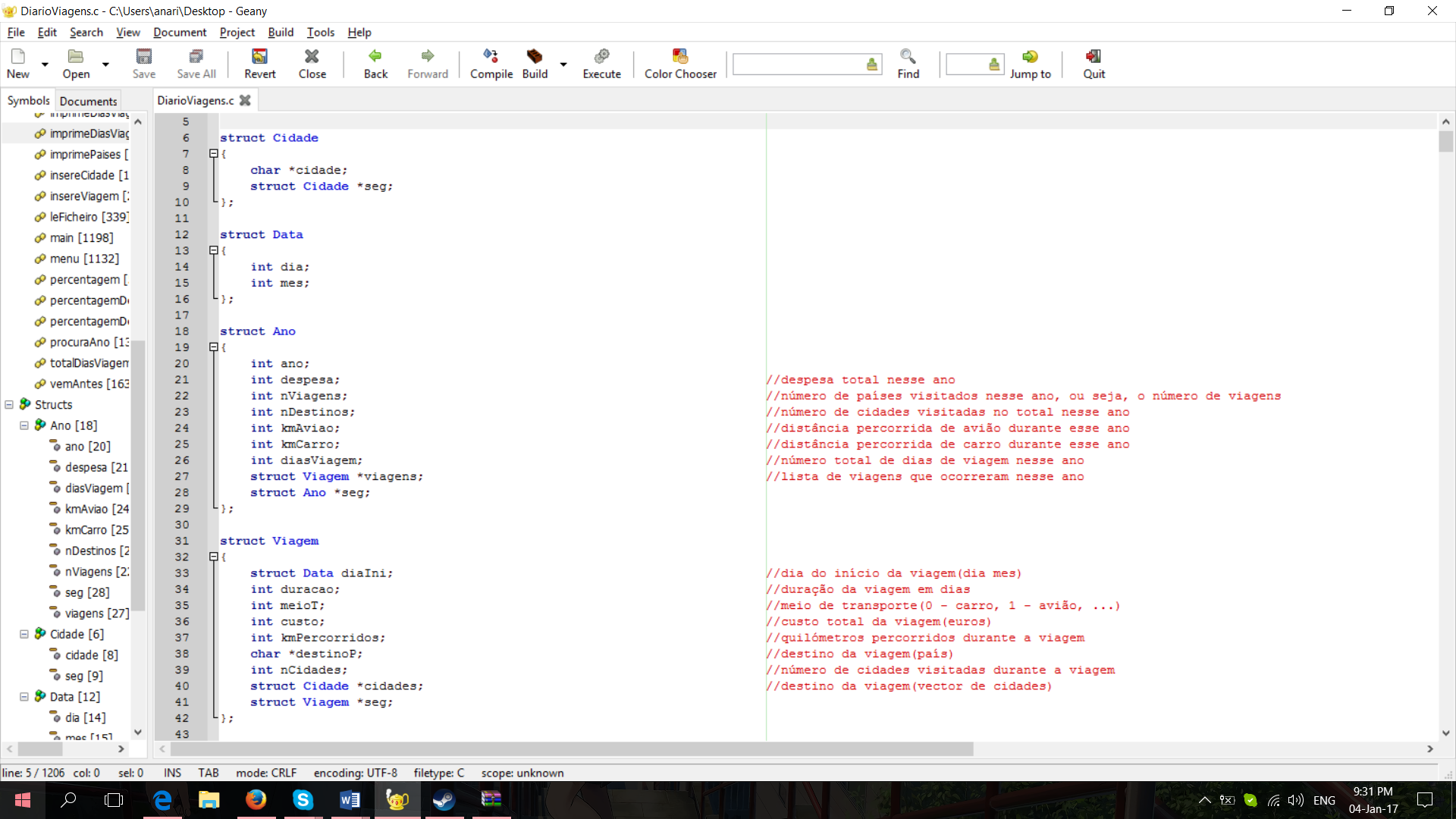


Figura 2 - Definição das estruturas (diarioviagens.h)

Variáveis utilizadas na lista *struct Ano* :

* **ano:** Número inteiro correpondente ao ano de início de viagem;
* **despesa:** Número inteiro correpondente à despesa total nesse ano;
* **nViagens:** Número de viagens visitados nesse ano.
* **nDestinos:** número de cidades visitadas no total nesse ano.
* **kmAviao:** Número inteiro correpondente à distância percorrida de avião durante esse ano.
* **kmCarro:** Número inteiro correpondente à distância percorrida de carro durante esse ano.
* **diasViagem:** Número inteiro correpondente ao total de dias de viagem nesse ano.

Variáveis utilizadas na lista *struct Viagem* :

* **struct Data diaIni:** Estrutura de dados que possui o dia do início da viagem (um inteiro *dia* e outro inteiro *mes*).
* **duracao:** Número inteiro correspondente à duração da viagem em dias.
* **meioT:** Número inteiro correspondente ao meio de transporte utilizado (1 – avião ou 2 - carro).
* **custo:** Número racional correspondente ao custo total da viagem (em euros).
* **kmPercorridos:** Número inteiro correspondente aos quilómetros percorridos durante a viagem.
* **destinoP:** Diz respeito a uma *string* de número variável de caracteres, a memória que esta *string* ocupa foi previamente dinamizada pela função *ajustaMemoria*, que armazena o destino da viagem (país).
* **nCidades:** Número inteiro correspondente às cidades visitadas durante a viagem.

Variáveis utilizadas na lista *struct Cidade* :

* **cidade:** Diz respeito a uma *string* de número variável de caracteres, a memória que esta *string* ocupa foi previamente dinamizada pela função *ajustaMemoria*, que armazena o nome de uma cidade.

As ligações entre as várias listas estão ilustradas na figura a baixo. Onde *Base* se refere a uma variável global denominada *calendario* que é do tipo *struct Ano* que serve de base para tudo o resto.

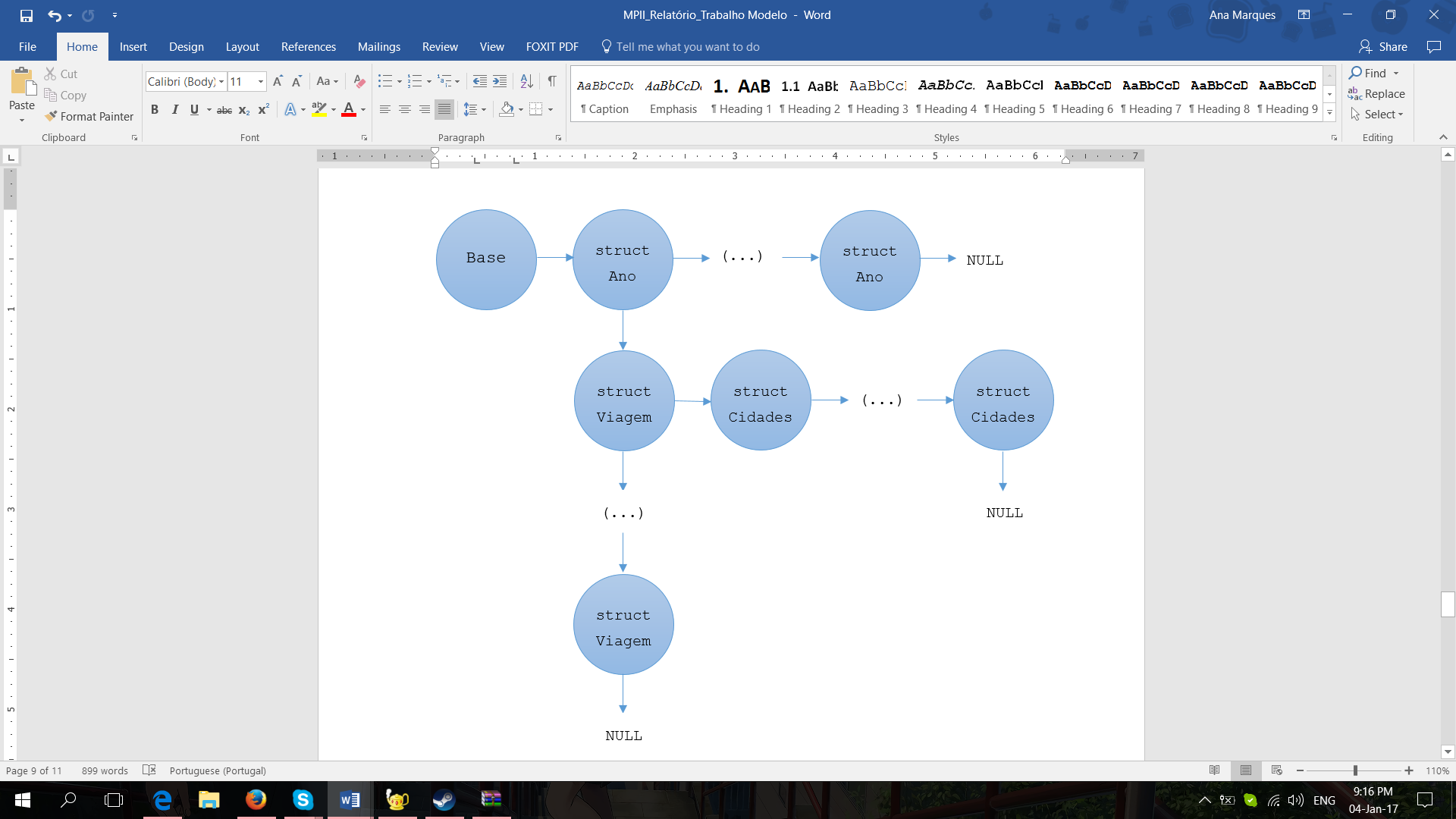


Figura 3 – Representação gráfica das listas

## Principais funções

A leitura de dados tem diversas fases, com recurso a várias funções.

1. *criaCalendario*: inicializa os valores da variável global *calendario.*
2. *carregaFicheiro*: prepara o ficheiro “DiarioViagens.txt” para poder ser lido pela função *leFicheiro*. Se não existir o ficheiro “DiarioViagens.txt” este é criado vazio.
3. *leFicheiro*: le recursivamente linha a linha até chegar ao fim do ficheiro. Em cada chamada do leFicheiro, ou seja em cada linha do ficheiro, le uma string ate chegar a um ‘;’ ou ‘\n’ ()
4. *criaViagem*: abre “DiarioViagens.txt” para escrita, lê do teclado a data de início de viagem, verifica se esta é válida, utilizando a função *verificaData* e se não for válida volta a ler a data de início de viagem. Quando esta for válida escreve-a no ficheiro com o formato pertendido para leitura (ou seja, com ‘;’ entre cada dígito). Em relação ao ano desta data, vai ser verificado se já existe uma viagem com este ano (com a função procuraAno), se já existir nao se faz nada, porém se for um ano novo este vai ser adicionado com a função *adicionaAno*. Todas as restantes variáveis da estrutura Viagem vão ser lidas apartir do teclado e escritas no ficheiro no formato anteriormente especificado.
5. *insereViagem*:

<Apresentação das funções mais significativas, podendo ser usado pseudo-código. Não deve ser usado o código-fonte desenvolvido, exceto nalguma particularidade que o justifique.>

# Apresentação de uma sessão de utilização

## Entrada

<Apresentação da forma de entrada na aplicação desenvolvida.>

## XXX

<Apresentação das diferentes vias de utilização da aplicação. Devem ser usadas imagens de suporte.>

# Conclusões

<Resumo do que foi feito e de como foi feito, dos resultados, das eventuais dificuldades encontradas e das melhorias e/ou perspetivas futuras que sejam identificadas.>

# Bibliografia

É desejável que sejam incluídas referências a livros e outros elementos com validação científica, e não apenas a fontes na Internet.

Para a identificação no texto e nesta secção, devem ser usados os mecanismos automáticos dos editores de texto, por exemplo inteiros.

Deverão ser usados os seguintes formatos na identificação das fontes bibliográficas:

**Informação impressa:**

Nome do Autor, Título, Local de publicação: Editor, Ano de publicação

• Exemplos:

CHOO, C. W., *Information Management for the Intelligent Organization: The Art of Scanning the Environment*, 2nd ed. Medford, NJ: Information Today, 1998.

AUSTER, Ethel and CHOO, Chun Wei (eds.), *Managing Information for the Competitive Edge*, New York: Neal-Schuman Publishers, 1996.

**Fontes de informação na Internet**

Nome do Autor (se for referido), Título, Online, Local de Publicação / criação do site (se referido): Editor (se referido).

URL: (Endereço na Internet) [Conferido em: ]

LEITE, Jair C., *Notas de aula de Engenharia de Software*, Universidade Federal do Rio Grande do Norte, 2000.

URL: <https://www.dimap.ufrn.br/~jair/ES/c7.html> [Conferido em: 3 de Janeiro 2017].

NULL

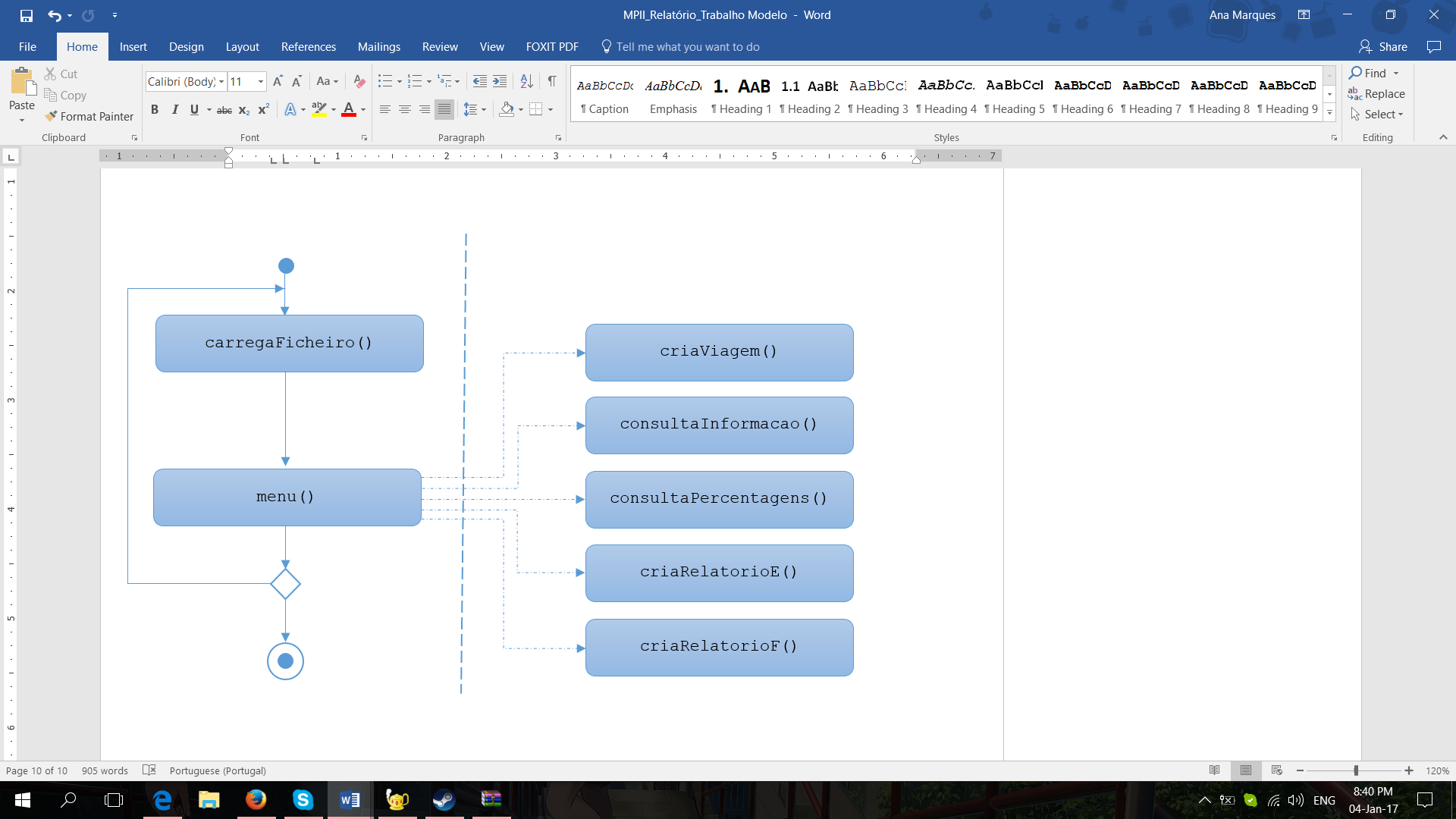
(...)

(...)

NULL

(...)

NULL



criaCalendario()

carregaFicheiro()

criaViagem()

criaRelatorioF()

criaRelatorioE()

consultaPercentagens()

consultaInformacao()

menu()