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 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 utilizar um meio de transporte denominado “Comboio”.

O relatório está dividido de modo a se introduzir primeiramente os requisitos dados pelo professor José Carlos Teixeira, no âmbito da disciplina de Métodos de Programação II, de seguida encontra-se a arquitetura da aplicação que facilita a sua utilização, seguido do interface com o utilizador, depois as estruturas de dados e por fim as principais funções.

# 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.

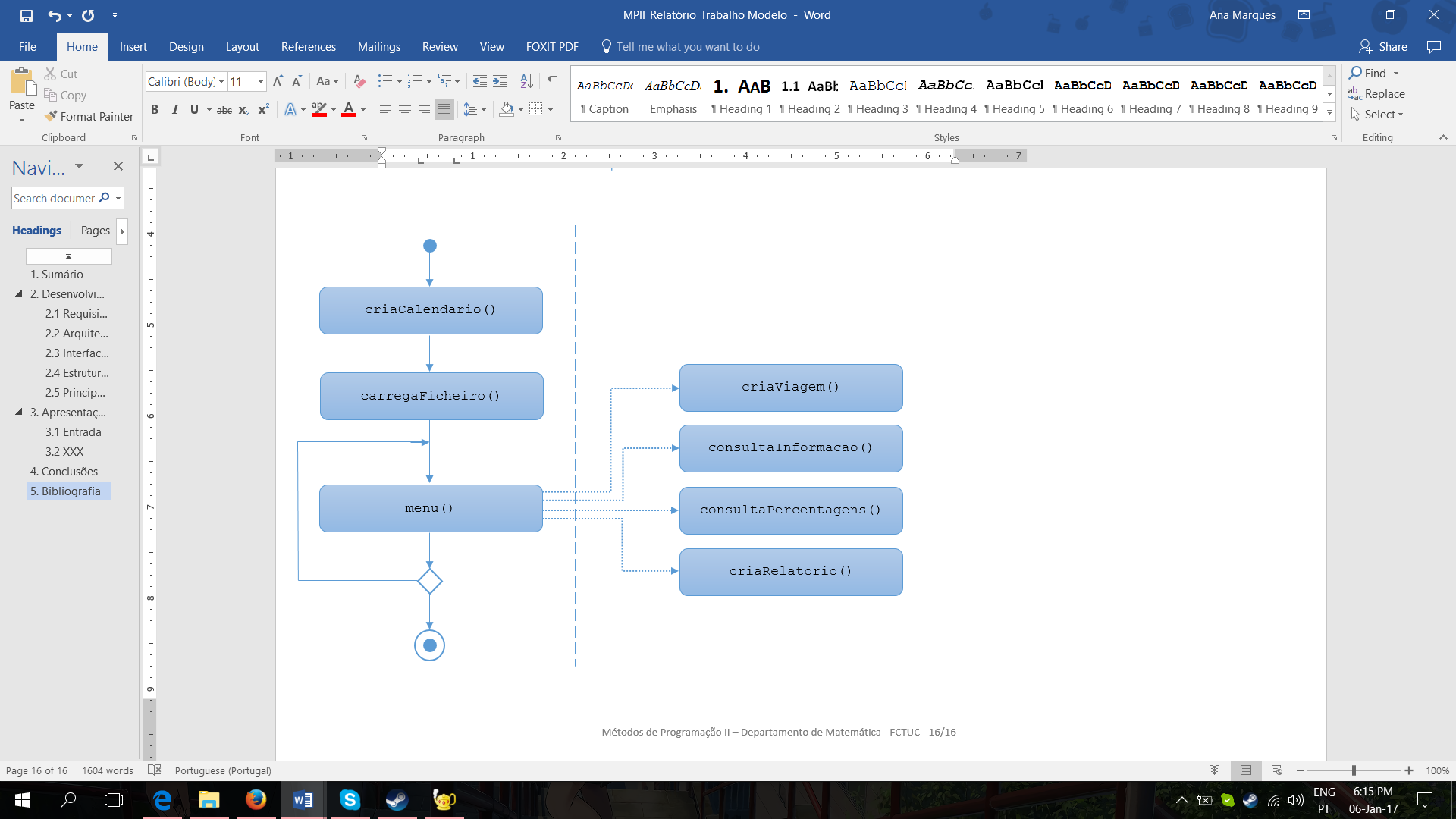


Figura 1 - Arquitetura da solução

## Interface com o utilizador

Assim que o programa é iniciado, aparece o seguinte menu:

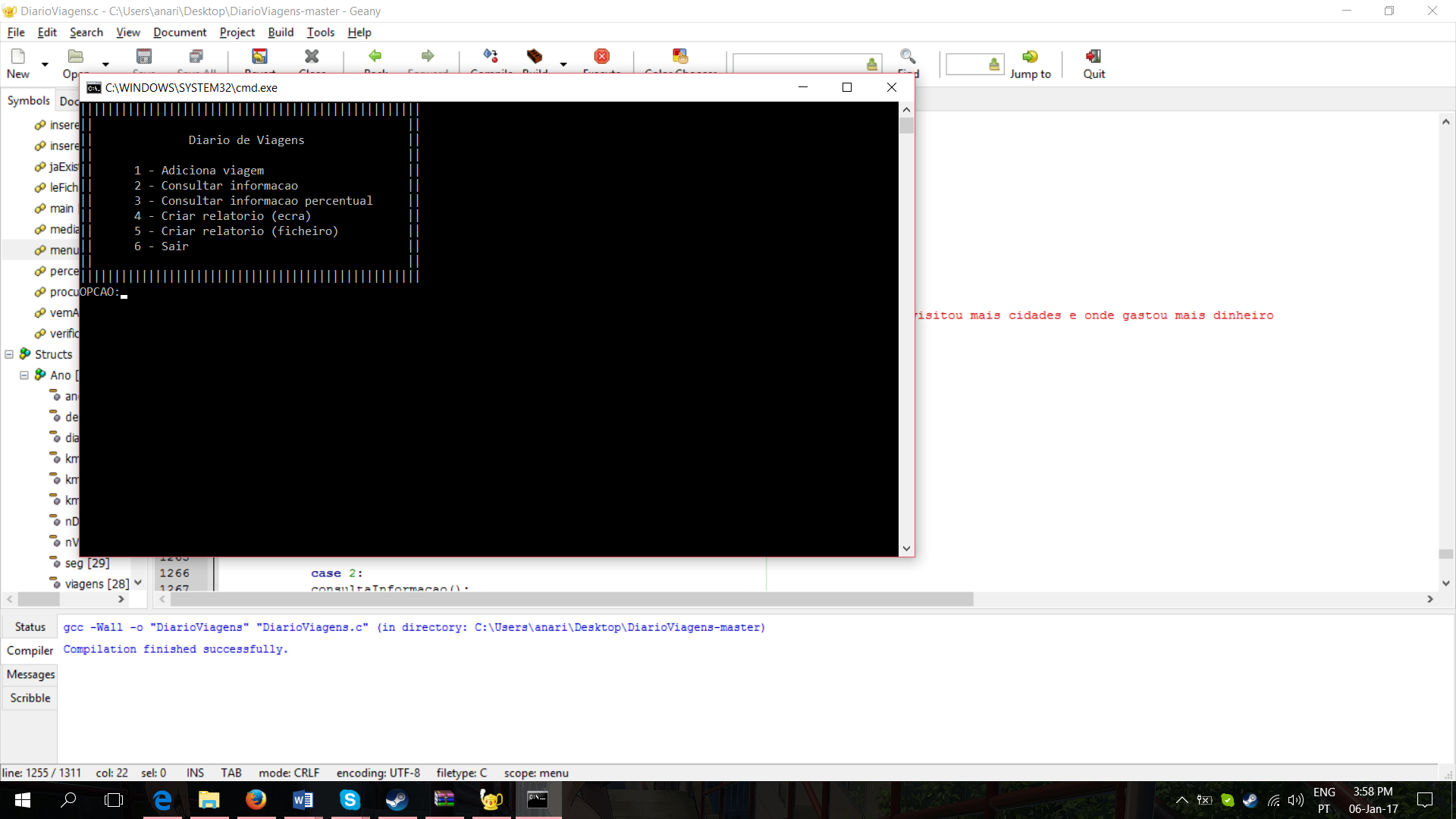


Figura 2 - Menu inicial

Caso o utilizador selecione a opção 1, a leitura da informação referente à viagem é feita individualmente. É executada a funções *criaViagem* e dentro desta a função *insereViagem*.

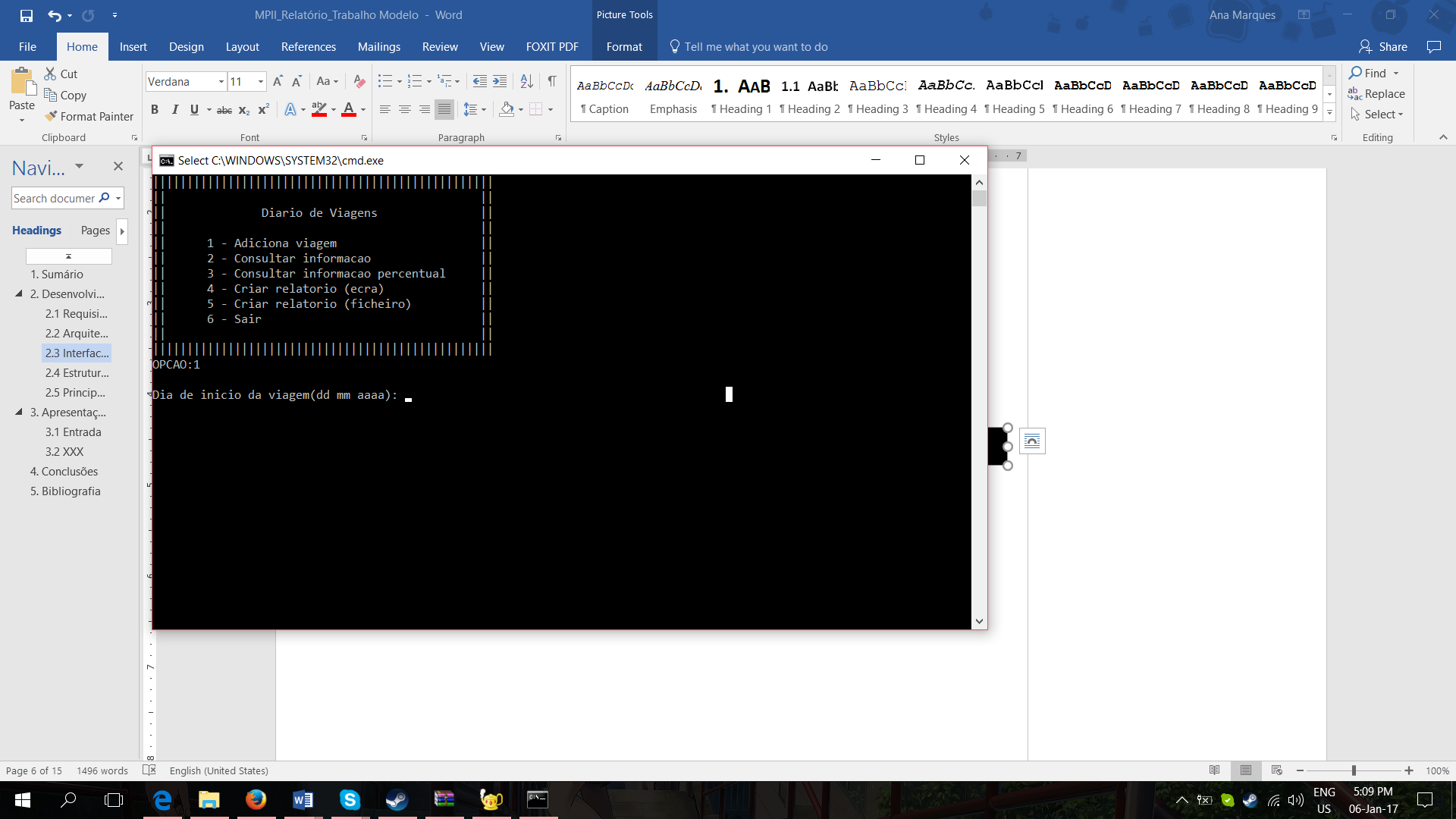


Figura 3 – Leitura da data de início da viagem

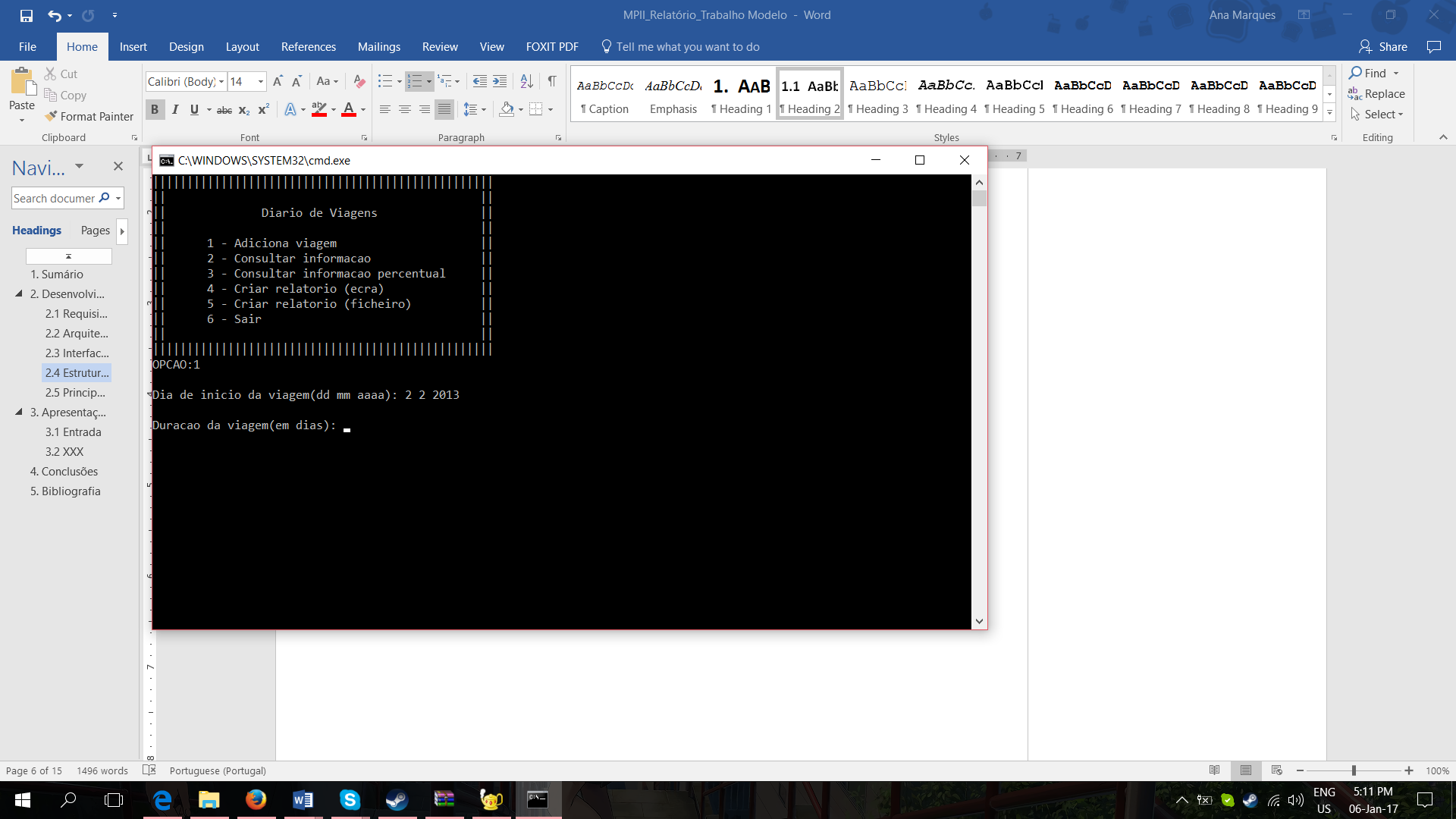


Figura 4 – Leitura da duração da viagem em dias

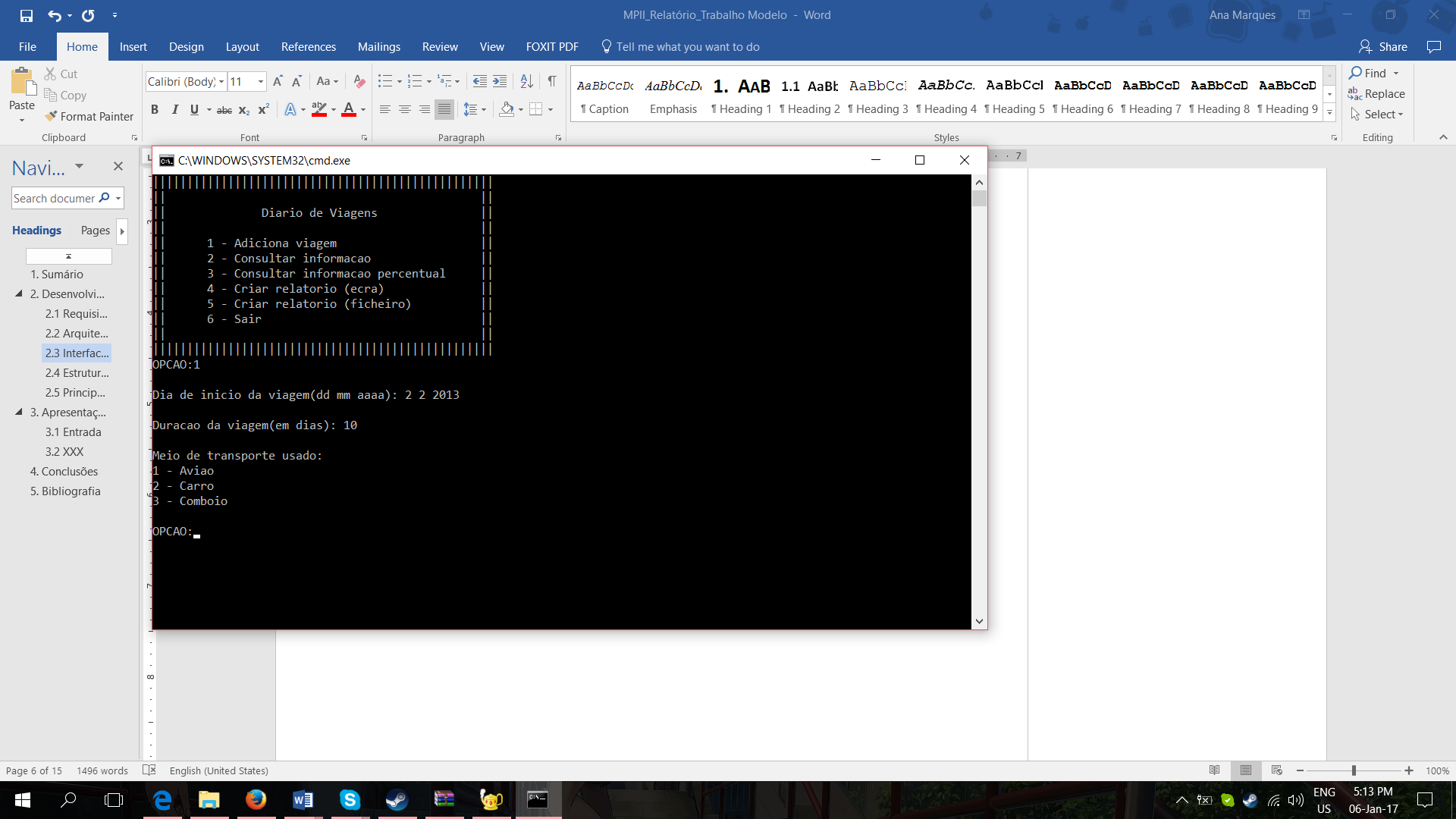


Figura 5 – Leitura do meio de transporte(1, 2 ou 3)

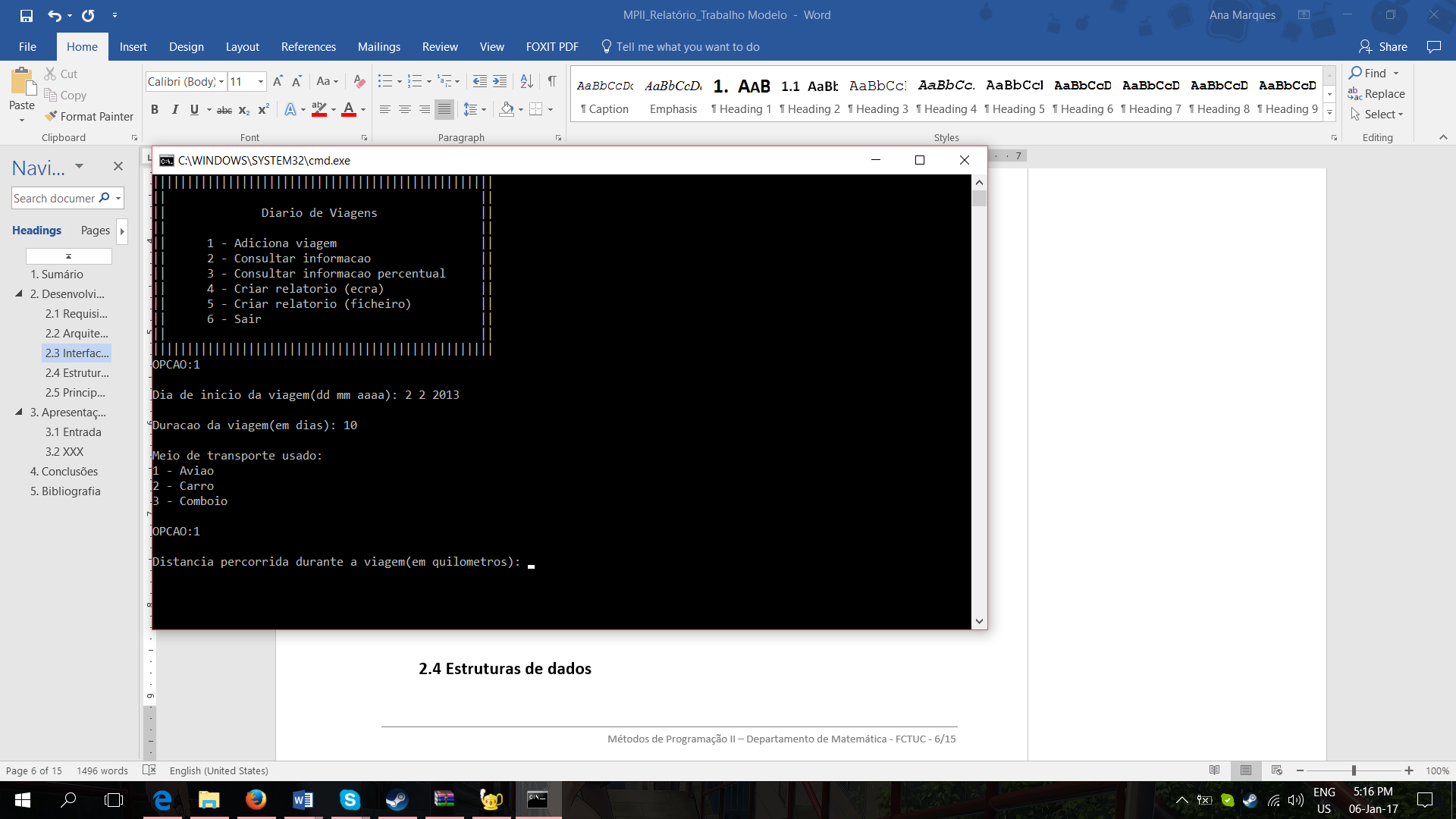


Figura 6 – Leitura da distancia percorrida durante a viagem em quilómetros

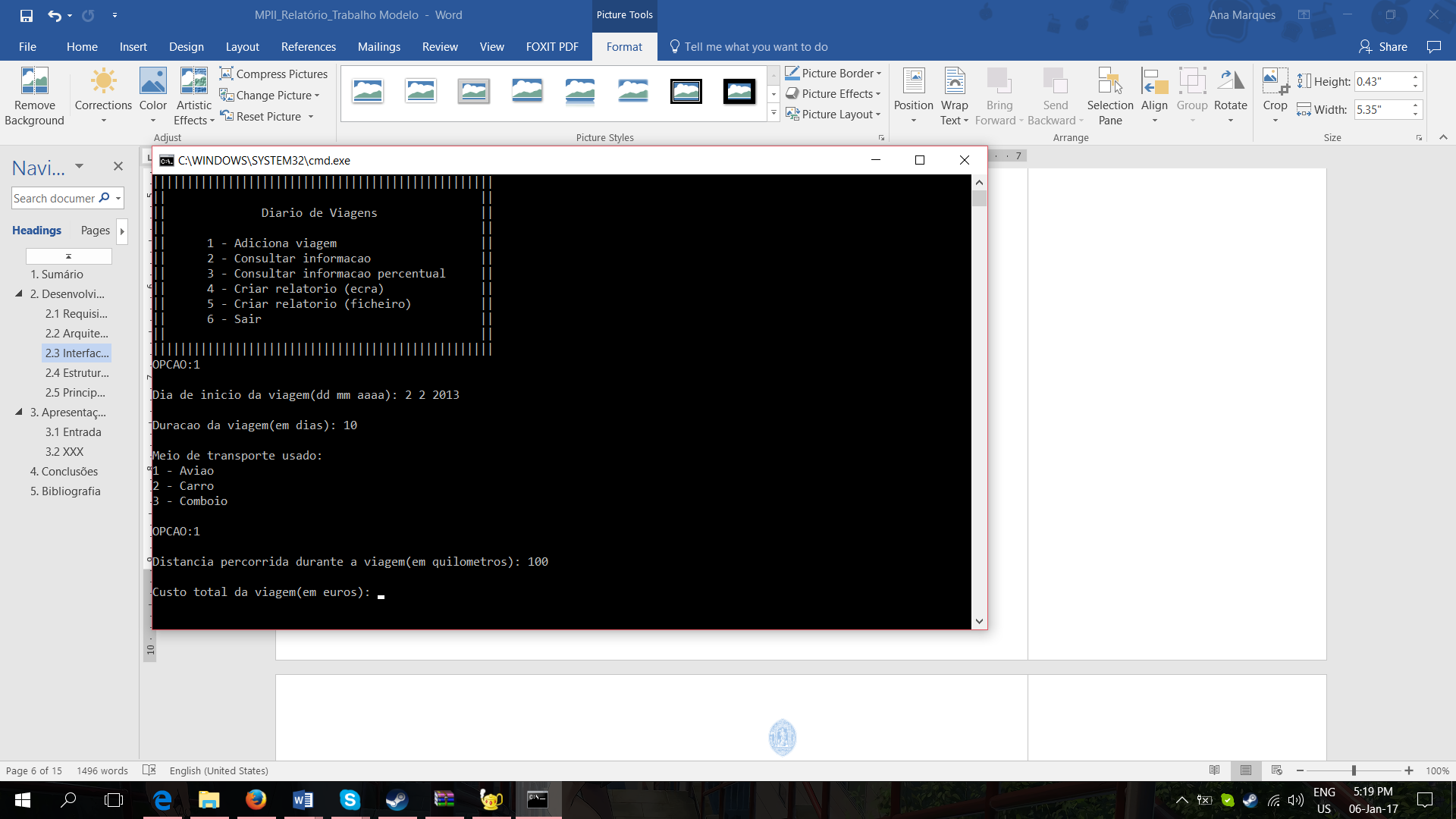


Figura 7 – Leitura do custo total da viagem em euros

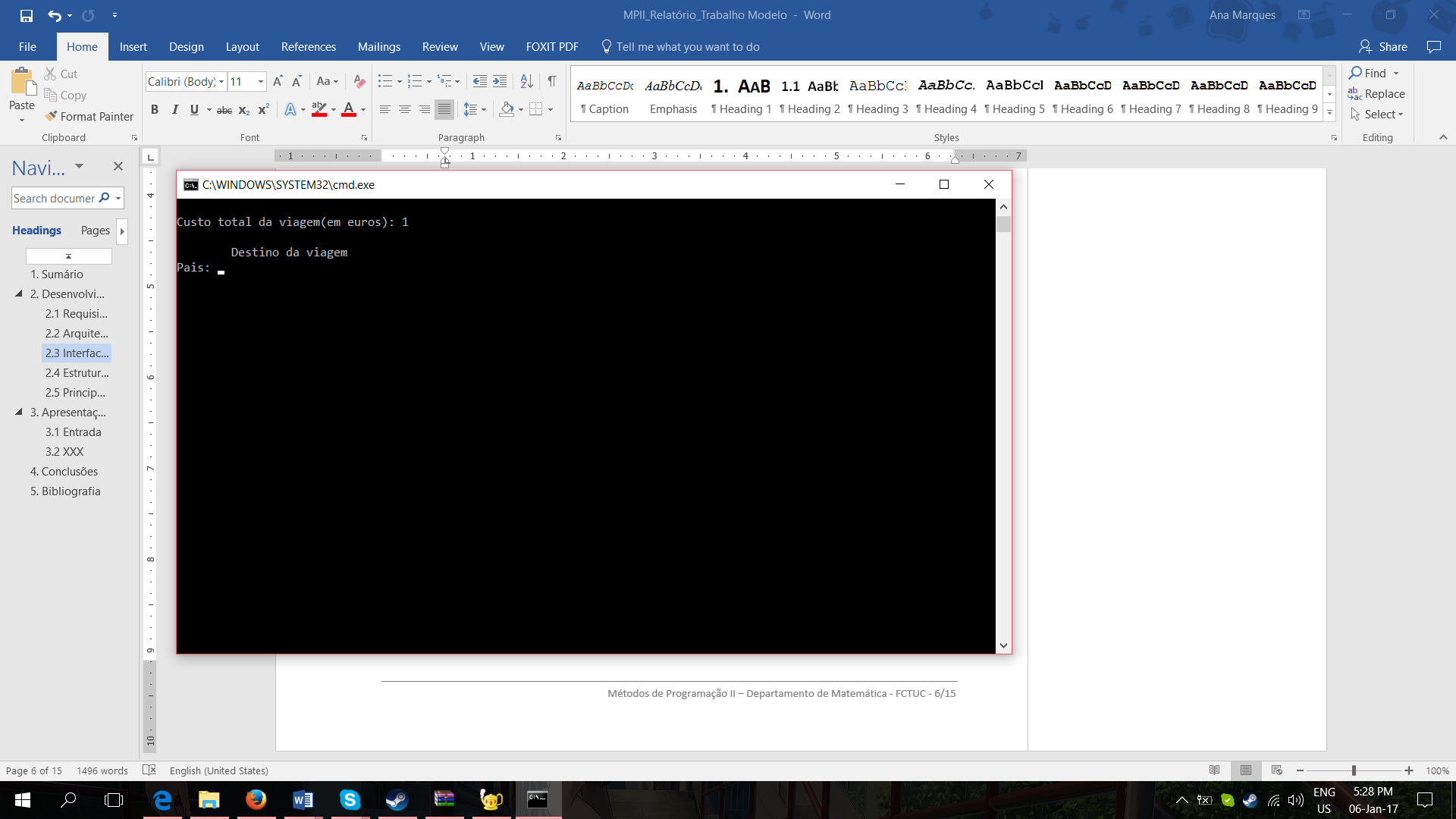


Figura 8 – Leitura do país a que se destinou a viagem

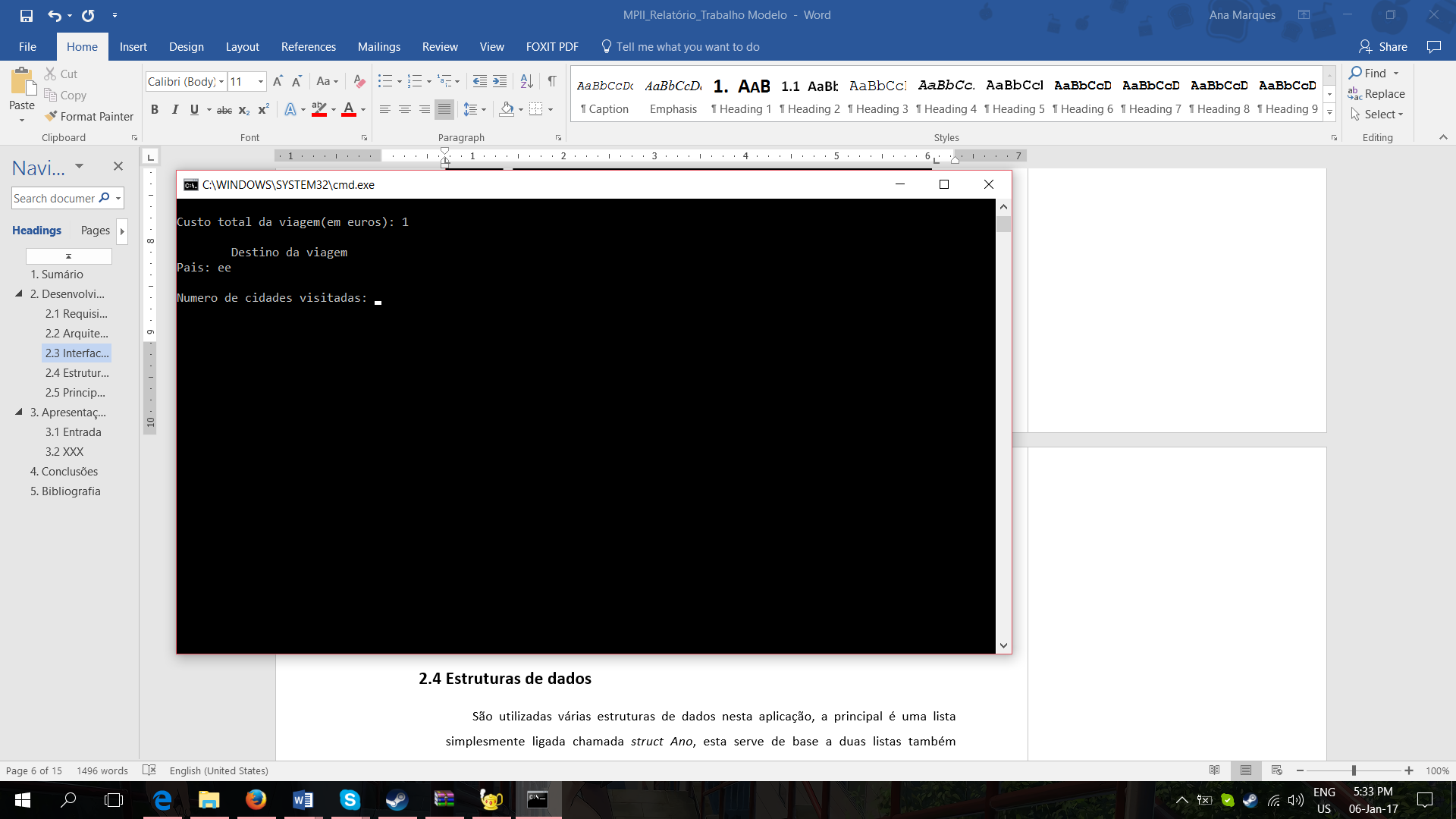


Figura 9 – Leitura do número de cidades visitadas nesse país

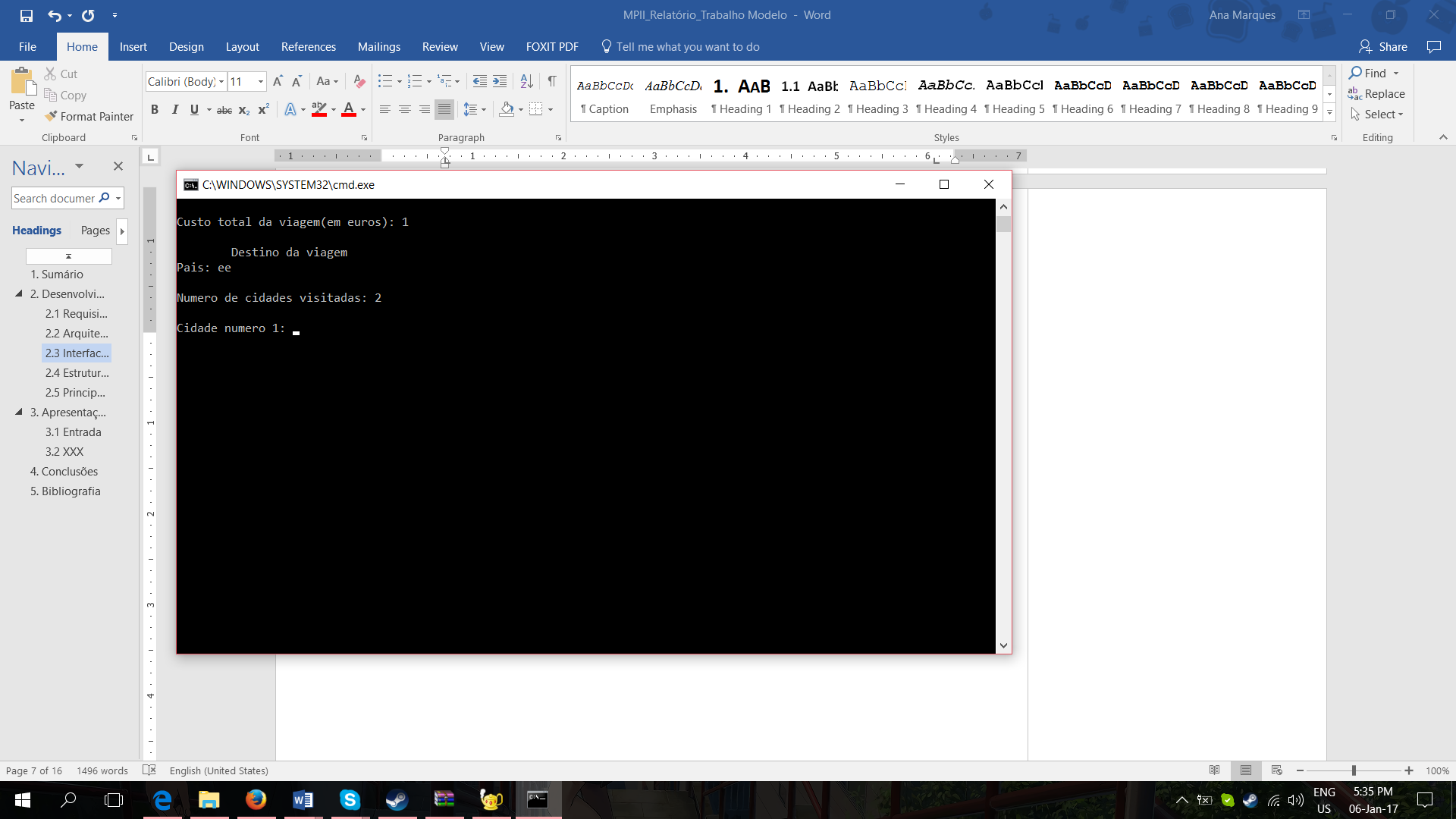


Figura 10 – Leitura sequencial das várias cidades numeradas de 1 ao número anteriormente inserido

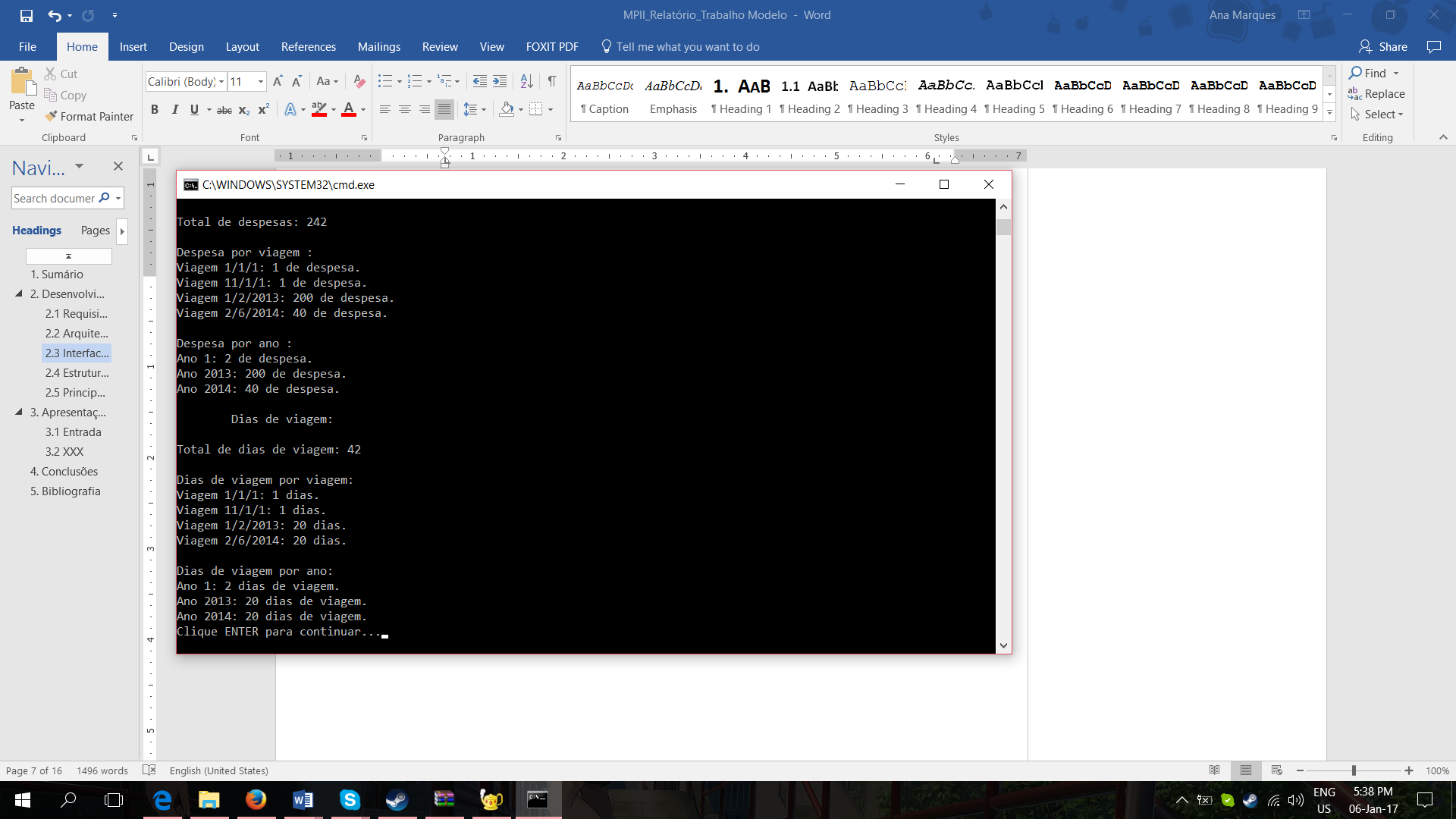


Figura 11 – No fim de cada opção é necessário pressionar a tecla ENTER

Caso o utilizador selecione a opção 2, irá mostrar as informações mais detalhadas em relação a todas as viagens até ao momento realizadas. Como exemplo: Figura 12a e 12b.

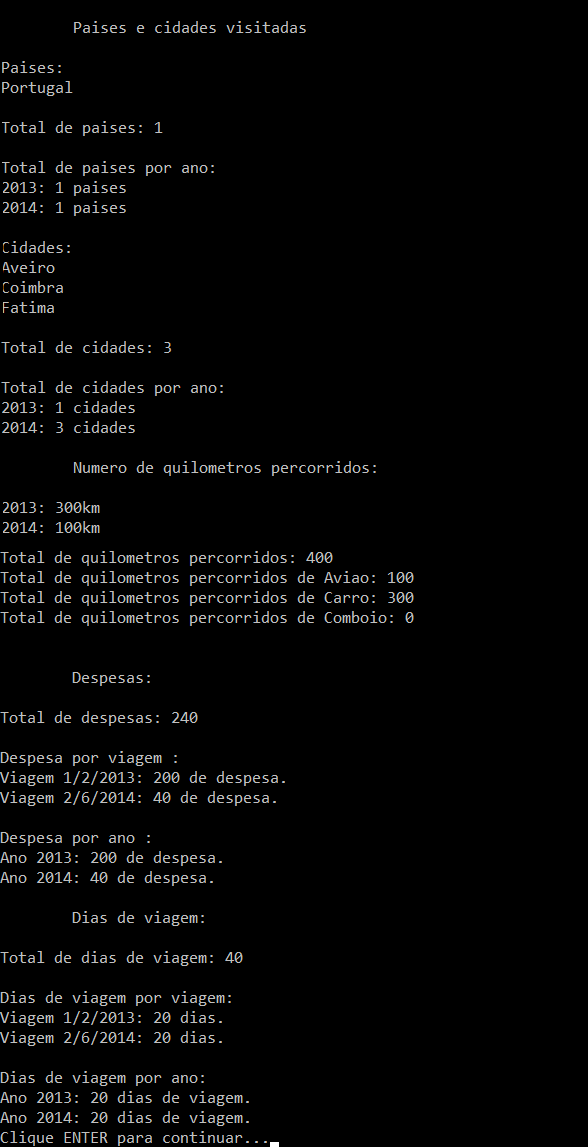


Figura 12a – Parte do exemplo do que poderia obter após escolhar a opção 2 (continua na Figura 12b)

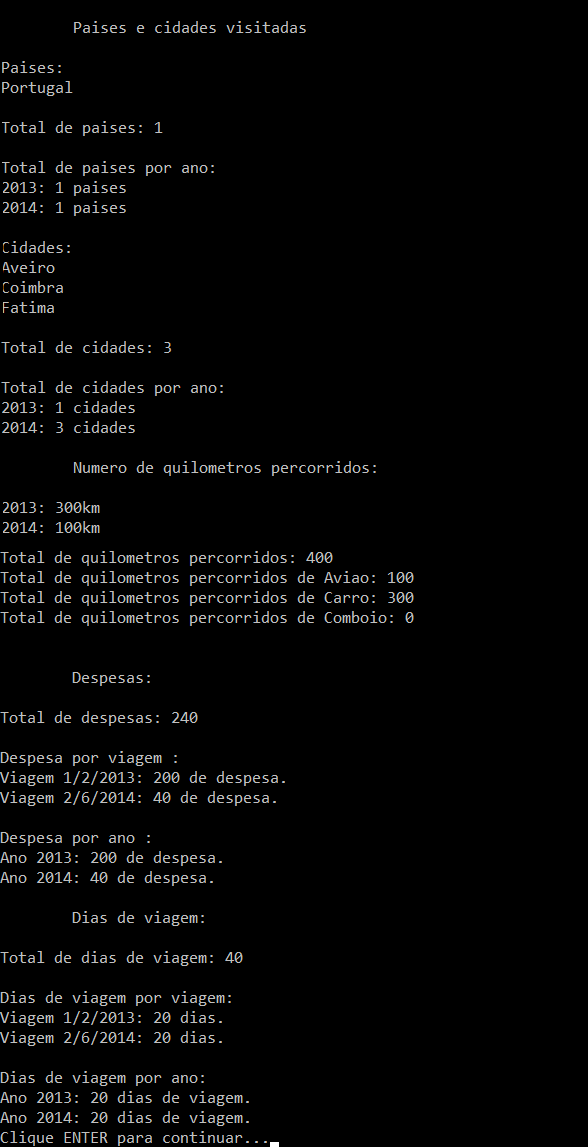


Figura 12b - Continuação do exemplo do que poderia obter após escolhar a opção 2

Caso o utilizador selecione a opção 3, irá mostrar informações percentuais relacionadas às viagens até ao momento realizadas. Como exemplo: Figura 13a e 13b.



Figura 13a - Parte do exemplo do que poderia obter após escolhar a opção 3 (continua na Figura 13b)



Figura 13b - Continuação do exemplo do que poderia obter após escolhar a opção 3

Caso o utilizador selecione a opção 4,



## 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*.

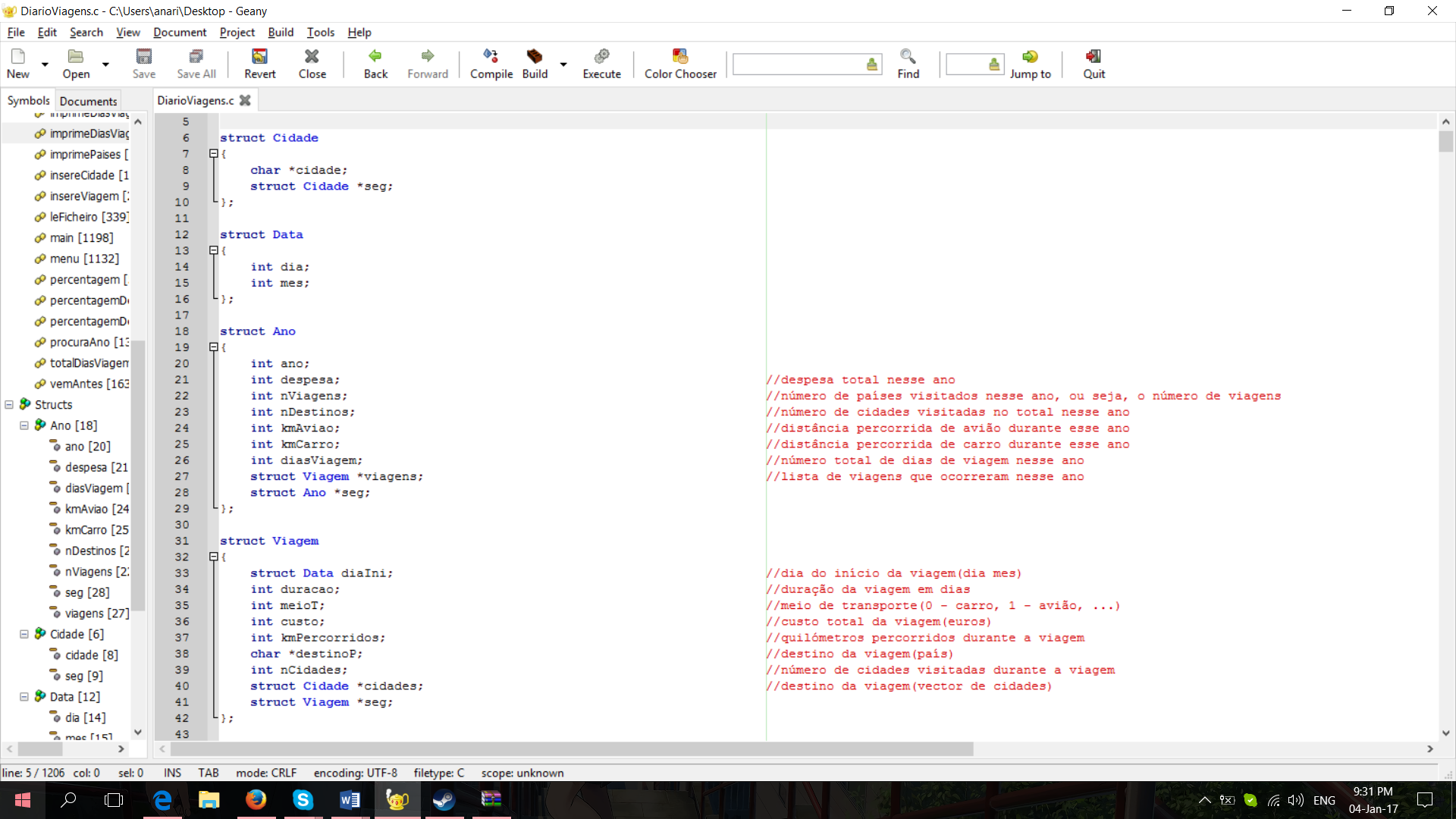


Figura 2 - Definição das estruturas (diarioviagens.h)

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;
* **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 inteiro 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.

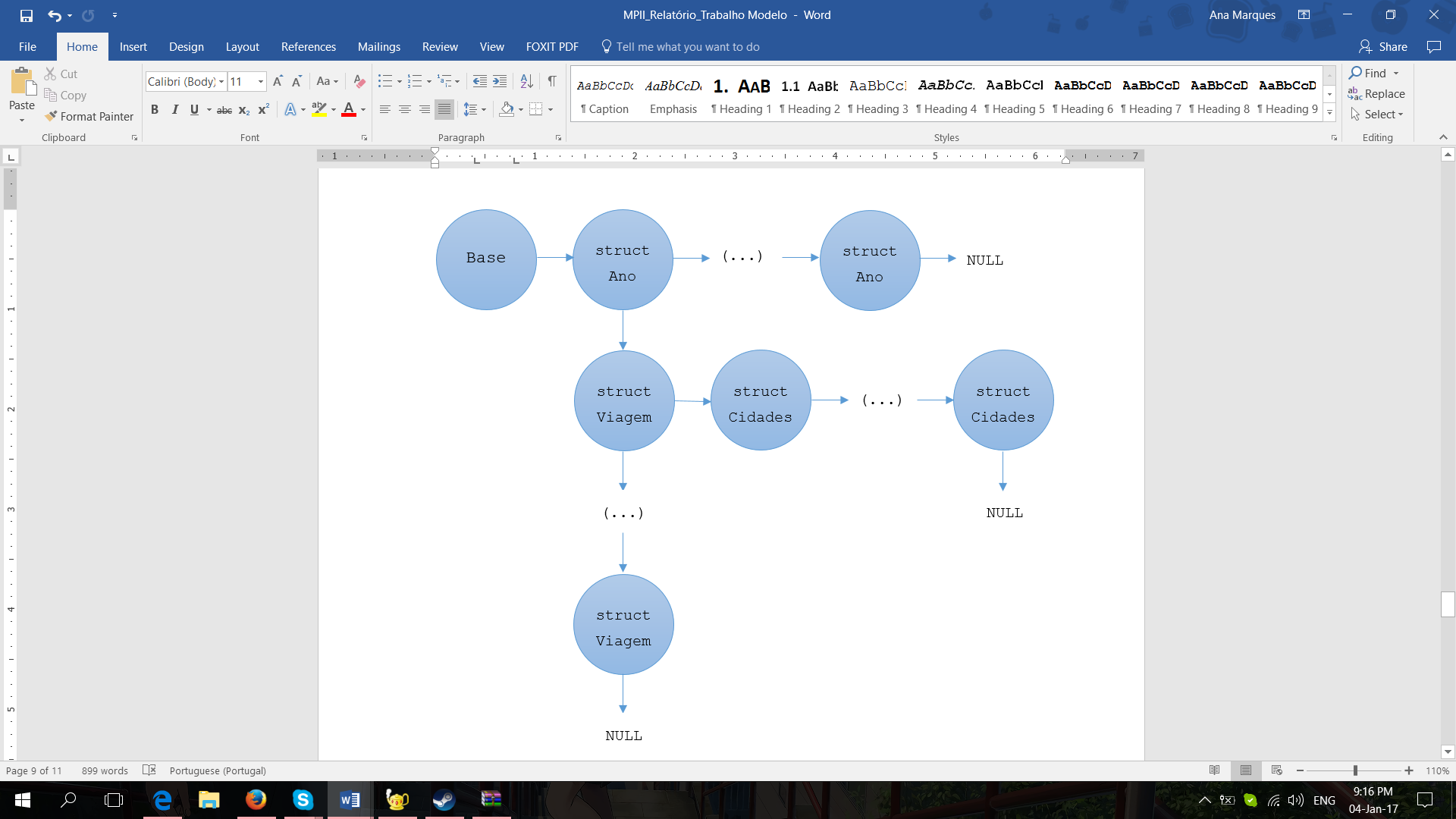


Figura 3 – Representação gráfica das listas

## 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, são lidos, letra a letra, os vários segmentos da linha, delimitados por ‘;’ até alcançar ‘\n’ ou EOF.
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 até ser válida. Quando esta for válida escreve-a no ficheiro com o formato pertendido para leitura (ou seja, com ‘;’ no final de 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 não 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 com formato específico, este é um ‘;’ entre cada entrada e por fim um ‘\n’. Fecha o ficheiro para escrita utiliza a função *atualizaDados* para como a própria dá a entender atualizar as dados contidos no calendário e na estrutura Ano a que pertence. E por fim chama a função *insereViagem* que, como vai ser explicado a seguir a insere na lista.
5. *insereViagem*: coloca a estrutura criada na função criaViagem no ano anteriormente criado ou encontrado.
6. *consultaInformacao*: permite consulta toda a 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).
7. *consultaPercentagens*: permite consultar informação percentual a partir na informção da função anterior.
8. *criaRelatorio*: coloca numa lista dinâmica todos os países que foram visitados, todos os custos e todas as cidades visitadas em cada país, ambos sem repetição. Calcula a media do número de cidades visitadas por país e da despesa por país. Verifica quais dos país estão acima da média de ambas as categorias e imprime-os, dependendo da escolha do utilizador poderá ser imprimido no ecrã ou num ficheiro denominado *Relatorio\_”nome”.txt*, sendo este “nome” escolhido pelo utilizador.

# Apresentação de uma sessão de utilização

Esta aplicação tem a capacidade de receber informação de duas formas distintas. Através da opção **1** menu do programa como já foi explicado no capítulo **2.3** ou inserindo directamente a informação no ficheiro de texto *DiarioViagens.txt* no seguinte formato:

dia;mês;ano;duração\_da\_viagem;meio\_de\_transporte;quilómetros\_percorridos;custo;destino\_da

viagem(país);número\_de\_cidades\_visitadas;nomes\_das\_cidades\_intercalados\_por\_‘;’

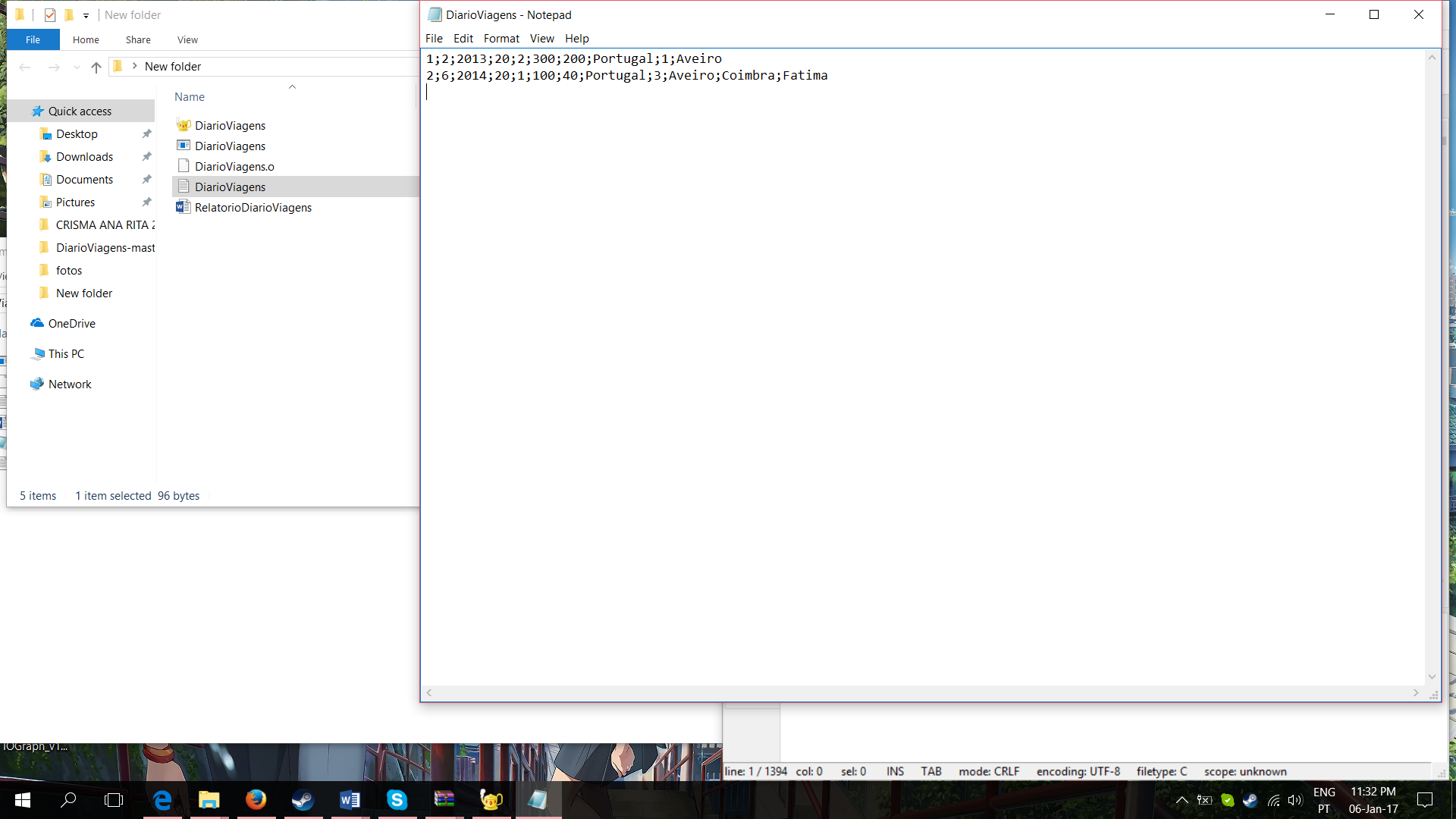


Figura 14 – Exemplo de inserção de viagens através do ficheiro “DiarioViagens.txt”

É também possível consultar informação sobre as viagens já existentes através das opções **2** e **3**.

A criação de relatórios pode ser realizada também em dois formatos. Directamente na consola ou, alternativamente, um relatório pode ser descarregado para um ficheiro de texto.

# Conclusões

Começámos este trabalho por planear as estruturas e por escrever as funções correspondentes. Apareceram alguns erros na inserção de nós nas listas que foram eventualmente corrigidos. O trabalho, em geral, foi realizado em ritmo constante e alternando entre escrever funções e corrigir seus erros. O resultado final ficou como esperado com exceção da ausência da ordenação dos nomes por ordem alfabética nos relatórios.

# Bibliografia

**Informação impressa:**

DAMAS, L., *Linguagem C*, 24ª ed., FCA, 2015.

**Fontes de informação na Internet:**

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].