 <small>ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO</small>	Tipo de Prova Exame Época de Recurso	Ano letivo 2020/2021	Data 26-02-2021
	Curso Lic. em Engenharia Informática Lic. em Segurança Informática e Redes de Computadores	Hora 10:00	
	Unidade Curricular Fundamentos de Programação	Duração 2h00	

#### Observações

- Se quiser desistir, assinale essa opção no formulário de desistência presente no Moodle.
- Após a realização da prova, submeta os ficheiros criados no Moodle (formato .zip) com o seguinte formato: <NUM\_ALUNO>.zip, ex.: 80000000.zip
- Não deve sair da sessão sem antes garantir que o docente consegue abrir os ficheiros submetidos no Moodle.
- Os estudantes devem marcar as presenças no Moodle.

### Parte 1 (1 hora)

Uma fábrica de calçado de alta sapataria, sediada em Felgueiras, criou uma linha especializada de produtos para senhora. Esta linha de produção é constituída por **MAX\_PRODUTOS** produtos que são armazenados em **MAX\_ARMAZENS** armazéns. A fábrica pretende desenvolver um programa para a gestão de stocks desta nova linha de produção. Os valores de stock dos produtos são armazenados numa matriz denominada **stocks**, como se exemplifica na imagem seguinte, em que **MAX\_PRODUTOS** e **MAX\_ARMAZENS** são iguais a 5 e 4, respetivamente.

<b>stocks</b>	Produto 1	Produto 2	Produto 3	Produto 4	Produto 5
Armazém 1	200	75	0	1	76
Armazém 2	350	58	3	222	21
Armazém 3	100	35	41	346	254
Armazém 4	123	289	600	324	2123

1. Implemente a função **inserirDados** que recebe a matriz **stocks** e peça ao utilizador o valor de stock (em unidades) de todos os produtos nos vários armazéns. Assegure-se que o valor do stock armazenado é positivo.
2. Implemente a função **obterPorcentagem** que recebe a matriz **stocks** e imprima a percentagem de produtos em cada armazém em relação à totalidade de produtos existentes no stock da empresa.
3. Implemente a função **totalStockProduto** que recebe a matriz **stocks**, o índice (valor inteiro) de um produto e retorna o total de stock desse produto em todos os armazéns.
4. Implemente a função **verificaStockMinimo** que recebe a matriz **stocks** e um vetor chamado **stock\_minimo** (exemplo, representado abaixo). O vetor **stock\_minimo** deverá ser preenchido com valores pedidos ao utilizador, antes de invocar a função. A função deverá escrever na consola, para cada produto, quais os armazéns que têm um stock abaixo do stock mínimo.

Exemplo:

<b>stock_minimo</b>	Produto 1	Produto 2	Produto 3	Produto 4	Produto 5
	150	50	100	100	25

Saída no ecrã:

Produto 1

Abaixo do stock mínimo em: Armazém 3, Armazém 4

Produto 2

Abaixo do stock mínimo em: Armazém 3

Produto 3


Abaixo do stock mínimo em: Armazém 1, Armazém 2, Armazém 3

Produto 4

Abaixo do stock mínimo em: Armazém 1

Produto 5

Abaixo do stock mínimo em: Armazém 2

 <small>ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO</small>	Tipo de Prova Exame Época de Recurso	Ano letivo 2020/2021	Data 26-02-2021
	Curso Lic. em Engenharia Informática Lic. em Segurança Informática e Redes de Computadores	Hora 10:00	
	Unidade Curricular Fundamentos de Programação	Duração 2h00	

## Parte 2 (1 hora)

Uma empresa de fiscalização e cobrança de estacionamento precisa de um software que permita acompanhar as infrações reportadas pelos seus funcionários. Cada infração é caracterizada por (entre parêntesis, apresentam-se os valores ou intervalo de valores admissíveis):

- Infração de estacionamento:
  - Identificador (valor único e sequencial);
  - Data/hora da infração: dia (1-31), mês (1-12), ano (> 2020);
  - Matrícula (máximo de 12 caracteres);
  - Marca do veículo (máximo de 25 caracteres);
  - Modelo do veículo (máximo de 25 caracteres);
  - Zona estacionamento (máximo de 25 caracteres);
  - Valor da infração (em €);
  - Descrição (máximo de 255 caracteres);
  - Tipo de infração (NORMAL, GRAVE);
  - Nome do funcionário que reportou a infração (máximo de 25 caracteres).

1. Defina num módulo a estrutura de dados necessária e suficiente para armazenar e manipular as infrações da empresa.

Considerando a estrutura definida no ponto 1.

2. Crie a função **iniciarInfracoes** que permita carregar todas as infrações contidas num ficheiro binário denominado **infracoes.bin** para uma estrutura de memória alocada dinamicamente. Esta função deverá ser invocada no início do programa.
3. Crie a função **inserirNovaInfracao** que permita inserir uma nova infração, validando os dados inseridos. Anexe, no ficheiro **infracoes.bin**, os dados da infração inserida.
4. Crie a função **obterTotalInfracoesData** que peça ao utilizador uma data (dia, mês e ano) válida e retorne o valor total das infrações reportadas nessa data. Escreva, em linhas consecutivas de um ficheiro de texto denominado **infracoes\_Totais.txt**, todos os valores de infração registados nesse dia.
5. No programa principal (**main.c**) crie um menu para as funcionalidades desenvolvidas anteriormente. Deverá permitir que o utilizador execute essas funcionalidades até que a opção para **Sair do programa** seja escolhida.