

Divisão das tarefas de implementação pelos membros do Grupo 23

As funções responsáveis por concretizar todas as funcionalidades exigidas no enunciado do projeto, juntamente com a respetiva autoria e alínea correspondente, estão discriminadas abaixo. As explicações detalhadas de cada função estão documentadas nos arquivos do código, conforme foi solicitado.

2.1, 2.2 e 3.5 – João Calheta (2103423)

```
int aleatorio_capacidade();
int aleatorio_numero_seccoes();
int aleatorio_numero_serie();
int aleatorio_preco();
int aleatorio_probabilidade_venda();
void categorias(string categoria[], int numero_categorias);
int contador_linhas_categorias();
void marcas(string marca[], int numero_marcas);
int contador_linhas_marcas();
peca * inicializacao_pecas(int numero_pecas, seccao * seccoes, int numero_seccoes, int
numeros_serie[], int& numeros_serie_gerados, int numeros_serie_disponiveis);
seccao * inicializacao_seccoes(int numero_seccoes);
void gravar_estado_armazem(seccao* seccoes, peca* lista_chegada, int numero_seccoes,
float total_faturacao, int dia,
                                int primeiro_livre, int numeros_serie_gerados, int
capacidade_espera, bool & estado_gravacao,
                                int primeiro_livre_secundario, peca * lista_de_espera);
```

2.3, 3.1, 3.2 e 3.6 – Filipe Gonçalves (2141023)

```
void inicializacao_lista_chegada(peca *& lista_chegada, int capacidade_maxima, int &
primeiro_livre, const peca & nova_peca, peca *& lista_de_espera, int
&capacidade_espera, int & primeiro_livre_secundario);
void venda_pecas(seccao* seccoes, int numero_seccoes, int & total_faturacao_armazem);
void transferencia_pecas(peca * lista_chegada, int & primeiro_livre, seccao * seccoes,
int numero_seccoes);
void venda_manual(seccao* seccoes, int numero_seccoes, peca* lista_chegada, int&
primeiro_livre, int &total_faturacao);
void adicionar_promocao(seccao* seccoes, int numero_seccoes, int dia_atual);
void aplicar_promocao(seccao* seccoes, int numero_seccoes, int dia_atual);
void carregar_estado_armazem(seccao* &seccoes, peca* &lista_chegada, int
&numero_seccoes, float &total_faturacao, int &dia,
                                int &primeiro_livre, int & numeros_serie_gerados, int
&capacidade_espera, bool & estado_gravacao,
                                int &primeiro_livre_secundario, peca * &lista_de_espera);
```

3.3 e 3.4 – Maryna Mramornova (2163623)

```
bool categoria_existe(string categoria_nova, int numero_categorias);
void adicionar_categoria(string categoria_nova);
void alterar_categoria(seccao * seccoes, string categoria_nova, int numero_seccoes);
void adicionar_seccao(seccao*&seccoes, int &numero_seccoes);
```

3.7 e 4 – Cláudio Fernandes 2186622

```
void armazenar_alfabetica(seccao* seccoes, peca * lista_chegada, int &primeiro_livre, int
&numero_seccoes);
void armazenar_preco(seccao* seccoes, peca * lista_chegada, int &primeiro_livre, int
&numero_seccoes);
```

É importante destacar que ao longo da construção do código, existiram funções que foram corroboradas e retificadas por alguns elementos do projeto aquando da junção das partes e conforme foi-se construindo o código. Isso abrange desde a adaptação de códigos preexistentes para integrar as funcionalidades designadas até à reestruturação da componente principal do programa. Todos contribuíram para o trabalho final entregue.