Projeto Prático – Versão C

Gestão de Parque de Estacionamento

Introdução

Uma empresa de Consultadoria e Produção de Software "AllManagement" pretende desenvolver uma aplicação para gerir parques de estacionamento. Dado que a aplicação se pretende instalar em qualquer parque (para ser assim o mais rentável possível), deve permitir a sua customização, de forma a poder ser adaptada a qualquer configuração do parque, considerando máximos estudados, como sendo razoáveis para aplicação na maioria das situações possíveis.

Assim, o parque poderá ter um máximo de 5 pisos, cada piso podendo ser configurável em termos de número de filas e lugares por fila, com um máximo admissível de 26 Filas (de A a Z) e 50 lugares por fila (de 1 a 50). Atenção que, para permitir a indisponibilização de lugares do parque (por motivo de obras, dano no piso do lugar de estacionamento ou de outro equipamento, ou, ainda, qualquer outro motivo), o lugar pode ser "marcado" como não disponível e, posteriormente, a situação deve poder ser revertida.

O parque de estacionamento é pago, sendo aplicado o seguinte tarifário:

a) 8:00 às 21:59 - T1 €/hora (valor calculado em períodos de 15 min);
b) 22:00 às 7:79 - T2 €/hora (valor calculado em períodos de 15 min);
c) dia completo (o período de estacionamento não poderá mudar de dia) - T3 €;
d) vários dias - T4 €/dia, sendo o número de dias a considerar o número de dias em que esteve no parque, independentemente da hora de entrada e saída.

Obs.

1) Para os tarifários das alíneas a) e b) o valor a pagar é calculado em períodos de 15 min, ou seja: 1) se a viatura entrar às 8:10, é considerado, para efeitos de cálculo que entrou às 8:15. Quanto à hora de saída, é considerado o período de 15 min seguinte, ou seja, se a viatura sair às 9:55, considera-se para efeitos de cálculo do valor a pagar, que a saída ocorreu às 10H00;
2) Se o valor a pagar ultrapassar o valor do dia completo, mas o período do estacionamento não tiver mudado de dia, deve ser considerado o valor de um dia completo (aplicando-se a tarifa da alínea c);
3) Se a viatura permanecer por um período de vários dias, o número de dias a considerar deve ser a soma dos dias completos e incompletos, só sendo considerada esta tarifa se o número de passagens de dia forem >=2;
4) Considere que T1=0.6€; T2=0.3€; T3=8€ e T4=6€;
5) Estes valores devem poder ser facilmente alterados no futuro.

A informação relativa a este tarifário é fornecida sob a forma de um fich. .csv, cujo formato se apresenta na Figura 1.

Aquando da chegada de um automóvel ao parque e, após verificar que existem lugares livres, é atribuído um lugar ao automóvel, sendo o condutor informado do lugar que lhe foi atribuído.

Todas as entradas no parque devem ser imediatamente registadas num dado estruturado / estrutura de dados apropriada, cuja conversão em ficheiro texto se apresenta na Figura 2.

Como se pode perceber, para cada entrada, deve registar-se um número sequencial de entrada (gerado automaticamente), a matrícula da viatura, o ano, o mês, o dia e a hora/minuto de entrada e o lugar ocupado. Quando o veículo sai, o registo correspondente ao estacionamento deve ser atualizado, registando-se o ano, mês, o dia e a hora/minuto de saída e o valor pago, devendo ser libertado o lugar ocupado, ficando disponível para estacionamento.

Para facilitar a vida aos automobilistas, são apresentados num placard existente em cada entrada do parque de estacionamento, o número de lugares disponíveis em cada piso, permitindo-lhes assim, optarem, de imediato por um dado piso do parque.

Tarifario.txt

<TpTarifa>\t<CodTarifa>\t<HoraInf>\t<HoraSup>\t<ValorHora>

H	CT1	08:00	22:00	0.60€
H	CT2	22:00	08:00	0.30€
D	CT3	00:00	00:00	8.00€
D	CT4	00:00	00:00	6.00€

Nota: 1) O atributo TpTarifa significa se a tarifa é calculada em função da data/hora de entrada e saída (H) ou do número de dias de estacionamento do veículo (D). Não esquecer que numa situação especial, um estacionamento cujo valor a pagar seria calculado em função da tarifa horária, vem a sê-lo com base numa tarifa diária.

Figura 1 - Formato e exemplo de linhas com os dados correspondentes a registos do tarifário

Estacionamentos.txt (adaptado do ficheiro binário para ficheiro de texto)

<numE>\t<matricula\t<anoE>\<mesE>\t<diaE>\t<horaE>\t<minE>\t<lugar>\t<anoS>\t<mesS>\t<diaS>\t<horaS>\t<minS>\t<valorPago>\<Obs. >

1	66-FK-23	2024	10	12	19	13	1A21	2024	10	12	22	38	1.88
2	3SAM123	2024	10	12	19	17	2D13	2024	10	17	23	55	36.00
3	I32.003	2024	10	12	19	17	1C22	2024	10	13	07	44	4.43
....													
11145	AA-67-FH	2024	11	30	15	44	3D12	2024	12	01	03	23	5.40
11146	AF-23-FH	2024	11	30	21	40	1A10	2024	12	01	04	23	8.00

Nota: 1) O código de cada lugar de estacionamento é constituído por 4 caracteres: o primeiro caracter indica o número do piso, sendo um dígito entre 1 e o número máximo de pisos a considerar; o segundo caracter é uma letra maiúscula que indica a fila (entre 'A' e 'Z' para as 26 filas) e os dois próximos caracteres referem-se ao lugar na fila (entre 01 e 50). Exemplo: 3C05 = Piso 3, Fila C, Lugar 05;
2) Qualquer novo campo que se revele necessário poderá ser acrescentado ao ficheiro.

Figura 2 – Formato e exemplo de linhas com os dados correspondentes a registos de estacionamento

Funcionalidades:

Atendendo à descrição geral da aplicação realizada na Introdução, o projecto a implementar deverá responder às seguintes funcionalidades:

1. Na fase de "instalação" da aplicação, ativável através de um qualquer mecanismo, que que considerarem mais adequado, onde:

   a) atendendo à topologia do parque de estacionamento, deverá ser especificado qual o número de andares, fila e lugares por fila do parque de estacionamento; o número máximo de registos que será possível guardar em cada um dos dados estruturados que vão guardar em memoria a informação que a aplicação vai tratar, exceto se os dados estruturados forem tratados dinamicamente. Neste caso, a existência de um máximo de registos a tratar e se há um número inicial de registos e depois, um valor para cada incremento, quando o número de registos disponíveis no momento for atingido, é uma opção do grupo. Poderá haver ainda o outra informação de inicialização a especificar, se a arquitectura do sistema concebida o exigir. Toda esta informação recolhida deverá,

obviamente, ser guardada em memória e assegurada a sua persistência (em ficheiro(s)), da forma que entenderem;

b) ainda durante a fase de inicialização da aplicação, deve ser lido o ficheiro fornecido, Estacionamentos.txt, onde cada linha contém os dados correspondentes a um registo de estacionamento, que deve ser inserido no dado estruturado conveniente, já com o valor pago calculado (devendo, para isso, ler o fich. Tarifas.txt e guardar os seus dados em memória), se os veículos já tiverem saído do parque de estacionamento. Não se esqueçam de assegurar a respectiva persistência dos dados existentes em memória, (p. ex., executando a opção 10 da aplicação);

Obs. Atenção que algumas das linhas do ficheiro (correspondentes a registos de estacionamento), ainda não têm a hora de saída, pelo que se considerará, para esses casos, que os veículos ainda estão no parque e, assim, devem ser considerados em termos da ocupação do parque de estacionamento;

c) deve ser também levado em consideração que haverá outros dados importantes para o funcionamento do sistema (p. ex., numeração dos registos de estacionamentos e outros) que deverão ser guardados em memória.

No final desta fase, deve ser assegurada a persistência dos dados existentes em memória, executando, p. ex., a funcionalidade 10 da aplicação.

2. Aquando da utilização "normal", ao arrancar a aplicação, ler os ficheiros de dados gravados na utilização anterior da aplicação ou na inicialização, carregando-os para dados simples ou estruturados, utilizando estruturas de dados convenientes, não esquecendo também de ler eventuais outros ficheiros que tenham sido acrescentados para guardar outros dados importantes para o funcionamento da aplicação (parâmetros do sistema, tais como: número de pisos, filas e lugares por fila), numeração de documentos e outros;

3. Criar uma estrutura de menus adequada, devendo, em qualquer deles (no caso de haver mais do que um), estar sempre visível, no canto superior direito, o número de lugares disponíveis em cada piso;

4. a) Permitir o registo da entrada de cada veículo, atribuindo o lugar e mostrando-o ao condutor, não devendo ser esquecido que, a cada entrada, deve ser atualizado o estado do parque de estacionamento;

   b) Deve ser possível também a consulta, eliminação ou alteração de um qq. registo de estacionamento, especificando-se o respectivo número de entrada, permitindo assim, p. ex., em casos onde o lugar atribuído esteja, indevidamente ocupado total ou parcialmente, inviabilizando o estacionamento, que o registo do estacionamento seja alterado para o novo lugar atribuído (não deve esquecer-se a atualização também do estado do parque de estacionamento, para reflectir a correção havida);

5. Quanto à atribuição do lugar, ele deve ser efectuado, atendendo àquele que estiver mais perto ou, se o grupo pretender usar um algoritmo melhorado, permitir que o utilizador possa seleccionar entre: a) o lugar mais perto da entrada (igual à versão base); b) o lugar mais perto da entrada (igual à versão base), mas de fácil estacionamento (onde estiverem um mínimo de 2 lugares desocupados; c) um lugar mais perto da saída pedonal (supõe-se esta será junto ao último lugar e na fila do meio); d) que esteja num lugar mais perto da saída pedonal, mas de fácil estacionamento (onde estiverem um mínimo de 2 lugares desocupados. Se não houver nenhum lugar vago que permita estacionar o veículo atendendo à opção pretendida pelo utilizador, deve procurar-se um lugar que a possa satisfazer num dos pisos mais próximos. Para efectuar este cálculo e efectuar a atribuição:

   a. deve considerar-se que o lugar mais perto da entrada, será o A01 em qualquer piso;

   b. que a distância a percorrer é 2 vezes maior por cada lugar de distância da entrada do que por cada fila.

6. Registar a saída do veículo, efetuando o cálculos do valor a pagar e atualizações consideradas necessárias, devendo poder-se efetuar a sua alteração ou anulação;

7. a) Poder especificar lugares não disponíveis, indicando-se e registando-se o motivo (um caracter), que poderá ser: i-dado estar em condições inadequadas; o-estar a ser objecto

de obras; e-estar reservado (no caso de haver eventos e ser necessário reservar lugares de estacionamento para o efeito), ou, ainda, o-outros motivos;

b) Poder reverter-se esta situação, alterando novamente a informação correspondente à situação que implicava a não disponibilidade do lugar.

8. Mostrar um mapa de ocupação de um piso do parque de estacionamento, em qualquer momento, de forma o mais explícita possível (um lugar ocupado deverá ser assinalado por um X, um lugar livre por um "-" e um lugar indisponível, mostrando o respectivo motivo);
9. Listar todos os veículos que saíram num determinado dia (a especificar pelo utilizador, AAAA/MM/DD) e do número total de veículos e respectivo valor total pago nesse dia, apresentando uma linha final (separada das linhas correspondentes à informação relativa aos estacionamento havidos com uma linha com "-----"), devendo a listagem dos estacionamentos ser apresentada por ordem decrescente do valor pago por cada estacionamento;
10. Assegurar a persistência dos dados existentes em memória, utilizando para o efeito ficheiros binários, sempre que o utilizador assim o entender e, obrigatoriamente, ao sair da aplicação, não devendo esquecer-se a atualização de outros ficheiros que tenham sido igualmente criados e utilizados para guardar outra informação importante e que terá sido carregada em memória, aquando do arranque da aplicação (neste caso, poderão ser guardados em fich. tipo txt), pois os dados que contêm poderão ser conhecidos com facilidade e, eventualmente, alterados, se for necessário.

Nas diversas listagens a implementar, dada a potencial elevada quantidade de informação, deve implementar um mecanismo de avanço página a página e deve ser oferecida a possibilidade de gravar a listagem em ficheiro de texto .txt e .csv, para posterior consulta num simples editor de texto e no Excel o aplicativo similar.

Extras:

E1: Permitir que a listagem criada na funcionalidade 9 seja apresentada sob, opção a) a forma de um gráfico de barras ou, opção b) uma tabela de 2 entradas (tabela dinâmica), mas, no 1.º caso (opção a)), para valores devem considerar-se o número total de veículos que saíram num determinado dia, em cada hora do dia e o valor respetivo (podendo ser especificado o intervalo de horas pretendido); no 2.º caso (opção b)), quanto aos valores, devem considerar-se os mesmos, mas poderá ser seleccionado um intervalo de dias de um intervalo de meses de um dado ano. Um exemplo de cada um dos casos é apresentado na Figura 3 (ex.: em abcissas, cada dia e, em ordenadas, as horas; na interceção das 2 coordenadas, será apresentado o valor observado do número de veículos que saíram e o valor total pago). No entanto, nesta listagem, só deve ter-se em consideração o número de veículos e o respectivo valor pago em cada dia/hora.

Dados Base: Respectivo Gráfico Gerado:

Filtros:
Dia: 05/01/2024
Período: 00:00 - 05:59

Hora	N.º Saídas	Valor
00:00 - 00:59	560	432,20 €
01:00 - 01:59	345	234,50 €
02:00 - 02:59	234	222,20 €
03:00 - 03:59	222	323,30 €
04:00 - 04:59	123	234,20 €
04:00 - 04:59	89	234,20 €
04:00 - 04:59	87	123,40 €
05:00-05:59	98	342,30 €
00:60 - 06:59	123	134,50 €
06:00 - 06:59	234	345,60 €
07:00 - 07:59	543	678,80 €
08:00 - 08:59	678	956,70 €
09:00 - 09:59	876	1 233,40 €
10:00 - 10:59	987	1 567,80 €
11:00 - 11:59	1234	2 132,50 €

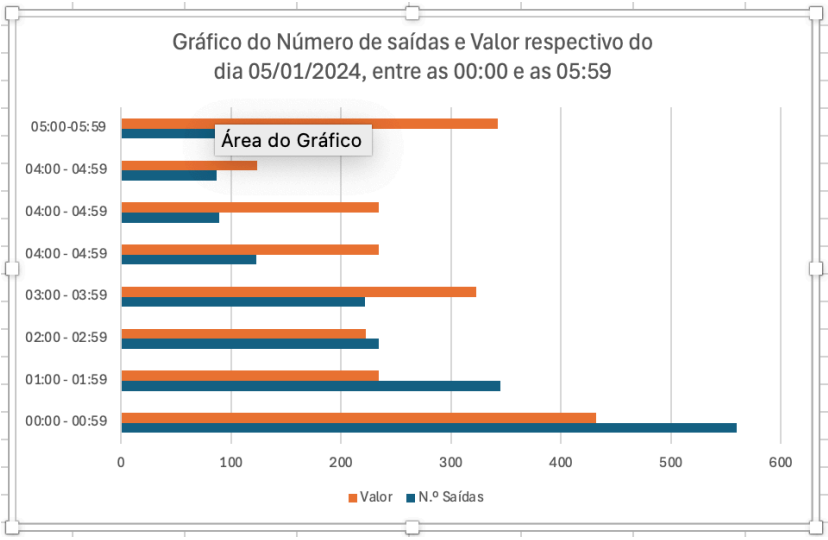

Figura 3 – Exemplo de Dados Base e respectivo Gráfico de Barras referidos no exercício A da parte dos Extras (à esquerda e direita, respectivamente)

Dados Base:

Filtros:
Ano/Mês: 2024/01 - 2024/04
Dia: 1 - 3

Mês	Dia	N.º Saídas	Valor((€)
Jan	1	5600	3 400,30
Jan	2	6533	3 565,00
Jan	3	6788	4 321,00
Fev	1	7690	5 633,00
Fev	2	4355	5 437,40
Fev	3	5433	3 456,00
Mar	1	3422	3 456,60
Mar	2	4322	5 675,00
Mar	3	6754	3 452,40

Respectiva Tabela Dinâmica Gerada:

Filtros:	Dias: 1 -3
	Mês: Jan; Fev

	Dias							
	1		2		3		Totais	
	Num. Saídas	Valor(€)	Num. Saídas	Valor(€)	Num. Saídas	Valor(€)	Num. Saídas	Valor(€)
Jan	5600	3 400,30	6533	3 565,00	6788	4 321,00	18921	11 286,30
Fev	7690	5 633,00	4355	3 244,00	5433	3 456,00	17478	12 333,00
Totais	13290	9 033,30	10888	6 809,00	12221	7 777,00	36399	23 619,30

Figura 4 – Exemplo de Dados Base e respectiva Tabela Dinâmica referidos no exercício A da parte dos Extras (à esquerda e direita funcionalidade E1, respectivamente)

E2: Gerar ficheiro texto tipo CSV para permitir exportar dados para poderem ser lidos por outras aplicações, onde o separador possa ser especificado pelo utilizador (normalmente, será "," ou ";"), dos estacionamentos efetuados, entre datas, com linha de cabeçalho que considerarem adequada (para o destinatário saber a informação de cada campo das linhas de dados) e, naturalmente, com um nome igualmente sugestivo.

Desenvolvimento Faseado

O desenvolvimento deste projeto deverá ser feito de uma forma faseada, devendo os alunos garantir que todas as funcionalidades codificadas até ao momento estão a funcionar corretamente.

"É preferível apresentar um programa que implementa poucas funcionalidades, mas que funcionam corretamente, do que um programa totalmente desenvolvido, mas que não faz nada."

Os estudantes deverão garantir a robustez da aplicação, verificando todos os casos de erro (por exemplo, ao nível dos ficheiros).

Documentação

O código produzido pelos estudantes deverá ser comentado. Os comentários presentes no código devem explicitar e explicar o funcionamento da aplicação, assim como as decisões tomadas. Devem seguir as seguintes regras:

- A função deve ser precedida por um comentário que explique a funcionalidade implementada e o significado, se for caso disso, de cada um dos parâmetros e do valor devolvido;
- O código deve ser comentado sempre que realize alguma operação não óbvia;
- Os comentários devem ser claros, gramaticalmente corretos e usando frases simples;
- A declaração de todas as variáveis e constantes devem ser acompanhadas de um comentário com uma breve descrição;
- Cada bloco de código (seleção ou repetição) deve ser precedido de um breve comentário explicativo.

Como regra geral deve considerar que cada função deve cumprir as seguintes regras:

- Inicializar sempre as variáveis na sua declaração;
- Testar a validade dos parâmetros recebidos;
- Declarar constantes e nunca usar números no código;
- Evitar repetições de código - usar funções, ciclos, etc.;
- Evitar o uso de variáveis globais;
- Escrever código simples e claro.

Observações

- Os grupos poderão integrar um máximo de 4 alunos;
- Os estudantes/grupos podem conversar entre si para discutir possíveis soluções para algum problema que tenham, mas não devem partilhar código fonte, sob pena de ambos os trabalhos serem penalizados na classificação atribuída, se o "plágio" for apenas pontual, mas, relativamente a este assunto, atentem ao descrito no penúltimo item desta lista;
- O código pode ser dividido em módulos, para que possa ser desenvolvido em paralelo (havendo uma partilha, da responsabilidade do grupo ou, preferencialmente, de um "gestor de projecto" que terá necessariamente, vários outros papeis, da maior importância). De qualquer modo, todos os elementos do grupo devem conhecer todo o trabalho desenvolvido, como se cada um tivesse realizado todas as fases que permitiram criar a aplicação e todos os demais componentes complementares);
- A aplicação pode ser estruturada sob a forma de um Projecto, devendo ser convenientemente distribuída por diversos ficheiros, cada um com um conteúdo com uma dada caracterização. P. ex. um fich. "main.c"; outro "func.c" com as funções, ou vários ficheiros de nomes expressivos com as funções divididas por grupos de funcionalidades (com algo de comum); ficheiros tipo "header" (xxxx.h) com structs ou outros elementos integrantes do programa que possam ser incluídos em vários ficheiros do projecto, etc. Esta estruturação, se bem realizada, será, naturalmente, valorizada;
- Deverão ser criadas as estruturas de dados necessárias;
- Deverão usar os ficheiros que forem fornecidos para carregar em memória os dados estruturados criados, cujos tipos devem ser os apropriados, possivelmente do tipo de cada uma das estruturas de dados criadas. Não esquecer que, quando especificado, a importação dos dados dos ficheiros, pode implicar algum processamento adicional (p. ex., o cálculo de

um ou mais valores ou a atualização do estado do sistema (dada a necessária atualização de outros dados estruturados em memória);

- Todo o processamento de dados, durante a utilização da aplicação (inserção de novos registos, alterações, consultas, e outras operações) deve ser feito utilizando os dados estruturados existentes em memória;
- A validação de dados será valorizada, quer na operação de leitura dos ficheiros para memória, quer relativamente aos dados especificados pelo utilizador;
- O trabalho deve ser implementado em linguagem C standard e deve apenas usar os conhecimentos lecionados nas aulas de Algoritmos e Programação;
- A funcionalidade do programa é, nesta unidade curricular, muito mais importante que os aspetos estéticos. Será valorizada a clareza e simplicidade do código. Deverá assim usar funções e estruturar o programa de modo a torná-lo simples, bem estruturado, e sem repetições desnecessárias de código. Será ainda valorizada a conveniente indentação do código e a inclusão de comentários, sempre que tal for interessante do ponto de vista da legibilidade (aliás, este tema foi já realçado na secção "Documentação";
- Possíveis melhorias incluídas no programa, devidamente fundamentadas e enquadradas com o programa desta unidade curricular, serão consideradas e valorizadas;
- Para os dados estruturados e sua manipulação, a) podem utilizar estruturas de tamanho fixo (ex. vectores declarados e criados, usando o operador [] e, sendo, depois, os elementos acedidos igualmente, através do mesmo operador); b) podem utilizar memória dinâmica e, depois, ponteiros e aritmética de ponteiros, para aceder ao cada elemento. Como a utilização da opção b) é, do ponto de vista da utilização de recursos mais eficiente, naturalmente, a aplicação que a utilizar será também alvo de uma valorização superior.

  Obs. Quanto à opção a seleccionar para esta característica da aplicação, é importante não esquecer a "*quote*" incluída na seção "Desenvolvimento Faseado" e que se passa a transcrever: "*É preferível apresentar um programa que implementa poucas funcionalidades, mas que funcionam corretamente, do que um programa totalmente desenvolvido, mas que não faz nada.*".

  Neste contexto, esta frase poderia ser adaptada para algo como "*É preferível apresentar um programa onde a declaração e manipulação de dados estruturados utilize o operador [], mas que funcionam corretamente, do que um programa totalmente desenvolvido utilizando ponteiros e aritmética de ponteiros para acesso aos registos, mas que não funciona.*".

  Ainda relativamente a esta opção de desenvolvimento, se não se sentirem "à vontade" com o uso de ponteiros, podem, numa 1.ª fase do desenvolvimento e teste da aplicação, usar o operador []. Usando esta solução, desenvolvem todas as funcionalidades (ou aquelas que conseguirem) e testam-nas convenientemente, corrigindo o que não estiver bem. Prosseguem com a realização das demais tarefas correspondentes aos elementos a entregar que devem integrar o MiniProjecto, observando as regras indicadas e providenciando a melhor qualidade de que sejam capazes. Finalmente, geram o respectivo pdf pronto a submete e guardam tudo numa pasta num disco e fazem uma cópia para a nuvem.

  Então, se ainda tiverem tempo disponível (significando isto que não irão prejudicar o estudo para outras UC a que ainda não obtiveram aprovação ou decidiram, para melhor se poderem preparar, submeter-se a avaliação em outra prova ou época), podem então converter a aplicação que têm pronta, adotando a solução a) para uma outra, onde tentarão usar a solução b). Não esquecer que, a cada transformação realizada, devem testar o que vai sendo desenvolvido e, corrigindo-o, se for o caso. No final, se a nova versão estiver também com as funcionalidades implementadas iguais à anterior, podem, então preparar o novo fich. .zip com a nova versão, não esquecendo de providenciar à alteração do relatório, para incluir a informação de que a aplicação foi agora implementada usando memória dinâmica e ponteiros e efectuar outras alterações, se algum código apresentado ou outro texto tenha

de ser agora adaptado, dada a adopção da nova solução. Sendo depois essa nova versão a ser submetida. Assim, não se correm riscos em embarcar numa solução "galáctica" e, depois, nem sequer conseguir fazer um avião de papel que consiga voar 5 metros...

- Deve ser elaborado um relatório que descreva o trabalho produzido e as funcionalidades implementadas, adotando o modelo e diretrizes publicadas no Moodle;
- Não serão admitidos plágios, mesmo que parciais - trabalhos copiados terão nota ZERO (tanto quem copiou como quem deixou copiar);
- Caso se revele necessário, poderão ser feitas atualizações ou alterações a este enunciado, pelo que os estudantes deverão estar atentos a esta eventualidade. Qualquer atualização ou alteração será devidamente anunciada no Moodle desta unidade curricular.

Entrega

O trabalho a entregar deve consistir num único ficheiro comprimido com o formato ZIP, devendo ser entregue até data a anunciar no link de submissão, tendo em atenção as seguintes indicações gerais:

1. O relatório solicitado deve estar em formato pdf;
2. A submissão deve ser realizada por um dos elementos do grupo, no link apropriado existente na página da unidade curricular no Moodle. Tal como estará indicado no texto que está junto ao link de submissão, nome do ficheiro deve ter o seguinte formato: EN_NomeApelido1_NomeApelido2_NomeApelido3_NomeApelido4 (com ordenação alfabética crescente) e deve conter todos os ficheiros: o código fonte da aplicação, bem como os ficheiros de dados necessários à execução, o relatório e um fich. txt com a identificação dos estudantes (nome e número). Um exemplo para o nome do ficheiro poderá ser: EN_AntonioCoelho_JoaquimFerreia_ManuelTomas.zip,
3. Em último recurso, o ficheiro com o trabalho pode ser enviado por e-mail, diretamente para jfialho@estgv.ipv.pt, caso por uma infeliz eventualidade, o Moodle não esteja a operacional.

Apresentação e Defesa

A defesa dos trabalhos, com a presença de **TODOS** os elementos do grupo de trabalho, é **obrigatória** em data, hora e local a indicar oportunamente.