Sistemas Operativos 2

2019/20

DEIS – Engenharia Informática

Trabalho Prático

As regras que regem o trabalho prático foram divulgadas nas primeiras aulas e encontram-se descritas na ficha de unidade curricular (*Nónio*), chamando-se a atenção para a sua leitura. É importante ter em conta que existem vários condicionantes e incógnitas ditadas pela epidemia COVID-19 e, por conseguinte, existem pormenores relativos a apresentação e defesa do trabalho que podem vir a ser definidos ou alterados posteriormente.

1 Descrição geral do tema – SOTáxi

Pretende-se um sistema de gestão e utilização informatizada de táxis que substitua a antiga central de rádio. O sistema proposto baseia-se na comunicação, por via digital, entre uma agência central, os carros e os passageiros. Numa aplicação real, a comunicação seria feita por rede. No âmbito deste trabalho, todos os intervenientes serão **processos** a correr **na mesma máquina** e não há comunicação em rede envolvida – apenas **comunicação entre processos**.

O sistema pretendido envolve diferentes processos com interfaces de consola e gráficas, e vários utilizadores (simulados por vários programas, sendo todos manipulados pelo mesmo utilizador do sistema operativo). Assim, este trabalho prático consistirá na implementação dos vários programas que, no seu conjunto, simulam este sistema de gestão e utilização de táxis.

Neste enunciado, "sistema" referir-se-á ao sistema de gestão e utilização de táxis. Sempre que for necessário referir o sistema operativo, serão usadas estas duas palavras explicitamente.

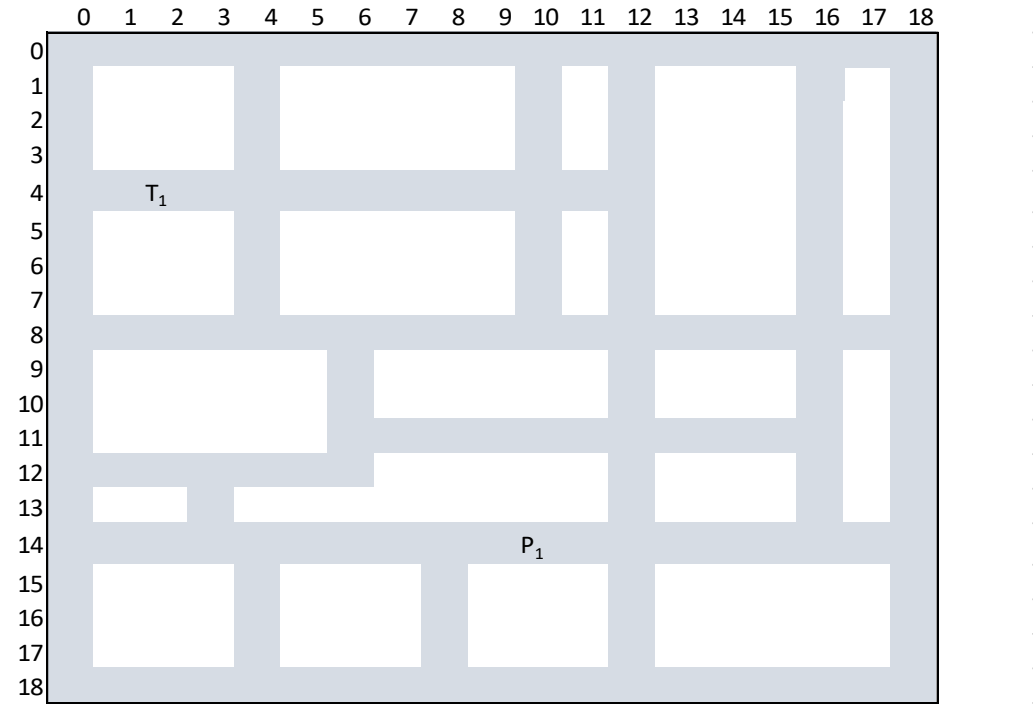
1.1 Utilizadores do sistema

O sistema tem os seguintes utilizadores: o administrador, os passageiros e os táxis.

Nenhum destes utilizadores acumula funções de outro tipo de utilizador. Ou seja, os conjuntos de ações desempenhadas por cada um são disjuntos entre si. Cada utilizador usará um (tipo de) programa, e cada um destes programas não é reutilizado por nenhum dos outros tipos de utilizadores.

1.2 Utilização geral do sistema

Existe uma central de táxis que efetua a atribuição de passageiros a táxis. Existe um mapa da cidade que é do conhecimento da central. Esse mapa é lido de um ficheiro de texto quando a central inicia. A cidade tem um aspeto muito regular, existindo ruas apenas na horizontal ou na vertical, e nenhuma rua tem ângulos: todas são segmentos de linhas retas. As localizações (partida, chegada e posição atual) são dadas por coordenadas definidas por linha e coluna. A Figura 1 apresenta um exemplo de cidade possível, com um táxi na posição (4,2) e um passageiro na posição (14,10). A dimensão mínima da cidade é de 50 linhas e 50 colunas. Existe uma aplicação *MapInfo* que serve apenas para visualizar o mapa da cidade, passageiros e táxis.



Legenda:

Segmento de recta a cizento -> Rua, Zona a branco -> Edifício
Há um Taxi T1 e um Passageiro a aguardar P1. No TP serão apresentados com imagens
Todas as ruas são de dois sentidos
Este mapa é apenas um exemplo (nem sequer cumpre o tamanho mínimo)
Sugestão de implementação: uma matriz bidimensional de estruturas

Figura 1 – Exemplo de mapa da cidade

A central de táxis é concretizada pela aplicação *CenTaxi*, da qual só pode haver uma instância a correr. Esta aplicação comunicará com as restantes através de vários mecanismos de comunicação inter-processo e será usada pelo administrador do sistema.

Movimento de passageiros

Um passageiro está inicialmente numa determinada posição inicial. Informa a central que deseja ser transportado para uma determinada posição destino. Esta informação é comunicada através da aplicação *ConPass*. Todos os passageiros partilham este programa, do qual deverá existir uma única instância a correr. A central irá informar o passageiro, através da aplicação *ConPass*, sobre qual o táxi que lhe foi atribuído, quando o táxi o apanhou, e quando chegou ao seu destino. Quando os passageiros chegam ao seu destino, deixam de existir no sistema (são esquecidos).

Movimento de táxis

Um táxi inicia o seu turno numa determinada posição e informa a central que entrou ao serviço nessa posição. O táxi deverá informar qual a sua matrícula, não devendo existir duas matrículas iguais no sistema ao mesmo tempo. Não existe qualquer necessidade de autenticação ou passwords. O táxi poderá estar em movimento ou parado. O táxi irá informar periodicamente a central acerca da sua posição, e informará também a central sobre o seu interesse em efetuar o transporte de um passageiro, de o ter recolhido, de o ter colocado no seu destino, e ainda da conclusão do serviço. A condução e operação do táxi é feita pelo seu condutor através da aplicação *ConTaxi*. Haverá uma aplicação destas a correr por cada táxi em funcionamento.

Sempre que existir um passageiro para ser transportado, a central notifica todos os táxis de que existe um passageiro à espera, ficando disponível durante algum tempo para aceitar manifestações de interesse por parte de táxis. Os táxis irão então consultar a informação que se encontra na central e informam se estão interessados nesse serviço. Findo o tempo, a central consulta as manifestações de interesse e escolhe de forma aleatória um táxi de entre os que se mostraram interessados nesse passageiro, notificando o escolhido, e devendo este dirigir-se para a posição do passageiro para o recolher e transportar para o seu destino.

1.3 Aplicações

De seguida são apresentados os detalhes das várias aplicações que compõem o sistema.

MapInfo

- Pode existir mais do que uma instância a correr.
- Utilizador genérico – Pode ser usada por qualquer tipo de utilizador.
- A funcionalidade principal é mostrar no ecrã o mapa e situação atual (táxis e passageiros).
- É uma interface gráfica que apresenta a cidade no ecrã com os táxis. Os táxis são representados graficamente e de uma forma que permita distinguir cada táxi dos restantes, assim como saber o estado de cada um (ocupado ou vazio). Esta informação estará permanentemente visível e sempre atualizada.

CenTaxi

- Apenas uma instância a correr no sistema operativo.
- Usada pelo administrador.
- Lê o mapa da cidade de um ficheiro de texto e faz a sua gestão.
- Tem conhecimento dos táxis existentes, da posição e estado dos táxis, e dos pedidos de transporte por parte de passageiros. Esta informação é-lhe dada a conhecer pelas aplicações responsáveis por cada elemento de informação. Por exemplo, quando um táxi se junta ao sistema, o táxi informa esta aplicação acerca da sua existência e posição inicial.
- Tem as seguintes funcionalidades principais:
    - Recebe pedidos de transporte de passageiros. Disponibiliza essa informação, aguarda que os táxis manifestem interesse e atribui esse transporte a um táxi, informando o táxi e o passageiro. Se não existir nenhum táxi interessado, informa o passageiro.
        - Pode assumir que existe um número máximo de passageiros em simultâneo suportado pelo programa. Esta quantidade é indicada como argumento de linha de comandos. Os passageiros que tentam entrar no sistema e que excedem essa capacidade serão ignorados. Assim que o número de passageiros no sistema diminuir (foram entregues no destino) já são aceites novos passageiros.

- Recebe atualizações das posições dos táxis, do início e fim do seu turno de serviço (entrada e saída de táxis no sistema).
    - Pode assumir que existe um número máximo de táxis fixo suportado pelo programa. Esta quantidade é indicada como argumento de linha de comandos. Os táxis que pretendem entrar, mas excedem o número máximo, ficam a aguardar.
- Recebe atualizações acerca da recolha de um passageiro e da entrega de um passageiro no seu destino.
- Recebe comandos do administrador para fechar o sistema, pausar/recomeçar a aceitação de mais táxis, e de expulsar um táxi (se estiver sem passageiros).
- Disponibiliza o mapa aos táxis e à aplicação *MapInfo*.

- Tem uma interface do tipo consola. O administrador tem acesso aos acontecimentos que estão a ocorrer por mensagens que aparecem na consola. O administrador pode, a qualquer instante, introduzir comandos escritos para:
    - Expulsar um táxi do sistema desde que este esteja nesse instante sem passageiros.
    - Encerrar todo o sistema (todas as aplicações são notificadas).
    - Listar táxis (estado atual) e passageiros em transporte e em espera.
    - Pausar e recomeçar a aceitação de novos táxis.
    - Definir a duração do intervalo de tempo durante o qual a CenTaxi aguarda por manifestações de interesse em transporte de um passageiro por parte de táxis.

ConTaxi

- Uma instância a correr por cada táxi.
- Usada pelo condutor do táxi.
- Tem as seguintes funcionalidades principais:
    - Interage com a *CenTaxi*.
    - Movimenta o táxi de forma autónoma, incluindo a possibilidade de responder automaticamente a pedidos de transporte.

      Movimenta-se das seguintes formas:
        - O táxi tem uma velocidade *default* de uma quadrícula do mapa por segundo. Esta velocidade pode variar e considera-se que o táxi muda de posição quando está inteiramente na quadrícula seguinte – ou seja, o táxi nunca está meio na quadrícula anterior, meio na quadrícula seguinte. Isto é mais fácil de entender se não pensar em termos de velocidade, mas sim em termos de pausa entre duas quadrículas (significado oposto, mas mais simples e mais fácil de representar visualmente).
        - O táxi deve ter um movimento o mais autónomo e lógico quanto possível. Se não estiver a ir recolher ou entregar nenhum passageiro desloca-se a direito, mudando aleatoriamente de posição sempre que chega a um cruzamento. Se estiver a ir recolher ou entregar um passageiro, deve tentar ir o mais direito possível ao destino pretendido. Sabendo que o mapa é constituído por ruas retilíneas com cruzamentos em 90 graus, este comportamento é relativamente simples de conseguir.

        - Sempre que entra numa nova quadrícula informa a central.
        - Se estiver vazio e for informado da existência de um passageiro, manifesta interesse nesse transporte se ele estiver a menos que NQ quadrículas de distância (horizontal e vertical). Por omissão, NQ é 10.
    - Permite ao utilizador (o condutor) dar indicações de comportamento (movimento, etc.)
    - O utilizador (condutor) pode, mediante comandos escritos:
        - Acelerar ou desacelerar: de cada vez que acelera ou desacelera, aumenta ou diminui (respetivamente) a velocidade em 0.5 quadrículas por segundo. A velocidade nunca pode ser negativa.
        - Definir NQ
        - Ligar/desligar resposta automática aos pedidos de transporte (independentemente do valor de NQ).
        - Indicar que quer transportar um passageiro se a resposta automática estiver desligada.
        - Os comandos são definidos pelo aluno, que deverá documentá-los no relatório.
    - Recebe os seguintes comandos do condutor (comandos escritos):
        - Para controlo do táxi, como descrito acima, e também para terminar a aplicação (o táxi sai do sistema).
- Tem uma interface do tipo consola.
    - Deverão ser apresentados na consola os acontecimentos relevantes para o táxi: novo passageiro, interesse em transportar enviado, confirmação recebida, recolha de passageiro iniciada, passageiro recolhido, passageiro entregue, etc.

ConPass

- Utilizada pelos passageiros.
- Uma única instância partilhada por todos os passageiros.
    - Os passageiros deverão ter um aidentificação qualquer, dada pelo utilizador, que permita distinguir os passageiros entre si. Pode assumir que não serão introduzidas identificações iguais.
    - A identificação do passageiro constará nas mensgens fornecidas, de forma a perceber o que está a acontecer no sistema.
- Tem as seguintes funcionalidades principais:
    - Indica à central a existência de um novo passageiro que está numa determinada posição e pretende ir para uma outra posição. Esta informação é introduzida pelo utilizador (passageiro).
    - Recebe informações da central: táxi atribuído ao passageiro X, táxi recolheu o passageiro X, táxi entregou o passageiro X.
    - Outras comunicações enviadas/recebidas (ver a restante descrição).
- Tem uma interface do tipo consola.
    - Deverão ser apresentados na consola os acontecimentos relevantes para os passageiros.
    - Os comandos (escritos) para interação com esta aplicação estão em aberto e são definidos pelos alunos.

1.4 Formas de comunicação entre as aplicações

A seguinte descrição refere-se apenas à comunicação (i.e., transmissão de informação) entre processos. Se na situação *S* o enunciado refere que deve ser usado o mecanismo de comunicação *M*, então tem mesmo que usar o mecanismo *M* nessa situação *S*.

- A comunicação entre a central e os táxis é feita de uma forma assimétrica, consoante a direção da informação. Toda a comunicação do táxi para a central é feita por memória partilhada. A disponibilização de informação acerca de um novo passageiro e respetivo aviso é feita também por memória partilhada e por outros mecanismos secundários que sejam necessários. A notificação da central ao táxi que ficou com o passageiro atribuído é feita por *named pipes*. Caso a central encerre, deverá comunicar aos táxis este evento. Estes dois casos são os únicos em que estas duas aplicações comunicam através deste mecanismo e sempre na direção *CenTaxi* → *ConTaxi*.
- A comunicação entre a aplicação *ConPass* e *CenTaxi* é feita por *named pipes*, em ambas as direções.
- A comunicação entra a aplicação *CenTaxi* e *MapInfo* é feita por memória partilhada.

A identificação e aplicação correta de mecanismos de notificação assíncrona e de sincronização fica a cargo dos alunos considerando os cenários em que são necessários segundo a arquitetura e implementação do trabalho. A não aplicação ou o uso incorreto destes mecanismos provoca penalizações na avaliação.

Os mecanismos de comunicação e de sincronização devem constar num diagrama claro e inequívoco a incluir no relatório.

1.5 Aspetos em aberto

Os seguintes aspetos devem ser definidos pelo aluno:

- Texto dos comandos escritos pelo utilizador.
- Pormenores gráficos de visualização.
- Formato das mensagens trocadas entre as aplicações.
- Detalhes do modelo de dados para o mapa da cidade.
- Mecanismos de sincronização: quais e onde são necessários.
- Outros aspetos não previstos ou não explicitamente descritos.

Devem ser tomadas decisões autónomas e lógicas quanto a estes aspetos, e que não desvirtuem o sistema pretendido nem evitem os conteúdos que se pretende ver aplicados. O sistema resultante deve ter uma forma de utilização lógica.

Ao nível do modelo de dados que representa o mapa de cidade, é apresentada a seguinte sugestão: um *array* bidimensional de estruturas. O conteúdo da estrutura deve ser compatível com aquilo que o sistema precisa e com os mecanismos de comunicação onde essas estruturas possam vir a ser usadas. Naturalmente, outras soluções existem e podem ser utilizadas.

É importante perceber que esta disciplina tem mais a ver com mecanismos de SO do que estruturas de dados. Se estiver a considerar usar listas ligadas e árvores binárias, estará a fazer uma má gestão do esforço e tempo.

1.6 Detalhes adicionais acerca dos requisitos

Ao desenvolver o sistema deve também ter em consideração os seguintes requisitos:

- A funcionalidade da aplicação *ConTaxi* relativa à comunicação com a *CenTaxi* deve ser colocada numa biblioteca dinâmica a criar pelo aluno. Esta biblioteca é para uso na *ConTaxi* apenas, ou seja, não se pretende ter uma biblioteca que unifique o funcionamento das duas aplicações diferentes, dado que estas têm lógica diferente. O que se pretende é apenas colocar numa biblioteca uma determinada funcionalidade que vai aparecer repetida em várias instâncias de um programa.
- A parte da memória partilhada onde são colocadas as informações de pedidos de transporte (para consulta pelos táxis) deve ser obrigatoriamente gerida segundo a lógica do paradigma produtor/consumidor com um *buffer* circular de 5 posições (o pedido é removido quando o transporte é atribuído a um táxi).
- A interface gráfica da aplicação *MapInfo* não deve cintilar (deve utilizar a técnica de *double buffering*, abordada nas aulas).
- A aplicação *MapInfo* terá o seguinte comportamento adicional:
    - *Mouseover* do rato em cima de um táxi mostra informação adicional, nomeadamente a matrícula e o destino (este último apenas se estiver a transportar um passageiro).
    - *Click* do rato em cima de um passageiro à espera mostra informação adicional do destino pretendido e a identificação do táxi (se já tiver um atribuído).
    - Terá um menu e uma *dialog box* associada para configurar o "tema", permitindo ao utilizador selecionar imagens/ícones para: táxi livre, táxi ocupado, passageiro à espera sem táxi atribuído, passageiro à espera com táxi atribuído. Esta configuração é memorizada no *Registry*.
- O uso de más práticas na implementação será penalizado. Alguns exemplos, não exclusivos, são:
    - Uso de variáveis globais.
    - Não utilização de Unicode.
    - Má estruturação do código.
    - Mau uso de ficheiros *header* (por exemplo, o típico e errado “geral.h”).

1.7 Biblioteca dinâmica fornecida

Será fornecida uma DLL, sem os ficheiros *.h* e *.lib*, que tem de ser utilizada por **ligação explícita**, e que é essencial para a avaliação do trabalho. Por isso, a sua **utilização é obrigatória**, sob pena de partes da implementação não poderem ser avaliadas na defesa (o que levará à não atribuição de nota nessas partes).
Descrevendo o conteúdo da DLL:

- Tem apenas duas funções:
    - ***void register(TCHAR * name, int type);***
    - ***void log(TCHAR * text);***

- A função ***register*** deve ser utilizada para registar todos os mecanismos de comunicação e sincronização do trabalho, após estes serem criados. O segundo argumento (*type*) pode assumir os seguintes valores:
    - Mutex = 1
    - Critical Section = 2
    - Semáforo = 3
    - Evento = 4
    - Waitable Timer = 5
    - File Mapping = 6
    - View Of (Mapped) File = 7
    - Named Pipe = 8

- A função ***log*** deve ser utilizada para fazer log de mensagens, de forma semelhante a um *printf (TCHAR)*, mas neste caso para a DLL. Devem ser geradas mensagens para o log nos seguintes cenários:
    - Sempre que existir um movimento no mapa por parte de um táxi, devendo a mensagem incluir todos os detalhes desse táxi (matrícula, posição atual, passageiro transportado / vazio, etc.).
    - Sempre que um processo receber informação de outro processo, identificando a origem e o destino dessa informação, o mecanismo de comunicação utilizado, e a informação em si.
    - Sempre que existir uma alteração nos dados "de negócio" do sistema (quando um novo táxi ou passageiro entra no sistema, quando um táxi ou passageiro sai, etc.).

- A função ***log*** não imprime a mensagem que lhe foi passada para o *stdout* (apenas envia a mensagem para um log interno da DLL). Nesse sentido, caso se queira apresentar a mensagem no *stdout*, continua a ser necessário fazer a sua impressão através da função *_tprintf*.

1.8 Aspetos extra

Poderá realizar funcionalidades adicionais com valorização extra. Esta valorização poderá compensar outros aspetos menos bons no trabalho, mas é aplicada de forma relativa à nota das componentes obrigatórias deste. Ou seja, um trabalho avaliado em 60% na sua componente obrigatória e que tenha um extra feito que vale 10%, tem de nota final 66%. A indexação dos extras à parte obrigatória do trabalho tem como objetivo evitar que os alunos façam esses extras em detrimento dos aspetos essenciais de Sistemas Operativos 2. De referir ainda que, caso a nota final com extras ultrapasse os 100%, esta será arredondada para 100%.

Valorização	Resumo	Descrição
10%	Condução inteligente	Condução inteligente dos táxis pela cidade, evitando a movimentação aleatória destes (implica a aplicação de métodos de inteligência artificial).
5 %	Estimativa do tempo de espera	Quando um passageiro pede um táxi, é informado qual é o tempo estimado que tem de ficar à espera. O tempo é calculado pela *CenTaxi* com base numa métrica que faça sentido, por exemplo, identificado o táxi, a sua velocidade média e a distância aprox. ao passageiro, etc. Nota: pretende-se aqui algo pensado pelo aluno e não o mero seguir de sugestões.

1.9 Algumas chamadas de atenção

- Não coloque ponteiros em memória partilhada (pelas razões explicadas nas aulas teóricas). Isto abrange ponteiros seus e também objetos de biblioteca que contenham internamente ponteiros (por exemplo, objetos *STL String*, *Vector*, etc.).
- Pode implementar o trabalho em C++, se assim quiser. Se o objetivo for utilizar polimorfismo, não se esqueça que o polimorfismo em C++ requer ponteiros e que não os pode colocar na memória partilhada. Tenha também em atenção que irá ter de usar uma DLL fornecida pelos docentes que estará feita em C e, portanto, deve garantir a interoperabilidade entre o seu código e essa DLL.
- Planeie cuidadosamente as estruturas de dados, a interação entre os diversos programas e a arquitetura dos mecanismos de comunicação antes de avançar no código.

- Se utilizar repositórios *git,* terá que garantir que são **privados**. Se utilizar um repositório público que depois seja copiado por terceiros, será considerado culpado de partilha indevida de código e terá o seu trabalho anulado.
- A deteção de situações de plágio leva a uma atribuição direta de 0 valores na nota do trabalho aos alunos de todos os grupos envolvidos, podendo ainda os mesmos estar sujeitos a processos disciplinares.
- Todo o código apresentado poderá ser questionado na defesa e os alunos têm obrigatoriamente que o saber explicar. A ausência de explicação coerente é entendida como possível fraude ou como ***falta de conhecimento, que naturalmente se reflete na nota***.

2 Regras e prazos

- **O trabalho é individual**. Existem diversas condicionantes óbvias que obrigam a este cenário e não é possível contemplar outras alternativas.
- O trabalho é composto por duas entregas: a Meta 1 e Meta Final.

Meta 1 – 22 de Maio

As aplicações *CenTaxi* e *ConTaxi* devem já estar parcialmente desenvolvidas, contendo todas as funcionalidades que envolvam os seguintes aspetos da matéria: Unicode, Processos, Registry, DLLs, Threads, Mutexes, Semáforos, Eventos, WaitableTimers e Memória Partilhada. Por outras palavras, todas as funcionalidades das aplicações **CenTaxi** e **ConTaxi**, exceto as que envolvem *named pipes* e comunicação com *MapInfo*, devem estar feitas.

O material a entregar deverá ser:

- Um relatório muito breve a explicar os pontos essenciais da implementação, as estruturas de dados definidas e a sua utilidade, e todos os aspetos que não estejam explicitamente mencionados no enunciado e que tenham sido decididos pelos alunos, e também o diagrama com os mecanismos de comunicação e de sincronização.

  O relatório deve contemplar uma tabela onde são indicados os requisitos implementados, ou parcialmente implementados, no formato:

Num.	Descrição funcionalidade / requisito	Estado
...	...	implementado / não implementado

- As aplicações *CenTaxi* e *ConTaxi* parcialmente implementadas e num estado em que se possa experimentar as funcionalidades já feitas.

Meta Final – 12 de Junho

O material a entregar deverá ser:

- O trabalho completo, com os programas que constituem o sistema totalmente implementados.
- Um relatório completo, com a descrição detalhada de todos programas que constituem o trabalho, e focando os mesmos pontos do relatório da Meta 1 para todas as funcionalidades.

3 Avaliação

O trabalho vale **10 valores**. A nota será atribuída com base nas funcionalidades implementadas, na qualidade das soluções adotadas, na forma como são explicadas no relatório e na qualidade da defesa.

A avaliação e as defesas são feitas, essencialmente, na Meta Final. Na Meta 1 avalia-se o cumprimento dos objetivos estabelecidos através de uma apresentação obrigatória. Tal como descrito na ficha da unidade curricular, considerando as alterações emergentes do plano de resposta ao COVID-19, a avaliação será feita da seguinte forma:

- Nota da Meta 1 = fator multiplicativo entre 0,8 e 1,0
    - A não comparência na apresentação obrigatória da Meta 1 fará com que a mesma não seja considerada. Ou seja, o fator multiplicativo obtido será o mesmo que seria obtido se não entregasse a meta (0,8).
- Nota da Meta Final = valor entre 0 e 10 (influenciado pela defesa)
- Nota final do trabalho = nota obtida na Meta Final * fator multiplicativo obtido na Meta 1
- A Meta 1 não tem qualquer valor sem a Meta Final. Ou seja, grupos que não entregarem a Meta Final ou que não compareçam na sua defesa terão uma nota final de 0 valores.
- A falta da Meta 1 não impede a entrega da Meta Final. Apenas prejudica a nota final, já que a nota mínima da Meta 1 será automaticamente assumida (0,8).

➢ **<u>Importante</u>: dadas as diferenças substanciais das regras de avaliação do trabalho face às do ano passado, o trabalho desse ano deixa de ser aceite, independentemente do valor obtido.**

O trabalho está planeado para ser feito ao longo do semestre e a entrega do trabalho prático é única para todo o ano letivo. Isto significa que não existirá trabalho prático na época especial ou noutras épocas extraordinárias que os alunos possam ter acesso.