Instituto Superior de Engenharia de Coimbra
Departamento de Engenharia Informática e de Sistemas

Sistemas Operativos 24/25

Trabalho Prático - Programação em C para UNIX

Regras gerais

O trabalho prático aborda os conhecimentos do sistema Unix, e deve ser concretizado em linguagem C, para plataforma Unix (Linux), usando os mecanismos deste sistema operativo abordados nas aulas teóricas e práticas. O trabalho foca-se no uso correto dos mecanismos e recursos do sistema e não é dada particular preferência a escolhas na implementação de aspetos de caráter periférico à disciplina (perguntas tais como "devo usar uma lista ligada ou um *array* dinâmico?" não são perguntas importantes. De igual forma, não são relevantes aspetos secundários mais associados ao tema do que à matéria da disciplina.

No que respeita à manipulação de recursos do sistema, deve ser dada prioridade ao uso de chamadas ao sistema operativo[1] face ao uso de funções biblioteca[2] (por exemplo, devem ser usadas *read()* e *write()* em vez de *fread()* e *fwrite()*). Excetua-se aqui a leitura de ficheiros de natureza secundária, tais como ficheiros de configuração ou salvaguarda de dados que podem ser lidos com o habitual *fread* e *fscanf* (já no uso de *named pipes* devem mesmo ser usadas as funções sistema).

Não é permitido o recurso a bibliotecas de terceiros que façam parte do trabalho, ou que ocultem o uso dos recursos e mecanismos estudados na disciplina. O uso de API do sistema Unix que não tenha sido abordado nas aulas é permitido desde que dentro da mesma categoria de recursos estudados - ou seja, variações menores das funções estudadas, e terá que ser devidamente explicado, em particular durante a defesa.

Não é permitida uma abordagem baseada na mera colagem de excertos de outros programas ou de exemplos. Todo o código apresentado terá que ser entendido e explicado por quem o apresenta, caso contrário não será contabilizado.

1. Descrição geral, conceitos principais e elementos envolvidos

O trabalho consiste numa plataforma de envio e receção de mensagens curtas, organizadas por tópicos. Um utilizador pode enviar mensagens para um determinado tópico e subscrever um ou mais tópicos. Sempre que uma mensagem é enviada para um determinado tópico, os utilizadores que o subscreveram recebem essa mensagem. Existirá um programa "***manager***" que gere a receção e distribuição das mensagens, e existirá um outro programa "***feed***" (envia e recebe) que é usado pelos utilizadores para interagir com a plataforma de mensagens. Qualquer ação descrita da forma "*o utilizador faz…*" deve ser entendida como "*o utilizador faz… usado o programa* ***feed***".

A interface com o utilizador é em ambiente de consola (texto). Haverá vários utilizadores, devendo cada um usar um terminal diferente, mas sempre na mesma máquina. *Utilizador da plataforma* e *utilizador do sistema* operativo são conceitos diferentes.

Os utilizadores começam por se identificar (ou seja, "começam por se identificar, usando para isso o ***feed***") junto da plataforma através de um *username*. Os utilizadores não necessitam de qualquer tipo de registo prévio, bastando indicar um *username* que ainda não tenha sido utilizado por qualquer um dos outros utilizadores que já se encontram ligados nesse momento. Não é necessária nenhuma password.

[1] Documentadas na secção 2 das *manpages*

[2] Documentadas na secção 3 das *manpages*

Qualquer utilizador pode enviar mensagens para um determinado tópico. Se esse tópico não existir, e ainda não tiver sido atingido o limite máximo de tópicos do servidor, então o tópico será automaticamente criado e os utilizadores poderão posteriormente subescrevê-lo. Os tópicos são identificados pelo seu nome (ex.: "futebol", "informática", "filmes", etc.), sendo assumido que o nome do tópico apenas uma palavra.

Sempre que um utilizador envie uma mensagem (usando o seu ***feed***) para um tópico, esta mensagem será entregue e apresentada de imediato a todos utilizadores que estejam online e que subscreveram esse tópico.

Existem dois tipos de mensagem: as ***não-persistentes***, e as ***persistentes***. As mensagens ***não-persistentes*** não são memorizadas: assim que o manager as recebe, procede à sua distribuição pelos utilizadores interessados no seu tópico e de seguida descarta-as. As mensagens ***persistentes*** são igualmente distribuídas aos utilizadores interessados no seu tópico, mas ficam também armazenadas no ***manager*** por algum tempo (designado "tempo de vida" da mensagem). Durante esse tempo, as mensagens são entregues a todos os utilizadores que, entretanto, se liguem à plataforma e que subscrevam o tópico dessas mensagens (ou, já estando já online, não tinham ainda subscrito o tópico e, entretanto, fizeram-no). Cada mensagem persistente tem o seu próprio tempo de vida, não tendo de ser igual para todas. O utilizador que envia a mensagem terá sempre de especificar a duração da mensagem (em segundos) quando a envia. Se a duração especificada for 0, a mensagem será tratada como não-persistente. Findo o tempo de vida especificado, o ***manager*** descarta a mensagem.

Ao ser iniciado, o ***manager*** irá buscar a um ficheiro de texto as mensagens persistentes que estivessem ainda dentro do seu tempo de vida (criando implicitamente os respetivos tópicos). Para estas mensagens, a contagem de tempo continua (ou seja, o seu tempo de vida não volta ao "início"). Mais adiante são fornecidos pormenores adicionais.

Programas envolvidos – detalhes adicionais

A plataforma é composta por dois programas: o programa ***feed*** e o programa ***manager*.**

- O programa ***feed*** é usado pelo utilizador para o envio e receção de mensagens, podendo estas ações decorrer em simultâneo (significa que enquanto o utilizador está a escrever uma mensagem, pode surgir no terminal o conteúdo de uma mensagem que tenha surgido num tópico por ele subscrito). A funcionalidade do programa ***feed*** é, essencialmente, a interface de utilizador, sendo toda a gestão da plataforma feita no programa ***manager***. O utilizador apenas conseguirá utilizar a funcionalidade da plataforma depois de se identificar e tendo essa identificação sido aceite. A identificação consiste num *username* (uma palavra apenas), sem qualquer password, sendo sempre aceite desde que não haja já outro utilizador com esse nome. Cada utilizador lançará uma instância do programa ***feed***, podendo haver mais do que uma em simultâneo a um dado instante.
- O programa ***manager*** gere os tópicos, mensagens e utilizadores da plataforma, e interage com os vários ***feed***. Pode interagir diretamente com um utilizador (administrador da plataforma), o qual especifica comandos (recebidos pelo ***manager*** no seu *stdin*) que permitem, por exemplo, desligar a plataforma, mostrar os utilizadores ativos, expulsar um utilizador, etc. Em cada momento existirá no máximo um único processo ***manager*** em execução.

Mais adiante são dados detalhes adicionais acerca destes programas.

Utilizadores - Existem dois tipos de utilizador neste sistema:

- **Cliente**. Este é aquele que, após se identificar, participa na plataforma. Não terá capacidade para interagir diretamente com outros utilizadores. A funcionalidade de envio de mensagens e subscrição de tópicos é feita através de comandos escritos. Toda a informação disponibilizada é apenas texto. Para acrescentar um novo utilizador será simplesmente aberto um novo terminal através do qual interage com o ***feed***.
- **Administrador**. Controla o ***manager*** e é responsável por lançar a sua execução. Pode interagir com a plataforma através do programa ***manager***, seguindo a lógica de comandos escritos no *stdin* deste (estes comandos serão descritos em detalhe mais adiante). É importante salientar que o "administrador" não tem qualquer relação com o administrador (*root*) do sistema operativo.

Todos os utilizadores da plataforma correspondem ao mesmo utilizador do sistema operativo onde os vários programas ***feed*** são executados. Simular a existência de mais um utilizador corresponde a executar mais um ***feed*** num outro terminal.

Conteúdo, gestão e envio das mensagens

Limites predefinidos: Na implementação pode assumir que existem máximos para as seguintes entidades

- Utilizadores: máximo 10
- Tópicos: máximo 20
- Mensagens persistentes (por tópico): máximo 5

Informação das mensagens: As mensagens são constituídas pela seguinte informação

- Nome do tópico: até 20 caracteres, uma única palavra (i.e., sem espaços).
- Corpo da mensagem: texto até 300 caracteres, podendo ter espaços (várias palavras).
- Informação adicional que seja necessária para cumprir os requisitos.
- Esta lista refere-se à informação e não necessariamente a uma estrutura específica de mensagem ou mensagens, assunto que fica a cargo de quem implementa os programas.

Armazenamento de mensagens em memória

- Se for preciso armazenar mensagens em memória, podem ser assumidos os valores máximos indicados acima.
- O armazenamento incluirá a informação necessária para cumprir os requisitos funcionais.
- No programa ***feed*** não é preciso armazenar as mensagens: basta apresentá-las assim que recebidas.

Salvaguarda e recuperação de mensagens persistentes em ficheiro.

- As mensagens do tipo persistente que ainda tenham tempo de vida restante são salvaguardadas em ficheiro quando se encerra o ***manager***, sendo posteriormente recuperadas quando se volta a iniciar o ***manager***.
- O ficheiro em questão é de texto, com o seguinte formato obrigatório:
    - Uma mensagem por linha.
    - Cada linha tem *<nome do tópico> <username do autor> <tempo de vida restante> <corpo da mensagem>*
        - Como é habitual nestes casos, os caracteres < e > delimitam o nome dos campos nesta descrição, não devendo existir realmente no ficheiro nem, tao pouco, ser introduzidos pelo utilizador.
        - Tópico, username e tempo de vida restante serão, cada um deles, "uma única palavra". No entanto, o corpo da mensagem pode ter espaços, devendo ser entendido como sendo tudo o que está entre o tempo de vida e o fim da linha.

    Exemplo

```
futebol sr_silva 110 aquele jogo para a liga foi mesmo estranho
petiscos mario 99 alguem sabe onde se vendem bifanas baratinhas?
```

- O nome do ficheiro está na **variável de ambiente MSG_FICH**, sendo obrigatório o seu uso.

Envio de mensagens

- O corpo da mensagem pode ter até 300 carateres, podendo, no entanto, ser significativamente menor. Uma solução que envie apenas a parte (os caracteres) efetivamente usada pela mensagem, evitando o envio de caracteres não usados, terá uma avaliação superior às soluções que simplesmente enviam mensagens sempre do tamanho máximo.

Gestão do tempo pelo programa manager.

As mensagens persistentes têm um tempo de vida máximo, especificado por quem a envia, tal como descrito atrás. Esse tempo é especificado em número de segundos. O programa ***manager*** deverá remover as mensagens assim que elas expiram. Esta gestão de tempo não está diretamente relacionada com a hora do sistema.

Ficheiros de dados envolvidos

O único ficheiro necessário é aquele que armazena as mensagens persistentes que ainda não expiraram quando o ***manager*** encerra. Os tópicos dessas mensagens não precisam de ser armazenados à parte: são deduzidos e criados à medida que as mensagens persistentes são recuperadas (de forma semelhante a quando é recebida uma mensagem de um tópico que ainda não existia e passa a existir nesse instante).

Sugere-se a utilização das funções biblioteca C (fread, fgets, etc.,) e não as funções de sistema (read, write) para a questão de salvaguarda e recuperação de mensagens persistentes dado que simplificam essas tarefas. Esta sugestão não se estende a outras funcionalidades da plataforma que possa envolver o uso de operações de escrita/leitura, onde prevalecerá a preferência por funções sistema.

2. Utilização da plataforma (funcionalidade vista pelos utilizadores)

Do ponto de vista do utilizador *cliente* (programa **feed**)

- O utilizador cliente executa o programa **feed** indicando o seu *username*, através da linha de comandos. Exemplo:

```
$ ./feed pedro
```

- O ***feed*** só aceita executar se o ***manager*** estiver em funcionamento e não deve permanecer em execução se o ***manager*** terminar.
- O ***feed*** envia inicialmente ao ***manager*** o nome do utilizador. Caso a validação seja bem-sucedida (não exista outro utilizador com o mesmo nome e não tenha sido atingido o limite máximo de utilizadores), o programa ***feed*** fica a aguardar comandos do utilizador. Caso contrário, ***manager*** informa o utilizador do sucedido através do ***feed***.
- O utilizador interage com o ***feed*** através de comandos de texto que permitem, entre outras coisas, o envio de mensagens de texto e a subscrição de tópicos, e correspondente receção das mensagens a eles associadas.
- A comunicação das mensagens entre clientes emissores e subscritores é sempre intermediada pelo ***manager***. A mensagem é enviada pelo emissor ao ***manager*** e este distribui a mensagem por todos os ***feed*** ligados que subscreveram o tópico onde ela foi publicada.

O programa ***feed*** deve lidar em simultâneo com a informação que chega do ***manager*** e com os comandos introduzidos pelo utilizador. Sem prejuízo da usabilidade da interface com utilizador, não é obrigatório o recurso à biblioteca ***ncurses***, sendo suficiente o uso de uma interface baseada no paradigma de consola (*scroll-up* do texto) desde que seja clara e lógica.

O utilizador ***cliente*** pode fazer:

- Obter a listagem de todos os tópicos: **topics**

  Mostra o nome dos tópicos existentes, o número de mensagens persistentes em cada um e o seu estado (bloqueado/desbloqueado – assunto explicado mais adiante).

- Enviar mensagem para um determinado tópico: **msg** <topico> <duração> <mensagem>

  A duração é sempre indicada. Se for diferente de zero a mensagem é automaticamente entendida como sendo persistente. As mensagens serão imediatamente entregues aos utilizadores atualmente ligados que estejam interessados no tópico, ou, se a mensagem for persistente e ainda não tiver expirado, assim que subscreverem o tópico, inclusive os que, entretanto, se liguem.

- Subscrever um determinado tópico: **subcribe** <topico>

  Caso o tópico já exista, o utilizador recebe de imediato todas as mensagens persistentes desse tópico. Caso não exista, e ainda não tenha sido atingido o limite máximo de tópicos do ***manager***, será criado. Em qualquer dos casos, o utilizador passará a receber as mensagens que forem enviadas para esse tópico.

- Deixar de subscrever um determinado tópico: **unsubcribe** <topico>

  O utilizador deixará de receber as mensagens que forem enviadas para esse tópico. Se não existir nenhuma mensagem persistente nesse tópico e nenhum utilizador estiver a subscrever o tópico, então ele desaparece de todo do ***manager***.

- Sair, encerrando o processo ***feed***: comando **exit**

  O ***manager*** deve tomar conhecimento de que o utilizador saiu e reagir de acordo.

Todos os comandos devem ter *feedback* no terminal que indique o sucesso/insucesso da operação e os dados relevantes, conforme o comando escrito.

Todas as mensagens recebidas (relativas aos tópicos subscritos) devem indicar o tópico em que foi submetida, o utilizador que a enviou e o seu conteúdo.

Outras funcionalidades associadas com a visualização

- O utilizador recebe uma notificação/mensagem de cada vez que um tópico por si subscrito é bloqueado/desbloqueado pelo administrador (explicado adiante). Quando um tópico é bloqueado, o cliente não poderá enviar novas mensagens para esse tópico até que o mesmo seja desbloqueado (os comandos **msg** do programa ***feed*** serão ignorados pelo ***manager***). É possível subscrever um tópico bloqueado, recebendo de imediato todas as suas mensagens já existentes.
- O utilizador recebe uma notificação se for eliminado da plataforma pelo administrador. Nesta situação, o programa ***feed*** deverá terminar de forma ordeira, regressando à linha de comandos.

Do ponto de vista do utilizador *administrador* (programa manager)

O administrador executa uma instância do programa ***manager***, sendo que apenas uma instância (processo) deste programa pode estar a correr num dado instante. A validação deste aspeto fica a cargo do programa e não do utilizador. Exemplo de execução:

```
$ ./manager
```

O único utilizador que interage com o ***manager*** é o **administrador** da plataforma, através de comandos escritos para efetuar ações de controlo. Não existe nenhum procedimento de *login*: o **administrador** é simplesmente quem executa o programa ***manager***, ficando a interagir com ele. As ações (comandos) disponíveis ao **administrador** são:

- Listar utilizadores atualmente a usar a plataforma: **users**
- Eliminar um utilizador: **remove** <*username*>
  Tem o mesmo efeito que o comando **exit** do programa **feed**. O utilizador em questão é informado, e o seu programa ***feed*** deve terminar automaticamente. Adicionalmente, todos os outros ***feed*** devem ser informados que o utilizador foi eliminado da plataforma.
- Listar os tópicos existentes na plataforma: **topics**

  São apresentados o nome dos tópicos e o número de mensagens persistentes.
- Listar as mensagens existentes num determinado tópico: **show** <topico**>**

  São apresentadas todas as mensagens persistentes de um determinado tópico.
- Bloquear um tópico: **lock** <topico>

  Bloqueia o envio de novas mensagens para o tópico indicado. As mensagens persistentes continuam a existir até que a sua duração termine, e continua a ser possível a um utilizador subscrever o tópico.
- Desbloquear um tópico: **unlock** <topico>
- Encerrar a plataforma: **close**

  Encerra a plataforma e termina o ***manager***. Todos os processos a correr o ***feed*** são notificados, devendo também terminar. Os recursos do sistema em uso são libertados.

3. Requisitos e restrições

Implementação

- Ficheiros regulares são repositórios de informação - não são mecanismos de comunicação.
- O mecanismo de comunicação entre ***feed*** e ***manager*** é o *named pipe*. O número de *named pipes* envolvidos, quem os cria, e a forma como são usados devem seguir os princípios exemplificado nas aulas. O uso de *named pipes* a mais do que aquilo que é necessário pode ser alvo de penalização.
- Só podem ser usados os mecanismos de comunicação que foram abordados nas aulas.
- A API do sistema deve ser aquela que foi estudada. Qualquer variação deve ser dentro da API estudada. Uma função pouco abordada, mas relacionada com as estudadas, é aceite, mantendo-se dentro do contexto do que foi abordado (exemplo: *dup2* em vez de *dup* é aceite – desde que explicada na defesa, mas uso de memória partilhada não).
- O uso de bibliotecas de terceiros (exceto as fornecidas pelos professores) não é aceite.
- As questões de sincronização que possam existir devem ser acauteladas e tratadas da forma adequada.
- Situações que obriguem os programas a lidar com ações que possam ocorrer em simultâneo não podem ser resolvidas com soluções que atrasem ou impeçam essa simultaneidade. Essas situações, se ocorrerem, têm formas adequadas de solução que foram estudadas nas aulas e devem ser observadas.
- Usos de excertos de código de exemplos (livros, stackoverflow, github, etc.) não poderão ser extensos nem abordar as questões centrais da matéria, devendo ser explicados na defesa.
- Todo o código terá de ser explicado na defesa, tenha ou não sido retirado de exemplos - se não for explicado, será entendido como "o conhecimento não está lá" e, por conseguinte, a parte não explicada não é valorizada.

Terminação dos programas

- Quer seja feita a pedido do utilizador, ou por não estarem reunidos os pressupostos para o programa executar, ou por situação de erro em *runtime*, os programas devem terminar de forma ordeira, libertando os recursos usados, avisando tanto quanto possível o utilizador e os programas com que interajam.

4. Regras gerais do trabalho

Aplicam-se as seguintes regras, descritas na primeira aula e na ficha da unidade curricular (FUC):

- O trabalho pode e deve ser realizado em grupos de dois alunos.
- Existe defesa obrigatória, que é individual e presencial. Podem existir moldes adicionais a definir e anunciados através dos canais habituais na altura em que tal for relevante (por exemplo, ser ou não necessário trazer computador).
- A entrega do trabalho prático é feita via nónio (inforestudante) através da submissão de um único **arquivo *zip*[3]** cujo **nome** respeita o seguinte padrão[4]:

  so_2425_tp_*nome1_numero1_nome2_numero2*.zip

  (*nome* e *número* são os nomes e números de aluno dos elementos do grupo)
- A não adesão ao formato de compressão indicado (.zip) ou ao padrão do nome do ficheiro será penalizada, *podendo levar a que o trabalho nem sequer seja visto*.

[3] Leia-se "***zip***" - não é *arj*, *rar*, *tar*, ou outros. O uso de outro formato poderá ser **penalizado**. Há muitos utilitários da linha de comando UNIX para lidar com estes ficheiros (zip, gzip, etc.). Use um.

[4] O não cumprimento do formato do nome causa atrasos na gestão dos trabalhos recebidos e será **penalizado**.

- Do arquivo zip deverão constar:
    - Todo o **código fonte** desenvolvido;
    - Ficheiro(s) ***makefile*** que possua(m) os *targets* de compilação "all" (compilação de todos os programas), "feed" (compilação do programa ***feed***), "broker" (compilação do programa ***manager***) e "clean" (eliminação de todos os ficheiros temporários de apoio à compilação e dos executáveis); e
    - Um **relatório** (em pdf) com o conteúdo que for relevante para justificar o trabalho realizado e que deverá ser da exclusiva autoria dos membros do grupo.
- Cada grupo submete o trabalho uma vez, sendo indiferente qual dos dois alunos o faz.
- **É obrigatório** que o aluno que faz a submissão **associe no nónio a entrega também ao outro aluno do grupo**.
- **É necessário que ambos estejam inscritos em turmas práticas (mesmo que seja em turmas diferentes). A não inscrição impede o registo do trabalho na plataforma, e, por conseguinte, pode levar à perda da nota**

Data de entrega: Domingo, 15 de dezembro, 2024.

5. Avaliação do trabalho

Para a avaliação do trabalho serão tomados em conta os seguintes elementos:

- **Arquitetura do sistema** – Há aspetos relativos à interação dos vários processos que devem ser planeados de forma a apresentar-se uma solução elegante, leve e simples. A arquitetura deve ser bem explicada no relatório.
- **Implementação** – Deve ser racional e não desperdiçar recursos do sistema. As soluções encontradas devem ser claras e bem documentadas no relatório. O estilo de programação deve seguir as boas práticas. O código deve ter comentários relevantes. Os recursos e API do sistema devem ser usados de acordo com a sua natureza.
- **Relatório** – O relatório deve descrever convenientemente o trabalho. Descrições meramente superficiais ou genéricas de pouco servirão. De forma geral, o relatório descreve e justifica a estratégia e os modelos seguidos, a estrutura da implementação e as opções tomadas. O relatório deve ser entregue juntamente com o código no arquivo submetido.
- **Defesa** – Os trabalhos são sujeitos a defesa individual, durante a qual será verificada a autoria e conhecimentos, podendo haver mais do que uma defesa caso subsistam dúvidas. A nota final do trabalho é diretamente proporcional à qualidade da defesa. Elementos do mesmo grupo podem ter notas diferentes consoante o desempenho e grau de participação individuais que demonstraram na defesa.
  A falta à defesa implica automaticamente a perda da totalidade da nota do trabalho.
- Plágios e trabalhos feitos por terceiros: o regulamento da escola descreve o que acontece nas situações de fraude.
- Os trabalhos que não funcionem serão fortemente penalizados independentemente da qualidade do código-fonte ou arquitetura apresentados. Trabalhos que nem sequer compilam terão uma nota extremamente baixa.
- A identificação dos elementos de grupo deve ser clara e inequívoca (tanto no arquivo zip como no relatório). Trabalhos anónimos não são corrigidos.
- Qualquer desvio quanto ao formato e forma nas submissões (exemplo, tipo de ficheiro) dará lugar a penalizações.

Importante: O trabalho deve ser realizado por ambos os elementos do grupo. Não são aceites divisões herméticas em que um elemento faz uma parte e apenas essa, nada sabendo do restante. Se num grupo existir uma participação desigual, o professor que faz a defesa deve ser informado do facto no início da defesa. Não tendo sendo informado e sendo detetada essa desigualdade, ambos os alunos ficarão prejudicados em vez de apenas aquele que trabalhou menos.