TÉCNICO LISBOA (http://tecnico.ulisboa.pt)

Introdução aos Algoritmos e Estruturas de Dados (https://fenix.tecnico.ulisboa.pt/disciplinas/IAED236/2024-2025/2-semestre)

Enunciado do Projecto - IAED 2024/2025

Data da entrega intermédia (checkpoint): 21 de março de 2025, às 19h59m

Data de entrega: 7 de abril de 2025, às 19h59m

LOG alterações

- 7-mar-25 - Publicação do enunciado.

1. Introdução

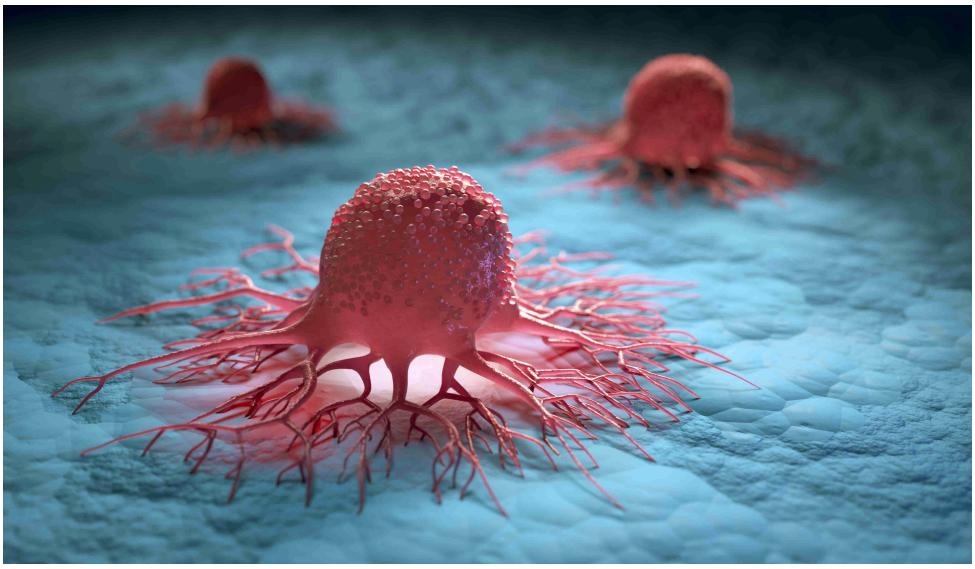
Pretende-se a construção de um sistema de gestão de vacinas e suas inoculações. Para tal, o seu sistema deverá permitir a introdução de doses de vacinas e a gestão das suas aplicações.

A interacção com o programa deverá ocorrer através de um conjunto de linhas compostas por uma letra (comando) e um número de argumentos dependente do comando a executar. Pode assumir que todo o *input* fornecido respeitará os tipos indicados, por exemplo onde é esperado um valor inteiro decimal nunca será introduzida uma letra. Os possíveis comandos são listados na tabela seguinte e indicam as operações a executar.

Comando	Acção
q	termina o programa
c	introduz no sistema um novo lote de uma vacina
l	lista as vacinas disponíveis
a	aplica uma dose da vacina a um utente
r	retira a disponibilidade de uma vacina
d	apaga o registo de aplicações da vacina
u	lista as aplicações a um utente
t	avança o tempo simulado

2. Especificação do problema

O objectivo do projecto é ter um sistema de gestão de inoculações de vacinas a utentes. Para tal são introduzidos no sistema um conjunto de lotes de vacinas e respetivas doses. O sistema gere o registo da aplicação das doses das vacinas aos utentes.



Cada lote de **vacina** é caracterizado por um nome, um lote, uma data de validade e o número de doses disponíveis. O nome não pode conter carateres brancos (espaços, tabuladores ou *newline*), com um comprimento máximo de **50** *bytes*. Notar que um carater acentuado em *utf-8* utiliza mais de um *byte*. Por exemplo `tétano` tem 6 letras mas ocupa 7 *bytes* (`char` em **C**). A designação do lote é constituída no máximo por **20** digitos hexadecimais, onde apenas são aceites dígitos decimais e letras maiúsculas de `A` a `F`. Um lote é único, mesmo para vacinas distintas. A data de validade, constituída por dia-mês-ano, inibe a aplicação de doses da vacina após o termo da validade. As data são sempre impressas com dois dígitos para o dia e mês mas a leitura é efetuada tanto com dois dígitos como com um só dígito. O número de doses que constituem o lote é um número inteiro positivo.

Uma **inoculação** é caracterizada por um nome do utente que a recebe, pelo lote da vacina de onde foi retirada a dose aplicada e pela data da aplicação. O nome que descreve o utente pode conter carateres brancos (espaços ou tabulador horizontal `\t`). Neste caso, o nome do utente é representado entre aspas. Caso não contenha carateres brancos, o nome do utente pode ser delimitado por aspas ou não. O nome do utente nunca contém o carater aspa na sua descrição. O nome do utente não tem comprimento máximo mas na maioria dos casos não excede **200** *bytes*.

Podem existir no máximo **1000** lotes de vacinas registados no sistema. Não existem limites para o número de inoculações nem no nome do utente, pelo que deve procurar utilizar a memória estritamente necessária. Não podem existir variáveis globais. Para facilitar a introdução dos dados, pode assumir que cada instrução não excede 65535 carateres. Se a memória se esgotar, o programa deve terminar de forma controlada, imprimindo a mensagem `No memory.` Antes de terminar, o programa deve libertar toda a memória reservada de forma dinâmica.

3. Dados de Entrada

Durante a execução do programa as instruções devem ser lidas do terminal (*standard input*) na forma de um conjunto de linhas iniciadas por um carácter, que se passa a designar por comando, seguido de um número de informações dependente do comando a executar; o comando e cada uma das informações são separados por pelo menos um carácter branco.

Os comandos disponíveis são descritos de seguida. Os caracteres `<` e `>` são utilizados apenas na descrição dos comandos para indicar os parâmetros. Os parâmetros opcionais estão indicados entre caracteres `[` e `]`. As repetições estão indicadas entre caracteres `{` e `}`. Cada comando tem sempre todos os parâmetros necessários à sua correta execução. Os caracteres `...` indicam possíveis repetições de um parâmetro.

Cada comando indica uma determinada ação que se passa a caracterizar em termos de formato de entrada, formato de saída e erros a retornar.

Se o comando puder gerar mais de um erro, deverá ser indicado apenas o primeiro.

- **q** - termina o programa:
  - Formato de entrada: `q`
  - Formato de saída: NADA
- **c** - introduz no sistema um novo lote de uma vacina:
  - Formato de entrada: `c <lote> <dia>-<mes>-<ano> <número-de-doses> <nome-da-vacina>`
  - Formato de saída: `<lote>` . Insere as doses do lote da vacina indicada.
  - Erros:
    - `too many vaccines` no caso de exceder a capacidade de registo de vacinas do sistema.
    - `duplicate batch number` no caso de já existir um lote com a mesma designação.
    - `invalid batch` no caso de a designação do lote exceder o limite de digitos ou não ser constituídos por dígitos hexadecimais maiúsculos.
    - `invalid name` no caso de o nome exceder o limite de carateres ou conter o carateres inválidos.
    - `invalid date` no caso de a data não representar um dia válido ou este dia já ter sido ultrapassado pela data atual.
    - `invalid quantity` no caso de o número de doses não ser um número positivo.
- **l** - lista os lotes das vacinas criadas:
  - Formato de entrada: `l [<nome-da-vacina> { <nome-da-vacina> } ]`
  - Formato de saída: Imprime informação de cada vacina com o formato `<nome-da-vacina> <lote> <dia>-<mes>-<ano> <número-de-doses-disponíveis> <número-de-aplicaçõe` Imprime as vacinas por ordem cronológica da validade do lote e alfabética do lote para vacinas com a mesma validade. Imprime todas as vacinas caso seja invocado sem argumentos. Imprime as vacinas cujos nomes são indicados, separados por um espaço ( ), pela ordem que são indicados nos argumentos. Lotes apagados (ver adiante) não são listados.
  - Erros:
    - `<nome-da-vacina>: no such vaccine` no caso de não existir uma vacina com o nome indicado.
- **a** - aplica uma dose da vacina a um utente:
  - Formato de entrada: `a <nome-do-utente> <nome-da-vacina>`
  - Formato de saída: Imprime a designação do lote aplicado ao utente, devendo ser escolhido o lote com pelo menos uma dose disponível com validade mais antiga mas ainda válida face à data atual.
  - Erros:
    - `no stock` no caso de não existir uma dose da vacina indicada válida e disponível.
    - `already vaccinated` no caso de o utente já ter sido vacinado nesse mesmo dia para a mesma vacina (`<nome-da-vacina>`).
- **r** - retira a disponibilidade de uma vacina:
  - Formato de entrada: `r <lote>`
  - Formato de saída: `<número-de-doses-já-aplicadas>` . Apaga o lote da vacina indicado caso não tenham sido efetuadas inoculações do lote ou, caso tenha havido inoculações, o lote passa a indicar como sendo constituído pelo número de doses já inoculadas, não ficando doses disponíveis.
  - Erros:
    - `<lote>: no such batch` no caso de não existir um lote com a designação indicada.
- **d** - apaga o registo de aplicações da vacina:
  - Formato de entrada: `d <nome-do-utente> [ <data-de-aplicação> [ <lote> ]]`
  - Formato de saída: `<número-de-aplicações-apagadas>` . Se o `<lote>` for omitido devem ser apagadas todas as aplicações de vacinas do utente para aquele dia. Se apenas for indicado o `<nome-do-utente>` devem ser apagadas todas as aplicações desse utente.
  - Erros:

        - `<nome-do-utente>: no such user` no caso de não existir nenhuma aplicação de vacina ao utente indicado.
        - `invalid date` no caso de a data não representar um dia válido ou este dia ser superior à data atual.
        - `<lote>: no such batch` no caso de não existir um lote com a designação indicada.
- **u** - lista as todas as aplicações ou as aplicações de um utente:
    - Formato de entrada: `u [<nome-do-utente>]`
    - Formato de saída: `<nome-do-utente> <lote> <dia>-<mes>-<ano>` por cada inoculação, por ordem cronológica de aplicação.
    - Erros:
        - `<nome-do-utente>: no such user` no caso de não existir nenhuma aplicação de vacina ao utente indicado.
- **t** - avança o tempo simulado:
    - Formato de entrada: `t [<dia>-<mes>-<ano>]`
    - Formato de saída: `<dia>-<mes>-<ano>` da nova data ou data data atual se for omitido o argumento. A data inicial do sistema é 01-01-2025.
    - Erros:
        - `invalid date` no caso de a data não representar um dia válido ou este dia ser anterior à data atual.

Deve ser possível invocar o programa com o argumento `pt`, ou seja `./proj pt`, devendo as mensagens de erro ser expressas em português, nomeadamente: `demasiadas vacinas`, `número de lote duplicado`, `lote inválido`, `nome inválido`, `data inválida`, `quantidade inválida`, `vacina inexistente`, `esgotado`, `já vacinado`, `lote inexistente`, `utente inexistente`, `sem memória`.

Só pode usar as funções de biblioteca definidas em `stdio.h`, `stdlib.h`, `ctype.h` e `string.h`

Nota importante: não é permitida a utilização da instrução `goto`, da declaração `extern`, nem da função `qsort` nativa do C e nenhum destes *nomes* deve aparecer no vosso código.

Exemplos de utilização dos comandos

Comando `c`

O comando `c` permite criar o registo de um lote de um determinado tipo de vacina, com a validade indicada.

```
c A0C0 31-7-2025 210 malária
```

Comando `l`

O comando `l` sem argumentos permite listar todos os lotes de vacinas existentes no sistema.

```
l
```

O comando `l` com argumentos permite listar apenas os lotes das vacinas cujos nomes são indicados.

```
l tosse_convulsa
l tétano tosse_convulsa malária
```

Comando `a`

O comando `a` permite aplicar uma vacina a um utente.

```
a "João Miguel" tétano
```

Comando `r`

O comando `r` permite remover um determinado lote, não podendo haver mais aplicações desse lote.

```
r A0C0
```

Comando `d`

O comando `d` com um argumento permite apagar todos os registos de aplicações de vacina a um dado utente.

```
d xico
```

O comando `d` com dois argumentos permite apagar todos os registos de aplicações de vacina de um dado utente na data indicada.

```
d xico 1-1-2025
```

O comando `d` com três argumentos permite apagar todos os registos de aplicações de vacina a um dado utente na data indicada para o lote indicado.

```
d xico 1-1-2025 FA54
```

Comando `u`

O comando `u` sem argumentos permite listar todas as aplicações de vacinas existentes no sistema.

```
u
```

O comando `u` com um argumento permite listar todas as aplicações de vacinas existentes no sistema do utente indicado.

```
u xico
u "João Miguel"
```

Comando `t`

O comando `t` sem argumentos permite imprimir a data atual do sistema.

```
t
```

O comando `t` com um argumento permite alterar a data do sistema para uma data posterior.

```
t 2-2-2025
```

4. Compilação e teste

O compilador a utilizar é o `gcc` com as seguintes opções de compilação: `-O3 -Wall -Wextra -Werror -Wno-unused-result`. Para compilar o programa deve executar o seguinte comando:

```
$ gcc -O3 -Wall -Wextra -Werror -Wno-unused-result -o proj *.c
```

O programa deverá escrever no *standard output* as respostas aos comandos apresentados no *standard input*. As respostas são igualmente linhas de texto formatadas conforme definido anteriormente neste enunciado. Tenha em atenção ao número de espaços entre elementos do seu output, assim como a ausência de espaços no final de cada linha. Procure respeitar escrupulosamente as indicações dadas.

Ver os exemplos de input e respectivos output na pasta `public-tests/`.

O programa deve ser executado da forma seguinte:

```
$ ./proj < test.in > test.myout
```

Posteriormente poderá comparar o seu output (`*.myout`) com o output previsto (`*.out`) usando o comando `diff`,

```
$ diff test.out test.myout
```

Para testar o seu programa poderá executar os passos indicados acima ou usar o comando `make` na pasta `public-tests/`.

Para utilizar o *valgrind* ou um debugger (*gdb*, *ddd*, ...) deverá substituir a opção de compilação `-O3` por `-g`.

5. Entrega do Projeto

Será criado um repositório `git` para cada aluno desenvolva e submeta o projeto. Este repositório é criado no GitLab da RNL (https://gitlab.rnl.tecnico.ulisboa.pt) e está ativado na data de publicação deste enunciado.

Na sua submissão do projeto deve considerar os seguinte pontos:

- Considera-se que os seus ficheiros de desenvolvimento do projeto (`.c` e `.h`) estão na raiz do repositório e não numa directoria. *Qualquer ficheiro fora da raíz não será considerado como pertencendo ao seu projeto*.
- A última versão que estiver disponível no repositório da RNL será considerada a submissão para avaliação do projeto. Qualquer versão anterior ou que não esteja no repositório não será considerada na avaliação.
- Antes de fazer qualquer submissão para o repositório da RNL, não se esqueça que deve sempre fazer `pull` para sincronizar o seu repositório local.
- Quando actualizar os ficheiros `.c` e `.h` no seu repositório na RNL, esta versão será avaliada e será informado se essa versão apresenta a resposta esperada num conjunto de casos de teste. Tal como no repositório dos laboratórios, o resultado da avaliação automática será disponibilizado no repositório do aluno.
- Para que o sistema de avaliação seja executado, tem que esperar pelo menos 10 minutos. Sempre que fizer uma actualização no repositório, começa um novo período de espera de 10 minutos. Exemplos de casos de teste serão oportunamente fornecidos.
- Data da entrega intermédia (checkpoint): **21 de março de 2025, às 19h59m**.
- Data limite de entrega final do projeto: **7 de abril de 2025, às 19h59m**. Até à data limite poderá efectuar o número de submissões que desejar, sendo utilizada para efeitos de avaliação a última versão. Deverá portanto verificar cuidadosamente que a última versão no repositório GitLab da RNL corresponde à versão do projeto que pretende que seja avaliada. Não existirão excepções a esta regra.

6. Avaliação do Projeto

Na avaliação do projeto serão consideradas as seguintes componentes:

1. A primeira componente será feita automaticamente e avalia o desempenho da funcionalidade do programa realizado. Esta componente é avaliada entre 0 e 16 valores. Na data da entrega intermédia (checkpoint) o projeto deverá passar a metade dos testes. Na entrega final poderá recuperar metade da cotação dos testes não passados no checkpoint. Seja N_1 a percentagem de testes passados no checkpoint e N_2 a percentagem de testes passados na entrega final. Então, a percentagem da nota de execução será $N_{exec} = N_2 - max(0, ((N_2 - N_1) - 50)/2)$. Assim, por exemplo, se no checkpoint passar a 10% dos testes e na entrega final a 70% dos testes então temos que $N_1 = 10$ e $N_2 = 70$, sendo a percentagem da nota de execução $N_{exec} = 70 - max(0, ((70 - 10) - 50)/2) = 70 - 5 = 65\%$. Suponha que na entrega final passa a apenas 55% dos testes, então se passou a 10% no checkpoint, temos que a percentagem da nota de execução $N_{exec} = 55 - max(0, ((55 - 10) - 50) / 2) = 55\%$. Caso não passe a nenhum teste no checkpoint ($N_1 = 0$) e a todos os testes na entrega final ($N_2 = 100$) então a percentagem da componente de execução $N_{exec} = 100 - max(0, (100 - 0) - 50) / 2) = 75\%$. Se passar a 50% (ou mais) dos testes no checkpoint e a todos os testes na entrega final, então tem a cotação total (100%). Note-se que a fórmula e os exemplos assumem que $N_1 <= N_2$. Em eventuais situações extraordinárias em que $N_1 > N_2$, será considerada a entrega intermédia e $N_{exec} = N_1$.

2. A segunda componente avalia a qualidade do código entregue, nomeadamente os seguintes aspectos: comentários, indentação, alocação dinâmica de memória, estruturação, modularidade e divisão em ficheiros, abstracção de dados, entre outros. Esta componente poderá variar entre -4 valores e +4 valores relativamente à classificação calculada no item anterior e será atribuída posteriormente. Algumas *guidelines* sobre este tópico podem ser encontradas em guidelines.md (https://fenix.tecnico.ulisboa.pt/disciplinas/IAED236/2024-2025/2-semestre/guidelines). Esta componente apenas é avaliada na entrega final.

3. Na segunda componente serão utilizadas as ferramentas *lizard*, *valgrind*, e a opção *-fsanitize* do *gcc* por forma a detectar a complexidade de código, fugas de memória (“memory leaks”) ou outras incorrecções no código, que serão penalizadas. Aconselha-se que utilizem estas ferramentas para fazer debugging do código e corrigir eventuais incorrecções, antes da submissão do projecto. Algumas dicas para degugging podem ser encontradas em debugging.md (https://fenix.tecnico.ulisboa.pt/disciplinas/IAED236/2024-2025/2-semestre/debugging).

4. A classificação da primeira componente da avaliação do projeto é obtida através da execução automática de um conjunto de testes num computador com o sistema operativo GNU/Linux. Torna-se portanto essencial que o código compile correctamente e que respeite o formato de entrada e saída dos dados descrito anteriormente. Projetos que não obedeçam ao formato indicado no enunciado serão penalizados na avaliação automática, podendo, no limite, ter 0 (zero) valores se falharem todos os testes. Os testes considerados para efeitos de avaliação poderão incluir (ou não) os disponibilizados na página da disciplina, além de um conjunto de testes adicionais. A execução de cada programa em cada teste é limitada na quantidade de memória que pode utilizar, e no tempo total disponível para execução, sendo o tempo limite distinto para cada teste.

5. Note-se que o facto de um projeto passar com sucesso o conjunto de testes disponibilizado na página da disciplina não implica que esse projeto esteja totalmente correcto. Apenas indica que passou alguns testes com sucesso, mas este conjunto de testes não é exaustivo. É da responsabilidade dos alunos garantir que o código produzido está correcto.

6. Em caso algum será disponibilizado qualquer tipo de informação sobre os casos de teste utilizados pelo sistema de avaliação automática. A totalidade dos ficheiros de teste usados na avaliação do projeto serão disponibilizados na página da disciplina após a data de entrega.

Attachments

- vacina.jpg (https://fenix.tecnico.ulisboa.pt/downloadFile/1126518382345852/vacina.jpg)