Licenciatura em
Engenharia Informática
UC de Programação Avançada
2º ano – Engenharia Informática
Regime diurno/pós-laboral
Ano letivo 2022/2023 - 1ºSemestre

Teste Prático #2 – Enunciado 11h15

2022.12.17/11h15 | Prova com consulta | **Duração:** 90 minutos

Nome Completo:_____

N.º de Estudante:_____ **Regime:** [ ] Diurno [ ] Pós-laboral

IMPORTANTE

É expressamente proibido o recurso à Internet durante a prova. Qualquer utilização não autorizada da Internet leva à anulação da prova e ao reportar da situação às autoridades competentes. O mesmo sucede com outros tipos de tentativa de fraude.

- **Antes de iniciar a prova:**
  - Execute os seguintes comandos:

    ```
    cd;  mkdir -p ~/ProvaP2/R_NUMERO/
    ```

    (em que **R** deve ser substituído pela letra **D** se for do regime diurno e **N** se for aluno do regime pós-laboral e **NUMERO** deve ser substituído pelo seu número ESTG);
  - Para garantir que o seu diretório de trabalho seja o correto, faça:

    ```
    cd ~/ProvaP2/R_NUMERO/
    ```

- **Após ter terminado a prova:**
  - Deverá proceder à criação de um arquivo TAR, fazendo uso do seguinte comando:

    ```
    cd ~/ProvaP2/R_NUMERO/;  tar cvf ProvaP2_YYYYMMDD_R_NUMERO.tar *
    ```

    (em que YYYYMMDD corresponde à data corrente, e.g., 20221217, e R_NUMERO obedece ao formato acima indicado);
  - Verifique que o arquivo ".tar" que criou não está vazio, através da execução de:

    ```
    tar tvf ProvaP2_YYYYMMDD_R_NUMERO.tar
    ```
  - Entregue o arquivo ".tar" através da plataforma moodle, no espaço reservado para o efeito. Em caso de dúvidas, pergunte ao professor;
  - Informe o professor para este validar a receção dos seus ficheiros.

Pergunta [20 valores]

(Escreva as suas respostas a esta pergunta no diretório **"~/ProvaP2/R_NUMERO/Pergunta"**. Deve indicar o seu nome completo e número de estudante IPLeiria no ficheiro **README.txt** a ser criado no diretório)
NOTA 1: não é permitida a chamada a comandos externos através da função *system* ou de outra com funcionalidade similar.
NOTA 2: a solução deve ser implementada com recurso aos ficheiros do diretório **EmptyProject-client-server-template.v2.02.zip**
NOTA 3: código entregue que **não compile** através do utilitário *make* e do respetivo *makefile* na máquina virtual da UC leva à atribuição da classificação de **0 (zero) valores** à resposta.

Recorrendo à linguagem C, elabore a aplicação cliente-servidor **phone_contacts** para a gestão de uma lista de contactos telefónicos dos estudantes da ESTG. O sistema cliente-servidor phone_contacts comunica através do protocolo UDP. Este serviço deve permitir registar, alterar e consultar o número de telefone de um estudante. O contacto do estudante é descrito no código através de uma estrutura que armazena dois valores inteiros de 32 bits sem sinal, sendo um para o número de estudante e o outro para o respetivo contacto telefónico (por exemplo, 2200123 e 912345678). A lista (ou vetor) de contactos pode conter, no máximo 200 contactos. O comportamento de cada elemento do serviço é descrito de seguida.

Servidor contacts_srv

O servidor desta aplicação funciona através do protocolo UDP e permite gerir remotamente a lista de contactos dos estudantes. O programa recebe um parâmetro de entrada obrigatório que representa o porto de escuta (-p/--port), cujo valor deve estar compreendido entre 1 e 65535, caso contrário o programa termina com uma mensagem de erro apropriada. Assim que inicia, o servidor fica à escuta de pedidos de clientes no porto indicado. O servidor suporta dois tipos de pedidos – **1)** REGISTAR/ALTERAR e **2)** CONSULTAR --, seguidamente descritos:

- **REGISTAR/ALTERAR**– Permite registar ou alterar um contacto de um estudante da ESTG. O servidor espera receber um único pacote de dados UDP, estruturado da seguinte forma:
  - **Operação** – 1 byte, um inteiro de 8 bits sem sinal, que identifica o tipo de pedido. Para a operação de registo/alteração, o primeiro byte do pedido deve conter o valor 1;
  - **Número do estudante** – 4 bytes, um inteiro de 32 bits sem sinal que identifica o número de estudante a registar. O servidor deve garantir que o número está compreendido entre 2 000 000 inclusive e 3 000 000 exclusive. Para além disso, não é permitido registar mais que um contacto para o mesmo número de estudante. Se já existir um registo para este número de estudante, então o servidor deve considerar que o cliente pretende alterar o número de telefone do estudante identificado nesse registo.
  - **Número de telefone** – 4 bytes, um inteiro de 32 bits sem sinal que identifica o número de telefone a registar. O servidor deve garantir que o número está compreendido entre 900 000 000 inclusive e 1 000 000 000 exclusive.

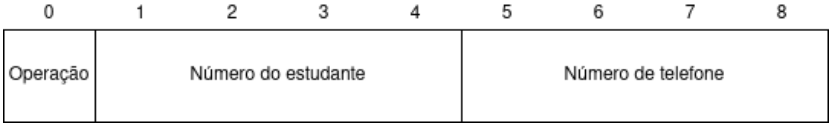

FIGURE 1: REPRESENTAÇÃO BINÁRIA DO PEDIDO REGISTAR/ALTERAR

- **CONSULTAR**– Permite obter o número de telefone de um estudante da ESTG previamente registado no servidor. O servidor espera receber um único pacote de dados UDP, estruturado da seguinte forma:
  - **Operação** – 1 byte, um inteiro de 8 bits sem sinal, que identifica o tipo de pedido. Para a operação de consulta, o primeiro byte do pedido deve conter o valor 2;
  - **Número do estudante** – 4 bytes, um inteiro de 32 bits sem sinal que identifica o número de estudante a consultar.

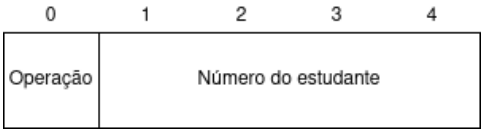

FIGURE 2: REPRESENTAÇÃO BINÁRIA DO PEDIDO CONSULTAR

(continua na página seguinte)

Independentemente do pedido realizado pelo cliente, o servidor responde sempre com um inteiro de 32 bits sem sinal. A tabela seguinte descreve os valores das respostas do servidor a cada tipo de pedido existente.

Pedido	Código da operação	Respostas do servidor
registar/alterar	1	**1** – Novo registo efetuado com sucesso **2** – Registo existente atualizado com sucesso **3** – Número máximo de contactos atingido **4** – Número de estudante inválido **5** – Número de telefone inválido
consultar	2	**1** – Número de estudante não encontrado **Diferente de 1** – Número de telefone do estudante
desconhecido	Diferente de 1 e 2	**0** – Operação desconhecida

Cliente contacts_cli

O cliente desta aplicação permite enviar ao servidor os dois tipos de pedidos descritos anteriormente e mostrar a resposta do servidor no terminal. Os parâmetros da linha de comandos deste programa são:

- **--ip <endereço IPv4>**: *String* com o endereço IPv4 do servidor. Parâmetro obrigatório;
- **--port <port>**: Número inteiro do porto onde o servidor está à escuta de pedidos. Parâmetro obrigatório;
- --**operation/-o <operation>**: *String* com o pedido a realizar. Aceita apenas 2 valores: "**register**" e "**get**". Outros valores devem ser considerados inválidos e a aplicação deve terminar com uma mensagem de erro apropriada (ver exemplos). Parâmetro obrigatório;
- **--student/-s <number>**: Número inteiro do estudante. Parâmetro obrigatório;
- --**phone/-p <phone number>**: Número de telefone do estudante. Parâmetro obrigatório quando a operação é do tipo "register". Caso contrário, o parâmetro deve ser ignorado pelo programa caso seja especificado na linha de comandos.

Considere os seguintes exemplos de funcionamento de um cliente do serviço **phone_contacts**. Note que, nos exemplos apresentados, o servidor foi iniciado localmente no porto 1234.

Exemplos:

```
$ ./client --ip 127.0.0.1 --port 1234 -o register -s 2200123 -p 912345678
[OK] Contact successfully registered!

$ ./client --ip 127.0.0.1 --port 1234 -o register -s 2200123 -p 987654321
[OK] Contact successfully updated!

$ ./client --ip 127.0.0.1 --port 1234 -o register -s 5678 -p 987654321
[ERROR] Invalid student number.

$ ./client --ip 127.0.0.1 --port 1234 -o register -s 2200123 -p 9876
[ERROR] Invalid phone number.

$ ./client --ip 127.0.0.1 --port 1234 -o get -s 2200123
[OK] 987654321

$ ./client --ip 127.0.0.1 --port 1234 -o get -s 2200000
[ERROR] Contact not found.

$ ./client --ip 127.0.0.1 --port 1234 -o invalid_op -s 5678 -p 987654321
[ERROR] Invalid operation. Must be one of: 'register', 'get'.

(…)

$ ./client --ip 127.0.0.1 --port 1234 -o register -s 2200456 -p 918273645
[ERROR] Maximum number of contacts reached.
```