## Redes de Computadores — 2017/2018

## Trabalho Prático de Avaliação nº 1

## Instruções para testes finais e entrega do trabalho

## Indíce

1. Avisos
2. Como fazer testes provisoriamente
3. Instruções para preparar um jar executável e fazer testes
4. Entrega do relatório e do código com a sua solução
5. ANEXO 1 - Como preparar um jar executável
6. ANEXO 2 - Como testar um cliente com um jar executável
7. ANEXO 3 - FAQ

## 1. Avisos

A data final de entrega do trabalho 1 é o dia 20 de Outubro de 2017 pelas 24 horas (meia noite).

Os trabalhos entregues até às 24 horas de dia 21 de Outubro têm 0,5 valores de penalização.

Os trabalhos entregues até às 24 horas de dia 22 de Outubro têm 1 valores de penalização total.

Os trabalhos entregues até às 24 horas de dia 23 de Outubro têm 2 valores de penalização total.

Os trabalhos entregues até às 24 horas de dia 24 de Outubro têm 3 valores de penalização total.

Os trabalhos entregues até às 24 horas de dia 25 de Outubro têm 4 valores de penalização total.

Os trabalhos entregues até às 24 horas de dia 26 de Outubro têm 6 valores de penalização total e assim sucessivamente.

Não são aceites entregas depois do dia 29 de Outubro.

## 2. Como fazer testes provisoriamente

Teste o seu trabalho na diretoria que contém a diretoria **ftp17** da seguinte forma:

Lance o servidor assim:

**java -Xbootclasspath/p:./ftp17/socket.jar ftp17/Ftp17Server**

Lance o seu cliente assim:

**java -Xbootclasspath/p:./ftp17/socket.jar ftp17/Ftp17Client file11000 localhost**

**Nota:** terá de usar **Java 8**

## 3. Instruções para preparar um jar executável e fazer testes

Construir um arquivo executável (por exemplo com o nome **trab1.jar)** com a implementação do cliente desenvolvido (ver as instruções em anexo).

Testar o trabalho usando o script **t1\_test.sh** (ver as instruções em anexo) com o seu jar executável (por exemplo **trab1.jar)** em parâmetro.

## 4. Entrega do relatório e do código com a sua solução

Quando achar que os seus testes e a sua solução são os que vai entregar, prepare as versões finais dos ficheiros com o jar executável (por exemplo **trab1.jar)** e do código fonte Java do seu cliente (por exemplo **Ftp17Client.java** e, eventualmente outros ficheiros necessários ao seu funcionamento), e

1. Responda online ao questionário correspondente ao seu relatório (pode emendar posteriormente a sua resposta):

<https://goo.gl/forms/KrNRo2lsVuYD4lkt1>

1. Entregue o trabalho através da página:

<https://goo.gl/forms/oYD8VNZqDrPJjgaZ2>

Terá de fazer o upload do seu jar executável (por exemplo **trab1.jar)** assim como dos ficheiros com o código fonte Java da sua solução, com linhas numeradas, em formato **pdf** (só do seu código). O script que se usa nos testes **deve** ser usado para gerar esses ficheiros em formato **pdf**.

**bash t1\_test\_sh pdf <zip\_com\_sources> ||| ./t1\_test\_sh pdf <zip\_com\_sources>**

Siga as instruções até obter a mensagem de sucesso da submissão. Se tiver de fazer mais do que uma submissão, só a última será considerada.

## 

## ANEXO 1

## Como construir um jar executável

Para testar uma versão estável do seu **Ftp17Client.java** construa um arquivo jar executável com o código do seu cliente. Notar que é apenas necessário incluir no **jar** executável o cliente e as classes de que este necessita. Para o efeito proceda da seguinte forma:

### Usando o ambiente Eclipse

Siga as instruções na documentação Eclipse que pode encontrar em: <https://eclipse.org/documentation/>

Procure na documentação da sua versão de Eclipse:

**Creating JAR Files > Creating a New Runnable Jar**

Depois de criar o jar executável pode testar se está correto e funciona, executando-o (como se fosse uma aplicação), verificando que exibe o mesmo comportamento do código do cliente que desenvolveu e que antes tinha testado.

### Usando a ferramenta jar numa linha de comando (shell)

Compile o cliente (com javac) obtendo todas as classes correspondentes à implementação e execução do cliente que desenvolveu. Criar na diretoria que contém a diretoria **ftp17** um ficheiro txt (ex., **“manifest.txt”**) com uma única linha

**Main-Class: classname**

em que **classname** será a classe com o Main do seu cliente (ex. **ftp17.Ftp17Client**)

Para criar um jar executável chamado **trab1.jar** (que será um único ficheiro ou arquivo com extensão **.jar**) basta correr o seguinte comando jar na diretoria onde está ftp17:

**jar cvmf ftp17/manifest.txt trab1.jar ftp17/\*.class**

Para verificar em qualquer momento o conteúdo do arquivo jar executável que foi criado dê o comando:

**jar tvf trab1.jar**

Deverá obter um output da forma:

0 Thu Oct 13 11:37:12 WEST 2016 META-INF/

96 Thu Oct 13 11:37:12 WEST 2016 META-INF/MANIFEST.MF

5526 Thu Oct 13 11:29:12 WEST 2016 ftp17/Ftp17Client.class

3817 Thu Oct 13 11:29:10 WEST 2016 ftp17/Ftp17Packet.class

… etc. …...

Para testar que pode executar o seu jar executável **trab1.jar** dê o comando:

**java -jar trab1.jar**

Deve observar o mesmo comportamento da execução do seu cliente que colocou no arquivo executável

## ANEXO 2

## Como testar um cliente com um jar executável

No folder de instruções para entrega do trabalho 1 encontra o seguinte ficheiro:

**t1\_test.sh**

que é um shell script que vai usar para correr e testar o seu trabalho em vários cenários diferentes. Este shell script **só funciona em Linux ou Mac OS X** e requer que o computador tenha o sistema Docker ([http://www.docker.com)](about:blank) instalado, o que já foi feito nos laboratórios nas imagens Linux.

Poderá ter que tornar o script executável com o comando: **chmod +x t1\_test.sh**

Para cada cenário executar o script de teste na seguinte forma:

**sh t1\_test\_sh cenario cliente**

**cenario** — pode ser 1, 2, 3, …...

**cliente —** pode ser um de dois parâmetros:

**SW** — para correr e testar o comportamento de um cliente Stop&Wait

**trab1.jar** — para correr e testar o seu cliente.jar

## Testes 1 a 4

Os testes 1 a 4 são obrigatórios e são assim executados:

**./t1\_test.sh 1 SW**  // Obterá os indicadores de execução de um cliente S&W no cenário 1

**./t1\_test.sh 1 trab1.jar** // Obterá os indicadores de execução do seu cliente no cenário 1

onde figura 1 pode estar 2, 3 ou 4.

**Tem de ter em atenção que o cliente será testado automaticamente com os seguintes parâmetros:**

*java …* ***ficheiro\_de\_teste localhost timeout blocksize window\_em\_blocos***

**O seu cliente tem de seguir o indicado nos parâmetros, em particular, o blocksize e a dimensão da janela em blocos. Se quiser, pode usar um valor de timeout específico inicializado no seu código.**

Os **testes 1 a 4 são obrigatórios** e testam a sua solução perante uma situação de rede cada vez mais deficiente: o jitter e a taxa de perda de pacotes são crescentes do teste 1 ao teste 4. O débito e o tempo de trânsito extremo a extremo são sempre iguais.

**Teste 5**

O cenário 5 é **optativo**. Destina-se a avaliar uma solução que se adapte a uma rede em que exista no seu interior um canal gargalo. Por essa razão, o cliente deve adaptar a janela máxima. Executa-se da mesma forma: **./t1\_test.sh 5 trab1.jar**

Neste cenário não faz sentido testar o cliente S&W.

Registe os resultados para os inserir no form online de submissão.

#### Não esquecer

1) Não deve utilizar o script **t1\_test.sh** em pastas (ou diretorias) com espaços ou caracteres especiais e não altere o conteúdo do script;

2) Faça **chmod +x t1\_test.sh** para tornar o script executável;

3) É normal o script demorar algum tempo a carregar a imagem docker da primeira vez que é executado;

4) No final do teste, o servidor ainda demora cerca de 15 segundos a terminar. A transferência é bem sucedida quando uma mensagem a indicar isso é mostrada;

5) Pode executar os testes tantas vezes quantas quiser.

## ANEXO 3

## Perguntas e respostas frequentes

**Pergunta:** Se pretendo usar um cliente com uma janela de dimensão fixa, qual o valor que devo escolher?

**Resposta:** Para efeitos da recolha de resultados para o relatório deve usar o valor passado em parâmetro. Para os seu próprios testes (sem usar o script) pode usar qualquer valor como 5, 10, 15, 20, 25, 30.