**1.**

Explain the usage/choice of the source UDP ports by the client(s)

Os servidores UDP utilizam portos diferentes para diferenciar clientes diferentes na mesma máquina (IP).

**2.**

Igual ao UDP, o TCP utiliza portos diferentes para diferenciar os clientes na mesma máquina (IP). Isto é igual para os sockets, cada cliente tem uma sessão diferente.

**3.2.**

O servidor usa seletores para gerenciar eficientemente várias conexões de clientes, e cada cliente é tratado independentemente pela função de retorno de chamada apropriada.  
Os objetos SelectorKey são usados principalmente no lado do servidor, onde há gerenciamento de vários sockets e eventos de I/O simultâneos.

**4.2**.

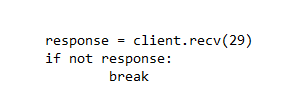
Em relação aos dados, o servidor aguarda conexões entrantes e cria uma nova thread para cada cliente conectado.  
Cada cliente envia um pacote para o servidor contendo a versão, ordem, tamanho e dados.  
O servidor recebe o pacote, desempacota e imprime as informações.

Em termos de descodificação, a função (struct.unpack) é usada em ambos os lados, do servidor e do cliente, para decodificar os pacotes recebidos/enviados. A string de formato ‘! BLL20s’ especifica como os dados estão estruturados no pacote.

**4.3.**

No serverTCPv2.py



No clientTCPv2

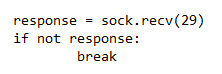
**5.2.**

O servidor e o cliente comunicam-se através de sockets TCP usando uma estrutura específica de pacote. O servidor recebe pacotes, desempacota o cabeçalho para obter informações sobre versão, tamanho e ordem, e então recebe a parte de dados. O cliente empacota os dados em um pacote e envia-os para o servidor. O uso de threads permite que o servidor lide com várias conexões de clientes simultaneamente.

**5.3.**

Server TCP v3  
  


Client TCP v3



Trabalho Realizado por:

* Hugo Dias | 114142 | hugomdias@ua.pt
* João Figueiredo | 112930 | joaofafigueiredo@ua.pt