1)

Uma das operações mais utilizadas para manipulação de arquivos, é a cópia, ou seja, pegar o conteúdo de um arquivo e colocar em outro.

É possível simular, em Java, o comportamento dos comandos *copy* para sistemas Microsoft ou *cp* para Linux, por exemplo, no entanto, nesse caso, não é possível usar o FileWriter seguido do PrintWriter pois os arquivos podem ter tamanhos muito grande e estourar a capacidade da memória, inviabilizando a cópia. Para esse tipo de transição, como comentado em aula, converte-se o conteúdo de arquivo para um array de bytes e utiliza-se um buffer para fazer a transferência do conteúdo do arquivo em espaços de memória que não tomem completamente sua capacidade. Um exemplo de como se fazer isso é:

```
OutputStream os = new FileOutputStream(newArq);
byte[] buffer = new byte[4096];
int len = fis.read(buffer);
while (len != -1){
      os.write(buffer, 0, len);
      len = fis.read(buffer);
}
```

Onde,
`OutputStream os = new FileOutputStream(newArq)` significa que vou inserir um fluxo de bytes no arquivo previamente declarada (newArq);
`byte[] buffer = new byte[1024]` é o buffer que será utilizado para passar o conteúdo do arquivo (nesse caso é 4kB, mas pode ser do tamanho que o programador determinar);
`len = fis.read(buffer)` é o tamanho do buffer que será lido do fis (FileInputStream já carregado anteriormente), o FileInputStream deverá ser lido até que o buffer fique com tamanho negativo (Não tenha mais conteúdo);
`os.write(buffer, 0, len)` vai escrever no o fluxo de bytes (os – OutputStream) que está no buffer, de 0 até o tamanho do buffer, que será colocado no Arquivo (newArq);

Sabendo isso, fazer um Java Project, com uma interface IArquivosController no package controller, cuja assinatura do método seja:
`public void copiaArquivos(String diretorio, String nomeArquivo, String novoDiretorio) throws IOException;`

uma classe ArquivosController, no package controller, que implementa a interface IArquivosController que receba o diretório de uma arquivo, o nome desse arquivo e o diretório destino (Pra onde o arquivo será copiado), valide a existência dos arquivos e diretórios, crie um novo arquivo com o mesmo nome do arquivo que será copiado, receba o FileInputStream do arquivo que será copiado e faça o processo de cópia com o OutputStream descrito acima, usando 1024 de buffer e,

uma classe ArquivosVisao, no package view, cuja a main carregue os endereços dos diretórios, o nome do arquivo que será carregado e chame o método copiaArquivo

2)
Fazer uma aplicação em Java que tenha uma classe de controle que contenha um método que receba um String com um caminho de diretório, faça as validações, e liste apenas os arquivos contidos, em ordem de tamanho (em MB). Para obter o tamanho do arquivo, pegar o double length() do File, que retorna o tamanho do arquivo em bytes.

* Lembrando 1 MB = ((bytes / 1024) / 1024)