

1. O que diferencia um arquivo texto de um arquivo binário?

Resposta:

Arquivo texto representa caracteres legíveis; arquivo binário armazena dados em qualquer sequência de bytes, sendo mais eficiente de processar.

2. O que é um caminho absoluto?

Resposta:

É o caminho que contém todos os diretórios desde a raiz, como `D:\Exemplo\Java\Arquivo.pdf`.

3. O que é um caminho relativo?

Resposta:

É o caminho definido a partir do diretório atual do programa, como `Java\Slide1.pdf`.

4. Para que serve o método `exists()` da classe `File`?

Resposta:

Verifica se o arquivo ou diretório realmente existe no sistema.

5. Qual a diferença entre `isDirectory()` e `isFile()`?

Resposta:

`isDirectory()` retorna true se o objeto representar um diretório; `isFile()` retorna true se representar um arquivo.

6. O que o método `length()` retorna?

Resposta:

O tamanho do arquivo em bytes. Válido apenas para arquivos.

7. Para que serve `listFiles()`?

Resposta:

Retorna um array com arquivos e diretórios contidos em um diretório específico.

8. O que acontece ao usar `new FileOutputStream(File)` quando o arquivo já existe?

Resposta:

O arquivo é recriado e seu conteúdo anterior é destruído.

9. Para que serve o parâmetro `append` em `FileOutputStream(File, boolean)`?

Resposta:

Permite acrescentar dados ao final do arquivo sem sobrescrevê-lo.

10. O que faz o método `write(int)`?

Resposta:

Grava um único byte no arquivo.

11. Qual a função de `flush()`?

Resposta:

Força a gravação dos dados do buffer no arquivo.

12. Qual valor `FileInputStream.read()` retorna ao chegar no final do arquivo?

Resposta:

Retorna -1.

13. O que é o ponteiro do arquivo?

Resposta:

É a posição atual (em bytes) de leitura ou escrita dentro do arquivo. Avança automaticamente.

14. Qual a função de `read(byte[])` em `FileInputStream`?

Resposta:

Lê dados suficientes para preencher o vetor e retorna a quantidade de bytes lidos.

15. Por que não é adequado gravar um `double` diretamente com `FileOutputStream`?

Resposta:

Porque `write()` grava apenas bytes e não interpreta tipos primitivos, causando perda de dados.

16. Para que serve o `DataOutputStream`?

Resposta:

Para gravar tipos primitivos e Strings de forma independente de máquina.

17. Cite três métodos de DataOutputStream.

Resposta:

`writeDouble(), writeBoolean(), writeUTF()`.

18. O que é necessário lembrar ao ler dados gravados com DataOutputStream?

Resposta:

Os dados devem ser lidos na **mesma ordem** em que foram gravados.

19. O que é a exceção EOFException?

Resposta:

É lançada quando se tenta ler além do final do arquivo em DataInputStream.

20. Para que serve o DataInputStream?

Resposta:

Para ler tipos primitivos e strings gravados via DataOutputStream.

21. O que é encadeamento de streams?

Resposta:

É usar múltiplos streams juntos para melhorar funcionalidades, como `FileInputStream` + `BufferedInputStream` + `DataInputStream`.

22. Para que serve o `BufferedInputStream`?

Resposta:

Reduz acessos ao disco, otimizando a leitura com buffer.

23. Qual o tamanho padrão do buffer do `BufferedInputStream`?

Resposta:

8 KB.

24. O que faz o método `skip()` em um input stream?

Resposta:

Avança o ponteiro do arquivo em um número específico de bytes.

25. Para que serve o `BufferedOutputStream`?

Resposta:

Melhora o desempenho da escrita armazenando temporariamente dados em buffer.

26. O que o método `write(byte[])` realiza em `FileOutputStream`?

Resposta:

Escreve todos os bytes de um array no arquivo.

27. Quando é necessário fechar um stream encadeado?

Resposta:

Somente o stream mais externo precisa ser fechado.

28. O que é um Stream na programação de arquivos?

Resposta:

Um objeto capaz de obter ou enviar dados de/para uma fonte como arquivo, teclado ou rede.

29. Quais exceções podem ocorrer durante gravação com `FileOutputStream`?

Resposta:

`SecurityException`, `FileNotFoundException`, `IOException`.

30. Para que serve `readAllBytes()`?

Resposta:

Lê todo o conteúdo do arquivo e retorna um vetor contendo todos os bytes.

1 – Marque V para verdadeiro e F para falso sobre arquivos binários e arquivos texto:

- a) () Arquivos binários não podem ser abertos por nenhum editor.
- b) () Arquivos texto representam caracteres legíveis.
- c) () Arquivos binários usam qualquer sequência de bytes.
- d) () Arquivos texto são mais eficientes para processamento do que arquivos binários.

Gabarito: F, V, V, F

2 – Marque V/F sobre caminhos de arquivos:

- a) () Caminho absoluto sempre começa na raiz.
- b) () Caminho relativo depende do diretório atual.
- c) () Caminho absoluto não pode conter subdiretórios.
- d) () Ambos os tipos podem usar "/" ou "\" como separador.

Gabarito: V, V, F, V

3 – Sobre a classe File, marque V/F:

- a) () File representa arquivos e diretórios.
- b) () exists() verifica se o arquivo realmente existe.
- c) () isFile() retorna true para diretórios.
- d) () length() retorna o tamanho do arquivo em bytes.

Gabarito: V, V, F, V

4 – Sobre operações de E/S, marque V/F:

- a) () listFiles() retorna conteúdo de um diretório.
- b) () createTempFile() cria arquivos temporários.
- c) () getName() retorna o caminho completo.
- d) () isDirectory() retorna true apenas se o File representar uma pasta.

Gabarito: V, V, F, V

5 – Sobre FileOutputStream, marque V/F:

- a) () Sempre apaga o arquivo existente antes de gravar.
- b) () write(int) grava um único byte.
- c) () flush() força a escrita no arquivo.
- d) () O parâmetro append permite sobrescrever o arquivo desde o início.

Gabarito: F (só apaga se não usar append), V, V, F

6 – Sobre exceções na gravação de arquivos:

- a) () FileNotFoundException ocorre quando o diretório não existe.
- b) () SecurityException ocorre por falta de permissão.
- c) () IOException ocorre para falhas genéricas de gravação.
- d) () Nenhuma exceção é lançada ao gravar um arquivo protegido.

Gabarito: V, V, V, F

7 – Sobre FileInputStream, marque V/F:

- a) () read() retorna -1 ao final do arquivo.
- b) () read(byte[]) lê vários bytes de uma vez.
- c) () readAllBytes() retorna um array com todo o conteúdo.
- d) () skip() move o ponteiro para trás.

Gabarito: V, V, V, F

8 – Sobre o ponteiro do arquivo, marque V/F:

- a) () A posição inicial do ponteiro é sempre 1.
- b) () Toda leitura avança automaticamente o ponteiro.
- c) () O ponteiro indica o número do byte atual.
- d) () É impossível alterar a posição do ponteiro.

Gabarito: F, V, V, F

9 – Sobre Streams, marque V/F:

- a) () Streams permitem ler e gravar dados de arquivos.
- b) () Streams só funcionam com arquivos texto.

- c) ☐ FileInputStream é um tipo de Stream.
- d) ☐ Streams podem ser encadeados.

Gabarito: V, F, V, V

10 – Sobre DataOutputStream, marque V/F:

- a) ☐ Permite gravar tipos primitivos.
- b) ☐ writeUTF() grava uma String.
- c) ☐ É dependente da arquitetura da máquina.
- d) ☐ Deve ser usado junto com FileOutputStream.

Gabarito: V, V, F, V

11 – Sobre DataInputStream, marque V/F:

- a) ☐ Deve ler os dados na mesma ordem da gravação.
- b) ☐ Pode ler boolean, double, int e String.
- c) ☐ readUTF() lê Strings gravadas com writeUTF().
- d) ☐ Nunca lança EOFException.

Gabarito: V, V, V, F

12 – Sobre buffers de leitura/escrita:

- a) ☐ BufferedInputStream reduz acessos ao disco.
- b) ☐ O buffer padrão é 8 KB.
- c) ☐ BufferedOutputStream impede a escrita no arquivo.
- d) ☐ flush() deve ser usado para enviar os dados do buffer.

Gabarito: V, V, F, V

13 – Sobre BufferedInputStream, marque V/F:

- a) ☐ Pode ser encadeado a um FileInputStream.
- b) ☐ read() lê um byte do buffer, não diretamente do disco.
- c) ☐ skip() só aceita valores positivos.
- d) ☐ É obrigatório usar DataInputStream junto com ele.

Gabarito: V, V, V, F

14 – Sobre gravação de dados primitivos:

- a) () Não é recomendado gravar double com FileOutputStream diretamente.
- b) () DataOutputStream converte os tipos primitivos automaticamente para bytes.
- c) () A ordem de gravação não importa.
- d) () O arquivo binário pode conter diferentes tipos de dados.

Gabarito: V, V, F, V

15 – Sobre leitura de dados binários:

- a) () DataInputStream permite ler dados de forma independente de máquina.
- b) () Lidos os dados, o ponteiro avança conforme o tamanho lido.
- c) () readDouble() lê 8 bytes.
- d) () DataInputStream pode ler arquivos texto normalmente.

Gabarito: V, V, V, F