Exercício A - SEMANA 01

Aluna: Analu Sorbara

1. Por que Java define fluxos tanto de bytes quanto de caracteres?

   R: Fluxos baseados em bytes representam dados no formato binário, especialmente úteis no trabalho com arquivos. Já fluxos baseados em caracteres armazenam os dados como uma sequência de caracteres, fornecendo um meio conveniente e eficiente de tratamento de caracteres e

2. Já que a entrada e a saída do console são baseadas em texto, por que Java ainda usa fluxos de bytes para esse fim?

   R: Porque no nível mais baixo, todo I/O continua orientado a bytes.

3. Mostre como abrir um arquivo para a leitura de bytes.

   R:

```
import java.io.*;

class MostrarArquivo{

public static void main(String args[])
{
        int i;
        FileInputStream fin;


//abre arquivo

try{
        fin= new FileInputStream(“Arquivo.txt”); //comando que abre
} catch(FileNoFoundException exc) {
        System.out.println(“File Not Found”);
        return;
}

//leitura


try{

do{
        i=fin.read();
        if(i != -1)
        System.out.print((char) i);
```

```
}while(i != -1);
} catch(IOException exc) {
        System.out.println(“Erro na leitura de arquivo”);
}
```

4. Mostre como abrir um arquivo para a leitura de caracteres.

   R:

```
import java.io.*;

class MostrarArquivo{

public static void main(String args[])
{
        char i;
        FileReader fin;


//abre arquivo

try{
        fin= new FileReader((“Arquivo.txt”); //comando que abre
} catch(FileNoFoundException exc) {
        System.out.println(“File Not Found”);
        return;
}

//leitura
try{

do{
        c= (char) fin.read();
        System.out.print(c);
}while(c != ‘.’);
} catch(IOException exc) {
        System.out.println(“Erro na leitura de arquivo”);
}

//leitura
```

5. Mostre como abrir um arquivo para I/O de acesso aleatório.
   R:

```
RandomAccessFile file = new RandomAccessFile(filePath, "r");
```

6. Como podemos converter um string numérico como “123.23” em seu equivalente binário?
   R:
   static double parseDouble(String str) throws NumberFormatException

7. Escreva um programa que copie um arquivo de texto. No processo, faça-o converter todos os espaços em hifens. Use as classes de fluxos de bytes de arquivo. Use a abordagem tradicional para fechar um arquivo chamando close( ) explicitamente.
   R:

```
import java.io.*;

class CopiaArquivo{

        public static void main(String args[]) throws IOException
        {
                int i;
                FileInputStream fin  null;
                FileOutputStream fout = null;

                If(args.length != 2){
                        System.out.println(“Usage: CopiarArquivo de para:”);
                        return;
                }

        try{
                fin = new FileInput.Stream(args[0]);
                fout = new FileInput.Stream(args[1]);

                do{
                        i=fin.read();
                        char c=(char)i;
                        if(c == ‘ ’ ) fout.write(‘-‘);    //transforma espaço em
hifen
                        else if(c != -1) fout.write(i);
                }while(i != -1);


                }catch(IOException exc) {
                System.out.println(“I/O Erro:” +esc);
                } finally {
                try{
                        if(fin != null) fin.close();
                } catch(IOExceptio exc{
                        System.out.println(“Erro para fechar arquivo de
                        saída”);
                }
        }
}
```

```
}
```

8. Reescreva o programa descrito na Questão 7 para que use as classes de fluxos de caracteres. Dessa vez, use a instrução try-with-resources para fechar automaticamente o arquivo.
   R:

```
import java.io.*;

class CopiaArquivo{

        public static void main(String args[]) throws IOException
        {
                char c;


        try(FileReader fin = new FileReader("arquivo.text");
            FileWriter fout = new FileWriter(("arquivo1.text"))
        {


                do{
                        c=(char) fin.read();
                        if(c == ' ' ) fout.write('-');    //transforma espaço em
hifen
                        else if(c != '.') (char) fout.write(c);
                }while(c != '.');


                }catch(IOException exc) {
                System.out.println("I/O Erro:" +esc);
        }
    }
}
```

9. Que tipo de fluxo é System.in?
   R: System.in é a entrada básica e é um fluxos de byte.

10. O que o método read( ) de InputStream retorna quando o fim do fluxo é alcançado?
    R: Retorna -1.

11. Que tipo de fluxo é usado na leitura de dados binários?
    R: Fluxo de byte.

12. Quando usamos o método tradicional de fechamento de arquivo, geralmente fechar um arquivo dentro de um bloco finally é uma boa abordagem. Verdadeiro ou falso?
    R: Verdadeiro, porque o finally é sempre chamado independente do que acontece dentro do try.