Le code ci-dessous calcul la distance de Levenstein entre deux chaînes de caractères - qui sont lues dans deux fichiers passés en paramètre. Une étape du calcul est faite sur le GPU via l'utilisation d'un framework. Après avoir pris connaissance du code, répondez aux 3 questions ci-dessous en cochant les cases.
***Note** : il n'est pas nécessaire de comprendre l'algorithme utilisé, mais plutôt l'utilisation du GPU.*

1. Quel est votre niveau d'expérience en programmation ?

Très grande expérience en programmation	1	2	3	4	5	Aucune expérience en programmation

2. Quel est notre niveau d'expérience en programmation GPU ?

Très grande expérience en programmation GPU	1	2	3	4	5	Aucune expérience en programmation GPU

3. Comment jugez-vous la difficulté à comprendre l'utilisation du GPU dans le code ?

Très difficile à comprendre	1	2	3	4	5	Très facile à comprendre

```java
import com.amd.aparapi.Kernel;
import java.io.FileReader;
import java.io.BufferedReader;
import java.io.IOException;

public class GPLevenshtein {

  public static void main(String [] args) {

    // Params handling
    if(args.length < 2) {
      System.err.println("Needs two arguments: <file a> <file b>");
      System.exit(0);
    }
    String filea = args[0];
    String fileb = args[1];
    char[] a = null; // First string
    char[] b = null; // Second string

    // Reading files, filling a & b
    try {
      FileReader fra = new FileReader(filea);
      BufferedReader bra = new BufferedReader(fra);
      a = bra.readLine().toCharArray();
      FileReader frb = new FileReader(fileb);
      BufferedReader brb = new BufferedReader(frb);
      b = brb.readLine().toCharArray();
    } catch (IOException e) {
      e.printStackTrace();
      System.exit(0);
    }

    // --------------------------------
    long startTime = System.currentTimeMillis();
    // --- Start of benchmark zone --->
    new AparapiLevenshtein(a,b).execute(a.length+1);
    // <--- End of benchmark zone -----
    long stopTime = System.currentTimeMillis();
    // --------------------------------

    // Benchmark time elapsed computation
    long elapsedTime = stopTime - startTime;

    // Output
    System.out.println(elapsedTime/1000);
  }

}
```

```
49
50 // Is the class that sends job on the GPU
51 class AparapiLevenshtein extends Kernel {
52
53   char[] a; // The first string
54   char[] b; // The second string
55
56   int[] costs; // Will contain the result
57   int blen;    // https://github.com/steelted/aparapi/issues/117
58
59   public AparapiLevenshtein(char[] a, char[] b) {
60
61     this.a = a;
62     this.b = b;
63     this.blen = b.length;
64
65     costs = new int [b.length + 1];
66     for (int j = 0; j < costs.length; j++)
67       costs[j] = j;
68
69   }
70
71   @Override
72   // Kernel method
73   public void run() {
74     int i = getGlobalId(); // The id of the thread
75     if(i==0) return;
76
77     costs[0] = i;
78     int nw = i - 1;
79     for (int j = 1; j <= blen; j++) {
80       int cj = min(1 + min(costs[j], costs[j - 1]), a[i - 1] == b[j - 1] ? nw : nw + 1);
81       nw = costs[j];
82       costs[j] = cj;
83     }
84
85   }
86
87   public int min(int a, int b) {
88     return a < b ? a : b;
89   }
90
91 }
```