Ejercicio 1 – Recursividad y transformaciones

Decidir si los elementos de una lista de enteros forman una progresión aritmética. Una progresión aritmética es una sucesión de números tales que su distancia (diferencia de cualquier par de términos sucesivos de la secuencia) es constante. Se considera que una lista vacía o con tamaño igual a 1 o 2, son progresiones aritméticas.

SE PIDE:

1. Implementar una solución iterativa (Java y C).
2. Implementar una solución recursiva.
3. Implementar una solución funcional.

TIPO LISTA DE ENTEROS EN C:

```c
typedef struct {
	int tam;
	int size;
	int * data;
}int_list;
```

SOLUCIÓN RECURSIVA (NO FINAL)

```java
public static Boolean esAritmetica(List<Integer> ls) {
	Boolean r = true;
	if (ls.size() > 2) {
		int dif = ls.get(1) - ls.get(0);
		r = esAritmetica(ls, 2, dif);
	}
	return r;
}

private static Boolean esAritmetica(List<Integer> ls, int i, int dif) {
	Boolean r = true;
	if (ls.size() - i > 1) {
		r = ls.get(i + 1) - ls.get(i) == dif &&
			esAritmetica(ls, i + 1, dif);
	}
	return r;
}
```

SOLUCIÓN RECURSIVA (FINAL)

```java
// Versión final equivalente a la anterior con acumulador all
public static Boolean esAritmeticaFinal(List<Integer> ls) {
	Boolean r = true;
	if (ls.size() > 2) {
		int dif = ls.get(1) - ls.get(0);
		r = esAritmeticaFinal(ls, 2, true, dif);
	}
	return r;
}
```

```
private static Boolean esAritmeticaFinal(List<Integer> ls, int i, Boolean b,
            int dif) {
      Boolean r = true;
      if (ls.size() - i > 1 && b) {
            b = ls.get(i + 1) - ls.get(i) == dif;
            r = esAritmeticaFinal(ls, i + 1, b, dif);
      }
      return r;
}
```

SOLUCIÓN ITERATIVA JAVA

```
public static Boolean esAritmeticaIterativa(List<Integer> ls) {
      Boolean b = true;
      if (ls.size() > 2) {
            int dif = ls.get(1) - ls.get(0);
            Integer i = 2;
            while (ls.size() - i > 1 && b) {
                  b = ls.get(i + 1) - ls.get(i) == dif;
                  i = i + 1;
            }
      }
      return b;
}
```

SOLUCIÓN ITERATIVA C

```
boolean esAritmeticaIterativa(int_list ls) {
      boolean b = true;
      if (lista.size > 2) {
            int dif = lista.data[1] - lista.data[0];
            int i = 2;
            while (lista.size - i > 1 && b) {
                  b = lista.data[i + 1] - lista.data[i] == dif;
                  i = i + 1;
            }
      }
      return b;
}
```

SOLUCIÓN FUNCIONAL

```
public static Boolean esAritmeticaStream(List<Integer> ls) {
      Boolean r = true;
      if (ls.size() > 2) {
            final int dif = ls.get(1) - ls.get(0);
            r = IntStream.range(2, ls.size() - 1)
                  .allMatch(i -> ls.get(i + 1) – ls.get(i) == dif);
      }
      return r;
}
```