Parte 3: Fuga de Memoria (Memory Leak)

El siguiente programa simula el procesamiento de un archivo encriptado. Aunque el programa compila y produce la salida de consola esperada, se ha detectado que su ejecución genera una **fuga de memoria (memory leak)**.

```
#include <iostream>

using namespace std;

class Archivo {
public:
    Archivo() {}
    ~Archivo() {
        // Nada que liberar en la base
    }
    virtual void procesar() = 0;
};

class ArchivoEncriptado : public Archivo {
private:
    int* llave;
    int tamano;
public:
    ArchivoEncriptado(int t) : tamano(t) {
        llave = new int[tamano];
        llave[0] = t * 2;
    }
    ~ArchivoEncriptado() {
        delete[] llave;
    }
    void procesar() override {
        cout << "Procesando llave: " << llave[0] << endl;
    }
};

int main() {
    int t;
    if (cin >> t) {
        Archivo* arch = new ArchivoEncriptado(t);
        arch->procesar();
        delete arch;
    }
    return 0;
}
```

Instrucciones: Tu tarea consiste en identificar la causa técnica de la fuga de memoria, corregir el código del programa y entregar la versión reparada que asegure la destrucción completa de los recursos asignados.