GUÍA DE AYUDA PARA EL EXAMEN DE C++

```
========================================
GUÍA DE AYUDA PARA EL EXAMEN DE C++
========================================

Contenido basado en las prácticas realizadas.

-------------------------
1. HERENCIA DE CLASES
-------------------------

Archivo: lab2_e1.cpp

- Se crea una clase base `Machine` y una clase derivada `CNC`.
- El constructor de `CNC` llama explícitamente al constructor de `Machine`.

Tips:
- Siempre usar el constructor base en la lista de inicialización: `CNC(string name) : Machine(name) {}`
- Para acceder a un método de la clase base desde la derivada: `ClaseBase::metodo()`

Ejemplo clave:
class Machine {
    public:
        Machine(string name) { cout << "Machine: " << name << endl; }
};

class CNC : public Machine {
    public:
        CNC(string name) : Machine(name) { cout << "CNC creada" << endl; }
};

------------------------------
2. CLASES ANIDADAS Y ARREGLOS
------------------------------

Archivo: lab2_e2.cpp

- Clases anidadas: `Program` tiene estudiantes, y `Student` tiene cursos.
- Se calcula el promedio de notas de cada curso.

Tips:
- Para copiar arreglos, se puede usar un bucle en el constructor.
- Se pueden usar listas de inicialización para arreglos fijos.

Punto clave:
- Se accede a las notas así: `students[i].courses[j].getAverageScore()`

-----------------------------------
3. USO DE COLAS (queue<int>)
-----------------------------------

Archivo: lab3ex2.cpp

- Se rellena una cola con enteros, se imprimen, se borran elementos y se controla un límite de tamaño.

Tips:
```

- `queue.front()` accede al primer elemento.
- `queue.pop()` elimina el primero.
- `queue.size()` da el número de elementos.

Truco:
- Para pasar la cola por valor y mostrarla sin modificarla:

```
void mostrar(queue<int> q) {
    while (!q.empty()) { cout << q.front(); q.pop(); }
}
```

---

4. COLAS CON CLASES (Applicant)

Archivo: lab03t3.cpp

- Uso de clase `Applicant` en una cola.
- Se controla el número de solicitantes, eliminando el más antiguo.

Tips:
- Para evitar sobrepasar el límite, verificar antes de insertar:

```
if (cola.size() >= limite) { cout << "Servicio suspendido"; return; }
```

---

5. MANEJO DE ARCHIVOS: lectura/escritura

Ejemplos:
- Lectura de archivo y cálculo de promedio.
- Lectura, expansión de caracteres y escritura invertida.

Tips:
- `ifstream` para leer archivos, `ofstream` para escribir.
- Siempre verificar si el archivo se abrió bien.

Ejemplo:

```
ifstream in("archivo.txt");
ofstream out("nuevo.txt");
if (in && out) {
    string linea;
    while (getline(in, linea)) out << linea << endl;
}
```

---

6. HILOS (THREADS) Y CONCURRENCIA

Archivos: varios ejemplos con `std::thread`

- Se crean hilos con funciones que imprimen mensajes.
- Uso de `join()` para esperar que terminen.
- `mutex` para evitar condiciones de carrera.

Tips:
- `std::thread` acepta funciones y argumentos.
- `std::lock_guard<std::mutex>` bloquea secciones críticas.

Ejemplo:

GUÍA DE AYUDA PARA EL EXAMEN DE C++

```
std::mutex mtx;
void tarea() {
    std::lock_guard<std::mutex> lock(mtx);
    cout << "hilo seguro";
}
```

TIPS GENERALES PARA EL EXAMEN

- Lleva estructuras básicas listas para copiar/pegar: clase, main, etc.
- Anota diferencias entre `struct` y `class` (por defecto: `public` vs `private`).
- Anota comandos comunes: `#include`, `using namespace std`, etc.
- Ten ejemplos de herencia, manejo de archivos, `queue`, `thread`, `mutex`.

¡Éxito en el examen!