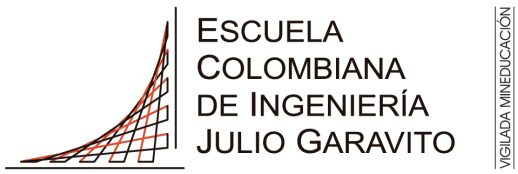


Parcial #1 Ejercicio #4

Algoritmos En Sistemas Electrónicos

Presentado a: Daniel Palomino

Integrantes: Jaime Enrique Galvis, Maira alejandra Prieto

1. 1. Estructura `Grade`

```cpp
// Definir la estructura de una nota (grade)
You, 14 minutes ago | 1 author (You)
struct Grade
{
    int studentId;
    double grade;
};
```

Descripción: Esta estructura modela una nota de un estudiante.

- `studentId`: Se ingresa el numero entero Id con el que se identifica el estudiante.
- `grade`: Nota del estudiante, un valor real entre 0.0 y 5.0.

2. Función `leerNotas(Grade notas[], int n)`

Propósito: Rellenar el arreglo `notas` de tamaño `n` con datos ingresados por el usuario o con valores predeterminados.

```cpp
void leerNotas(Grade notas[], int n)
{
    char respuesta;
    std::cout << "¿Desea ingresar las notas manualmente? (s/n): ";
    std::cin >> respuesta; // leer la respuesta del usuario

    // Verificar si la respuesta es 's' o 'S'
    if (respuesta == 's' || respuesta == 'S')
    {
        for (int i = 0; i < n; ++i)
        {
            while (true)
            {
                std::cout << "Ingrese las notas del estudiante " << i + 1 << " (id nota): ";
                int id;
                double cal;
                std::cin >> id >> cal;

                if (std::cin.fail())
                {
                    std::cout << "Entrada invalida. Intente de nuevo.\n";
                    std::cin.clear();
                    std::cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
                    continue;
                }
                if (cal < 0.0 || cal > 5.0
                {
                    std::cout << "La nota debe estar en el rango 0.0 - 5.0. Intente otra vez.\n";
                    continue;
                }

                notas[i].studentId = id;
                notas[i].grade = cal;
                break;
            }
        }
    }
```

(const char [61])"La nota debe estar en el rango 0.0 - 5.0. Intente otra vez.\n"
Generate Copilot summary

```cpp
    else
    {
        // Usar valores predeterminados (si n > defaults, se reciclan)
        std::cout << "Usando valores predeterminados...\n";
        Grade defaults[4] = {
            {1010, 1.0},
            {1020, 1.0},
            {1030, 1.0},
            {2056, 1.0}
        };
        for (int i = 0; i < n; ++i)
        {
            notas[i] = defaults[i % 4];
        }
    }
}
```

a) Se le pide al usuario si desea ingresar las notas manualmente (`'s'`/`'S'`) para sí.

b) Si la respuesta es afirmativa:

- Para cada posición $i = 0 \ldots n - 1$:
  1) Solicita `id` y nota.
  2) Si la lectura falla, limpia el estado, descarta la línea, y pide la entrada nuevamente.
  3) Valida que la `nota` esté en el rango [0.0, 5.0]. Si no, lo solicita de nuevo.
  4) Al ser válida, asigna `notas[i].studentId = id` y `notas[i].grade = cal`.

c) Si la respuesta no es afirmativa:

- Se usan 4 valores predeterminados en el arreglo `defaults` y se copian cíclicamente en `notas`.

3. **Función calcularPromedio**

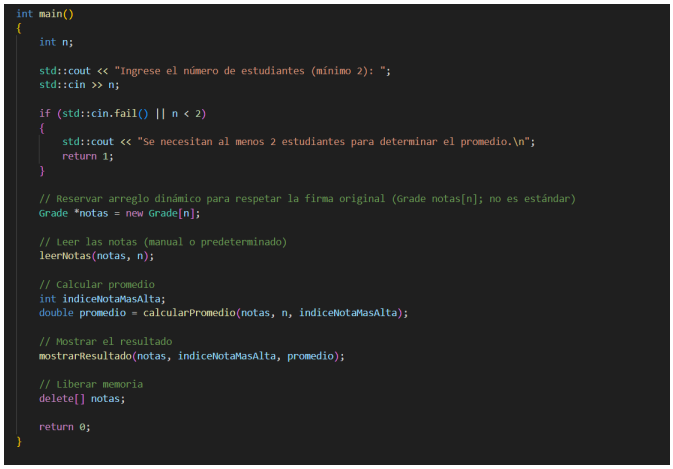
La función `calcularPromedio` recibe un arreglo de enteros y su tamaño n para calcular el promedio de los valores.

```
double calcularPromedio(Grade notas[], int n, int &indiceNotaMasAlta)
{
    double suma = 0.0;
    indiceNotaMasAlta = 0;

    for (int i = 0; i < n; ++i)
    {
        suma += notas[i].grade;
        if (notas[i].grade > notas[indiceNotaMasAlta].grade)
        {
            indiceNotaMasAlta = i;
        }
    }

    return suma / static_cast<double>(n);
}
```

- Declara una variable `suma` que inicia en cero.
- Recorre todos los elementos del arreglo acumulando sus valores en `suma`.
- El resultado final se obtiene dividiendo la suma entre n.
- Se realiza un `cast` a `double` para asegurar que el promedio sea un valor decimal.

4. **Función main**

La función `main` es el punto de entrada del programa.



```cpp
int main()
{
    int n;

    std::cout << "Ingrese el número de estudiantes (mínimo 2): ";
    std::cin >> n;

    if (std::cin.fail() || n < 2)
    {
        std::cout << "Se necesitan al menos 2 estudiantes para determinar el promedio.\n";
        return 1;
    }

    // Reservar arreglo dinámico para respetar la firma original (Grade notas[n]; no es estándar)
    Grade *notas = new Grade[n];

    // Leer las notas (manual o predeterminado)
    leerNotas(notas, n);

    // Calcular promedio
    int indiceNotaMasAlta;
    double promedio = calcularPromedio(notas, n, indiceNotaMasAlta);

    // Mostrar el resultado
    mostrarResultado(notas, indiceNotaMasAlta, promedio);

    // Liberar memoria
    delete[] notas;

    return 0;
}
```

- Solicita al usuario el tamaño del arreglo (`n`).
- Declara un arreglo dado por el usuario en base al valor ingresado.
- Llama a la función `leerArreglo` para llenar el arreglo con datos ingresados por el usuario.
- Llama a `calcularPromedio` y guarda el resultado en la variable `promedio`.
- Muestra en pantalla el promedio calculado.

Aquí está el enlace a mi repositorio: GitHub Repo