

# Ayudantía 3

## Repaso y Arreglos

Benjamín Aceituno

Programación Avanzada S09



# Arreglos de Objetos

Nosotros podemos agregar a un arreglo de objetos a otra clase aparte de nuestra clase que contiene nuestros objetos, donde el arreglo de objetos será un atributo de nuestra clase.

```
#include <iostream>
using namespace std;

class Persona {
private:
    string nombre;

public:
    Persona(string nombre) {
        this->nombre = nombre;
    }

    void mostrarNombre() {
        cout << "Nombre: " << nombre << endl;
    }
};

class Grupo {
private:
    Persona* personas[3]; // arreglo de punteros de objetos personas
    // se accede con un puntero a la clase persona
public:
    Grupo() {
        // Inicializar el arreglo en NULL
        for (int i = 0; i < 3; i++) { // ciclo que repasa todas las personas
            personas[i] = NULL; // por cada indice del arreglo, se inicialzia en null
        }
    }
}
```

# En síntesis...

Tenemos que pasarle el parámetro  
**persona**

```
// método que agrega una persona al grupo en el primer espacio disponible
void agregarPersona(Persona* persona) { // recibe como parámetro un objeto persona de la clase persona
    for (int i = 0; i < 3; i++) { // ciclo que repasa todas las personas
        if (personas[i] == NULL) { // si el índice personas está vacío
            personas[i] = persona; // el parámetro que recibe el método, se define en personas[]
            cout << "Persona agregada al grupo." << endl;
            return;
        }
    }
    cout << "El grupo está lleno, no se puede agregar más personas." << endl;
}
```

**Y con un ciclo e if, agregar al arreglo usando su índice y el parámetro que le pasamos.**

## En síntesis...

```
// Mostrar las personas del grupo
void mostrarPersonas() {
    for (int i = 0; i < 3; i++) { // recorre todas las personas
        if (personas[i] != NULL) { // si personas no está vacío (osea, se ingresó una persona)
            // se llama al metodo mostrarNombre, y muestra a la persona de dicho indice
            personas[i]->mostrarNombre();
        } else {
            cout << "Persona en índice " << i << " está vacía." << endl;
        }
    }
}
};
```

Este método es más de lo mismo, llama al método `mostrarNombre` dentro de "Personas". Este método está en "Grupo". Así podemos mostrar multiples elementos del arreglo.

## En síntesis...

```
int main() {  
    Grupo* grupo = new Grupo(); // se crea un grupo  
    Persona* p1 = new Persona("René Puente"); // se crean personas  
    Persona* p2 = new Persona("Isidora Ortiz");  
    Persona* p3 = new Persona("Diego Mena");  
  
    //agregar personas al grupo  
    grupo->agregarPersona(p1);  
    grupo->agregarPersona(p2);  
    grupo->agregarPersona(p3);  
    //mostrar las personas del grupo  
    grupo->mostrarPersonas();  
    return 0;  
}
```

Apuntamos a agregarPersona, y le pasamos el objeto que se acaba de crear.

Si se dan cuenta es la clase que contiene los objetos la que realiza todas las llamadas a los métodos.

## En síntesis...

```
void soyUnMétodo() {  
    for (int i = 0; i < 3; i++) {  
        if (personas[i] != NULL) {  
            //Aquí está lo que hago normalmente!!  
        } else {  
            cout << "Está vacío, aquí no hay nada que ver... bruh..." << endl;  
        }  
    }  
}
```

Casi siempre se repite lo mismo, recorreremos todos los elementos del arreglo y verificamos si tiene un contenido dentro de este mismo para realizar una acción

## En síntesis...

```
void soyUnMétodoQueRecibeAlgo(Persona* persona) {  
    for (int i = 0; i < 3; i++) {  
        if (personas[i] == NULL) {  
            personas[i] = persona;  
            cout << "Persona agregada al grupo." << endl;  
            return;  
        }  
    }  
    cout << "El grupo está lleno, no se puede agregar más personas." << endl;  
}
```

Aquí lo mismo, recorremos y verificamos...

Y lo que estamos recibiendo lo ingresamos. PERO, solo si es nulo, ojo. Y definimos nuestro índice del arreglo como el parámetro que estamos pasando.

# Revisión LAB

Ahora se hará una  
revisión del  
laboratorio,  
explicando paso  
por paso qué es  
lo que había que  
hacer.

```
#include <iostream>
using namespace std;

class Criminal
{
private:
    string nombreCriminal; // Atributo para almacenar el nombre del criminal
    int numeroIdentificacion, tiempoCondena; // Atributos para almacenar el número de identificación y tiempo de condena

public:
    Criminal(string nombreCriminal, int numeroIdentificacion, int tiempoCondena)
    {
        this->nombreCriminal = nombreCriminal;
        this->numeroIdentificacion = numeroIdentificacion; // Inicialización del número de identificación
        this->tiempoCondena = tiempoCondena; // Inicialización del tiempo de condena
    }

    // SE HACE UN GET POR CADA ATRIBUTO DEL CRIMINAL
    string getNombre() // Método para obtener el nombre del criminal
    {
        return nombreCriminal;
    }

    int getNumeroIdentificacion() // Método para obtener el número de identificación del criminal
    {
        return numeroIdentificacion;
    }

    int getTiempoCondena() // Método para obtener el tiempo de condena del criminal
    {
        return tiempoCondena;
    }

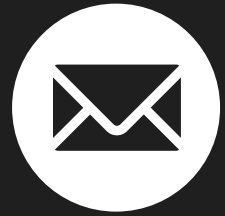
    void actualizarTiempoPrision(int diasPasados) // Método para actualizar el tiempo de condena
    {
        tiempoCondena -= diasPasados; // Disminuye el tiempo de condena por los días que han pasado
    }
};
```



# Contacto



+569 86031881



benjamin.aceituno@mail.udp.cl



benja.mp4



[https://github.com/benjamp4/  
prog.av-2024-2](https://github.com/benjamp4/prog.av-2024-2)

