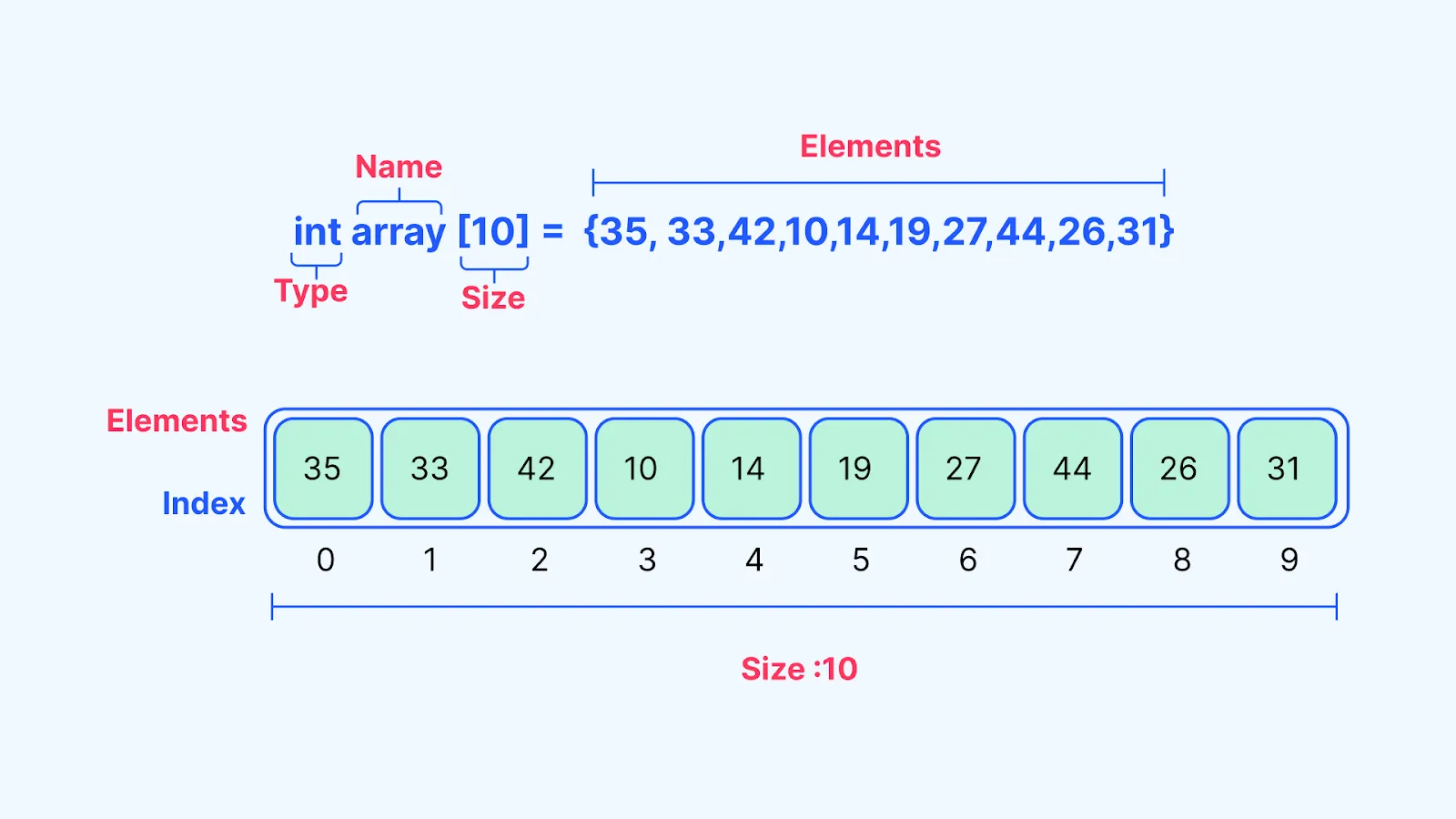
**TP N ° 10 - Vectores de Clases**

Alumno: Nahuel vilca



1. Leer el material adjunto, ejecutar y analizar el código de ejemplo, luego:

**Resolver las siguientes actividades:**

1. ¿Cuál es la principal diferencia entre un arreglo y un vector en C++? Elaborar un cuadro comparativo
2. ¿Cuál es la diferencia entre un vector tradicional y un vector de clase en C++? Elaborar un cuadro comparativo con las principales características de cada uno
3. Resolver las actividades planteadas en el TP Nº10.

**NOTA**: Organizar las respuestas en un repositorio en GitHub, compartir el link como respuesta a la actividad

# Definicion Arreglos y Vectores en C++

Los arreglos tienen un tamaño fijo y menos funciones, mientras que los vectores son dinámicos, pueden crecer o reducirse en tiempo de ejecución y ofrecen métodos que facilitan su manejo. En general, los vectores son más flexibles y seguros que los arreglos tradicionales.

|  |  |
| --- | --- |
| **Concepto** | **Descripción** |
| **Definición de Arreglo** | Un arreglo es una estructura de datos de tamaño fijo que almacena una colección de elementos del mismo tipo, definidos en memoria de forma contigua. |
| **Definición de Vector** | Un vector es una estructura dinámica de la biblioteca estándar de C++ (<vector>) que permite almacenar elementos del mismo tipo y cambiar su tamaño durante la ejecución. |

# Arreglo vs Vector en C++

|  |  |  |
| --- | --- | --- |
| **Característica** | **Arreglo (array)** | **Vector (vector)** |
| **Tamaño** | Fijo: no puede cambiar después de ser declarado. | Dinámico: puede crecer o reducir su tamaño en tiempo de ejecución. |
| **Declaración** | int numeros[5]; | vector<int> numeros; |
| **Gestión de memoria** | La memoria se asigna de forma estática. | La memoria se asigna dinámicamente según sea necesario. |
| **Métodos disponibles** | No posee métodos integrados (solo acceso con índices). | Tiene muchos métodos útiles (push\_back, size, clear, etc.). |
| **Facilidad de uso** | Menor flexibilidad; se deben conocer los tamaños previamente. | Más flexible; se puede agregar o quitar elementos fácilmente. |
| **Seguridad** | No realiza comprobación de límites. | Puede lanzar excepciones o comprobar límites con .at(). |
| **Uso típico** | Cuando el tamaño es fijo o se busca máxima eficiencia. | Cuando el tamaño puede variar o se necesita manipular datos fácilmente. |

# Definicion Vector Tradicional y Vector de Clase

Un vector tradicional almacena datos simples como números o textos, mientras que un vector de clase guarda objetos con atributos y métodos. Esto permite manejar información más completa y organizada, como listas de alumnos o productos.

|  |  |
| --- | --- |
| **Concepto** | **Descripción** |
| **Vector tradicional** | Es un vector que almacena datos simples como enteros, decimales o cadenas, usados para listas básicas. |
| **Vector de clase** | Es un vector que almacena objetos de una clase creada por el programador, permitiendo trabajar con estructuras de datos más complejas. |

# Vector Tradicional y Vector de Clase en C++

|  |  |  |
| --- | --- | --- |
| **Característica** | **Vector tradicional** | **Vector de clase** |
| **Contenido almacenado** | Tipos de datos simples (int, float, string, etc.). | Objetos de una clase creada por el programador. |
| **Ejemplo de declaración** | vector<int> numeros; | vector<Alumno> alumnos; |
| **Tipo de datos** | Elementos primitivos. | Cada elemento es un objeto con atributos y métodos. |
| **Manipulación de datos** | Se manejan valores simples directamente. | Se manejan objetos, usando getters, setters y métodos propios. |
| **Complejidad** | Menor, ideal para listas simples de datos. | Mayor, permite trabajar con estructuras más completas. |
| **Ejemplo de uso** | Lista de números, nombres, etc. | Lista de alumnos, productos, empleados, etc. |
| **Métodos disponibles** | Los mismos que cualquier vector (push\_back, size, etc.). | Los mismos, pero aplicados a objetos de clase. |