PARADIGMAS DE PROGRAMACION

Trabajo Práctico N° 1 Fecha: 20/08/25

Tema: Paradigma Imperativo – Conceptos de C++

1) Implementar un ADT VECTOR (item) que le permita trabajar con arreglos del tipo item que pueden crecer dinámicamente si es necesario:

a. **Tipificación del ADT VECTOR**:
- *elementos*: puntero a elementos de tipo item.
- *max*: capacidad asignada al vector.

b. **Operaciones del ADT VECTOR**:

- *crearVector*: función que recibe un tamaño ***n*** (**parámetro por omisión**) y crea un vector con un arreglo dinámico de ***n*** elementos. Si se omite el parámetro ***n***, se crea un vector de tamaño 10. Si no se pudo realizar la reserva dinámica, la capacidad del vector queda en 0.

- *insertar*: función que recibe un vector, un entero positivo ***p*** y un elemento ***x*** del tipo item y agrega el elemento ***x*** al vector en la posición ***p***. Si la posición donde se desea ingresar el elemento supera la capacidad asignada al vector, éste debe crecer dinámicamente para poder almacenar el elemento en dicha posición.

- *capacidad*: función que recibe un vector y retorna la capacidad asignada al mismo.

- *elemento*: función que recibe un vector y una posición ***p*** y retorna una referencia al elemento del vector que se encuentra en la posición ***p***. De no existir la posición, retorna el elemento de la posición 0.

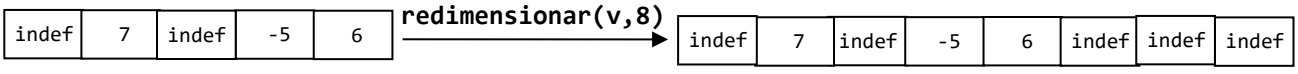
- *borrar*: **funciones homónimas** que reciben un vector y permiten
  - quitar el elemento dada una posición
  - quitar todos los elementos del vector.

  Las funciones solo asignan el valor indefinido en la posición cuyo valor se desea borrar.

- *destruir*: destruye el vector liberando la memoria asignada a sus elementos (**operador delete**).

Operaciones auxiliares:
- *reservarMemoria*: función booleana auxiliar que, dado un vector y un entero positivo ***n***, realiza una reserva dinámica de memoria (**operador new**) de tamaño ***n*** para almacenar los elementos del vector y los inicializa con valor indefinido del tipo item. Retorna verdadero si la operación se realizó con éxito.

- *redimensionar*: función booleana auxiliar que recibe un vector y un tamaño ***n*** y redimensiona el vector con la nueva capacidad. Retorna verdadero sólo si la operación se realizó con éxito.



2) **Sobrecargar el operador ==** como **una función del ADT Vector** de manera de que reciba dos vectores y retorne true si son iguales. Caso contrario retorna false.

3) **Sobrecargar los operadores de inserción y extracción de flujo** como **usuarios del ADT Vector** de manera de poder leer y escribir elementos del tipo Vector con el formato adecuado. Realice **pasaje de parámetros por referencia** utilizando referencias de objetos de C++.

4) Pruebe su implementación del ADT VECTOR con el programa de prueba dado en clases.

IMPORTANTE

En los casos que sea necesario las operaciones deben realizar control de índices del vector

FORMA DE TRABAJO

a. Escribir en un archivo **Vector.h** la definición del tipo de dato Vector y los prototipos de las funciones.
b. Escribir un archivo **Vector.cpp** con la definición de cada una de las operaciones del Vector.
c. Escribir un archivo **ioVector.h** con los operadores para inserción y extracción de flujo.