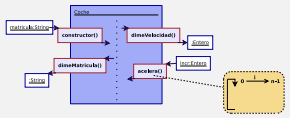
Grado Tecnologías Interactivas

Programación 1



Práctica 6. "Conjunto"

UNIVERSIDAD POLITECNICA DE VALENCIA

Escola Politècnica Superior de Gandia

DSIC

Departament de Sistemes Informàtics i Computació

Objetivos

- General: Diseño de clases.
- General: algoritmos sencillos sobre listas.
- C/C++: Implementación de una clase dado su diseño.
- C/C++: uso de los anteriores conceptos en este lenguaje.

¡ Atención !

▷ Se recuerda que las prácticas deben prepararse antes de acudir al aula informática, anotando en el enunciado las dudas que se tengan.

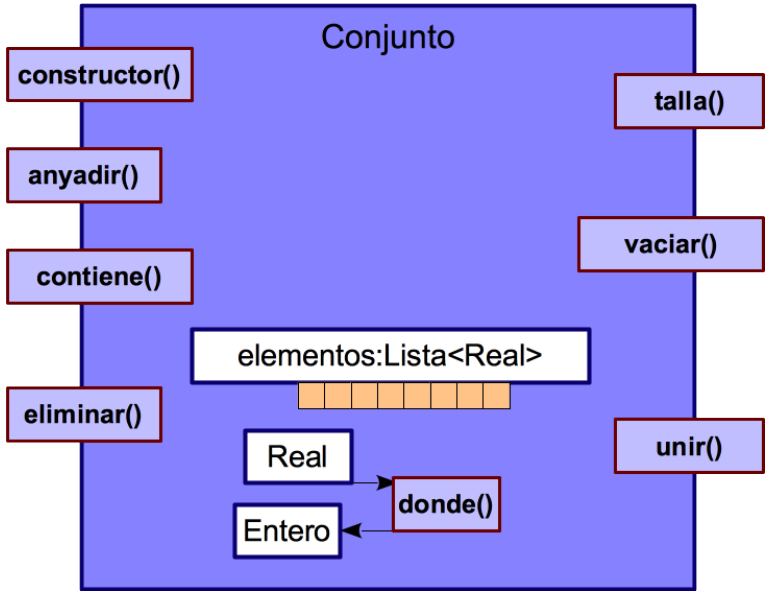
▷ Los diseños y algortimos que se piden en esta práctica deben escribirse en la libreta de apuntes para poder ser revisados.

▷ La realización de las prácticas es un trabajo individual y original. En caso de plagio se excluirá al alumno de la asignatura. Por tanto, es preferible presentar el trabajo realizado por uno mismo aunque éste tenga errores.

1

Diseño de la clase Conjunto

Hay que completar el diseño de la clase Conjunto



atendiendo a estas observaciones:

- Las operaciones públicas que tiene un conjunto y que deben implementarse son:
  - un constructor por defecto.
  - `talla()`: devuelve la cantidad de elementos que contiene el conjunto.
  - `anyadir()`: añade un nuevo elemento al conjunto.
  - `contiene()`: devuelve `true` si el elemento dado pertenece al conjunto.
  - `eliminar()`: elimina el elemento indicado.
  - `unir()`: une dos conjuntos, devolviendo uno nuevo.
  - `vaciar()`: vacía el conjunto.

- Los elementos del conjunto se guardan de forma privada en una lista de números reales llamada elementos.

2 Ejercicios

Atención: los programas deben estar correctamente comentados. Antes de cada función debe haber una comentario que resuma el diseño de la misma. En el código del cuerpo de la función hay que intercalar los pasos del algoritmo.
Es obligatorio utilizar GIT.

Hay que implementar la clase Conjunto dos veces:
1. Primera versión: utilizando un array básico.
2. Segunda versión: utilizando la clase de biblioteca vector.

Habrá un fichero mainPruebas.cpp donde se escribirán las pruebas automáticas de la clase conjunto. *Obligatoriamente, estas pruebas son las mismas independientemnte de cómo se haya implementado el conjunto*.

Pasos para implementación (ambas versiones):
1. Implementar el constructor.
2. Implementar vaciar().
3. Implementar talla().
4. Probar, de forma automática, los métodos anteriores.
5. Implementar donde(). (sólo para la versión con array básico).
6. Implementar contiene(), llamando donde() (sólo para la versión de array básico).
7. Implementar anyadir() llamando a contiene(). (Un conjunto no puede tener elementos duplicados).
8. Probar todos los métodos anteriores (incluyendo los primeros).
9. Implementar eliminar() llamando a donde().
10. Probar todos los métodos anteriores.

11. Implementar `unir()` llamando a `anyadir()`.
12. Probar todos los métodos anteriores.

Trabajo voluntario valorable

Implementar otros métodos para la clase `Conjunto` como intersección, diferecia, diferencia simétrica, etc.

3 Codigo Proporcionado

▷ Fichero Conjunto.h

```
// ------------------------------------------------
// Conjunto.h
// ------------------------------------------------
#ifndef CONJUNTO_YA_INCLUIDO
#define CONJUNTO_YA_INCLUIDO
#include <iostream>

// ------------------------------------------------
// ------------------------------------------------
class Conjunto {
 private:
  // completar: declaracion de variables privadas

 public:
  Conjunto();

  unsigned int talla() const;

  // completar perfil del resto de método
}; // class
// ------------------------------------------------
#endif
```

▷ Fichero Conjunto.cpp

```
// ------------------------------------------------
// Conjunto.cpp
// g++ -c Conjunto.cpp
// ------------------------------------------------
#include "Conjunto.h"
// ------------------------------------------------
// ------------------------------------------------
Conjunto::Conjunto()
{
  // completar
} // ()

// ------------------------------------------------
// ------------------------------------------------
unsigned int Conjunto::talla() const {
  // corregir
  return 0;
} // ()
```

▷ Fichero main.cpp

```
// ------------------------------------------------
// main.cpp
// g++  Conjunto.cpp main.cpp
// ------------------------------------------------
#include <iostream>
#include "Conjunto.h"

// ------------------------------------------------
// ------------------------------------------------
void probarTalla() {
  Conjunto c1;
  unsigned int a = c1.talla();
  if ( a != 0 ) {
       std::cout << " mmm, parece que no va bien talla()\n";
       return; // pues termino
  }
  // tal vez escribir más casos ...
  std::cout << " parece que talla() va bien \n";
} // ()

// ------------------------------------------------
// ------------------------------------------------
void probarOtraCosa() {
  std::cout << " parece que esta prueba funciona ... ";
  std::cout << " pero es porque no he probado nada aún :-) \n";
} // ()

// ------------------------------------------------
// ------------------------------------------------
int main() {
  probarTalla();
  probarOtraCosa();
} // ()
```

14 noviembre 2018