6 STL iteradores y algoritmos

Elaborado por: Ukranio Coronilla

Además del contenedor vector visto en la sección pasada, existen: arreglo, lista, pila, cola y otros más que se pueden consultar en: http://www.cplusplus.com/reference/stl/

Un elemento útil en las STL son los **iteradores**, los cuales son objetos que permiten navegar dentro de una clase Template o contenedor de datos.

Podríamos ver al iterador como una especie de "apuntador" que funciona para acceder a los elementos de un Template. Así como existen apuntadores distintos para cada tipo de dato, existe un iterador para cada Template. Los operadores sobre un iterador son ++, --, ==,!= y * para incrementar, decrementar, ver si son iguales, diferentes u obtener el dato al que "apuntan", respectivamente.

Continuando el ejemplo de la sección anterior, es posible declarar un iterador `i` para el vector `v` de enteros como:

```
std::vector<int>::iterator i
```

La instrucción `v.begin()` devuelve un "apuntador" al primer elemento del vector `v`. Mientras que la instrucción `v.end()` devuelve un "apuntador" al fin de vector `v`. De modo que podríamos obtener todos los elementos del vector `v` mediante la siguiente instrucción.

```
for (i = v.begin( ); i != v.end( ); i++)
   cout << *i << endl;
```

Debido a que `v.end()` no devuelve el "apuntador" al último elemento. Para recorrer el vector en sentido inversø (de fin a principio), se debe utilizar el iterador reversa siguiente:

```
vector<int>::reverse_iterator ir;
for (ir = v.rbegin( ); ir != v.rend( ); ir++)
cout << *ir <<  endl;
```

Ejercicio 1: Después de elaborar el ejercicio 2 de la práctica 5 , en la función principal instancie un objeto `PoligonoIrreg` al cual posea n vértices, todos ellos con valores reales aleatorios tanto positivos como negativos en sus coordenadas comprendidos entre -100 y 100, y con tres cifras decimales distintos de cero. Posteriormente modifique el método `imprimeVertices` de la clase `PoligonoIrreg` para imprimir las coordenadas y la magnitud(distancia de la coordenada al origen) de cada elemento del vector, pero haciendo uso de un iterador para recorrerlo. Para mejorar el desempeño es necesario añadir un dato miembro para almacenar la magnitud.

Algoritmos genéricos

Así como hay clases template, también existen funciones template conocidas como algoritmos genéricos. Existe una gran cantidad de algoritmos especificados en:

http://www.cplusplus.com/reference/algorithm/

Es normal que se especifique en la referencia para cada función template, el algoritmo que se ha implementado, así como la eficiencia del algoritmo. Para utilizar un algoritmo genérico se debe incluir la librería:

```
#include <algorithm>
```

Ejercicio 2: Utilice el algoritmo generico `sort`, cuyo ejemplo de implementación se muestra en:

http://www.cplusplus.com/reference/algorithm/sort/

para ordenar de manera ascendente el vector de coordenadas del objeto `PoligonoIrreg` . Para lograrlo implemente el método `ordenaA` dentro de la clase.

Proyecto 1 para subir al MOODLE (continúa):

7.-Elabore un programa para encontrar los números primos menores al valor N que introduzca el usuario, mediante el algoritmo de la Criba de Eratóstenes:

http://es.wikipedia.org/wiki/Criba_de_Erat%C3%B3stenes

Use un template vector para almacenar objetos de una clase, cuyos miembros son los números enteros y un dato tipo bool, para descartar al objeto en el caso de que no sea primo (originalmente todos los elementos del vector tienen un valor true en este dato miembro).