Programación Orientada a Objetos

Unidad 8: STL

Teoría 11- 10/11/2011 - Pablo Novara

STL: Standart Template Library

La STL esta compuesta por:

- **Contenedores**: clases para almacenar muchos datos de un mismo tipo:
  - *vector, list, map, queue, stack, set, ...*
- **Iteradores**: clases que permiten señalar datos o posiciones de los contenedores
- **Algoritmos**: funciones para operar sobre los elementos de los contenedores:
  - *copy, replace, remove, sort, find, ...*

STL: vector

```
vector<int> v; // crea un vector de enteros vacío

v.resize(3); // el vector ahora tendrá 3 elementos
v[0]=1; v[1]=5; v[2]=7; // asigna valores a esos 3 e.

v.push_back(10); // agrega un 10 al final
v.push_back(20); // agrega un 20 al final
v.push_front(5); // agrega un 5 al principio
v.insert(v.begin()+2,7); // agrega un 7 en la 3ra pos.
v.erase(v.begin()+2); // elimina el 7 (3ra pos)
v.pop_back(); // elimina el último

for (unsignet int i=0;i<v.size();i++)
    cout<<v[i]<<" ";
```

STL: list

```
list<int> l; // crea un vector de enteros vacio
list<int>::iterator it; // iterador

v.push_back(10); // agrega un 10 al final
v.push_back(20); // agrega un 20 al final
v.push_front(5); // agrega un 5 al principio

it=l.begin(); l++; l++; // apunta con l al 3er elemento
it=l.insert(it,7); // agrega un 7 en la 3ra pos.
it=l.erase(it); // elimina el 7 (3ra pos.)

for (it=l.begin();it!=l.end();it++)
    cout<<*it<<" ";
```

STL: map

```
map<string,int> calif;

calif["Fundamentos de Programación"]=9;
calif["Programación Orientada a Objetos"]=7;
calif["Matemática Básica"]=8;
calif["Física I"]=7;
calif["Álgebra"]=7;
calif["Ingeniería de Software"]=7;

string materia;
getline(cin,materia);

if (calif.find(materia)==calif.end())
    cout<<"No se registra nota para esa materia.";
else
    cout<<"La nota es: "<<calif[materia];
```

STL: <algoritm>

```
bool es_par(int x) { return x%2==0; }


list<int> l;
list<int>::iterator it;


// busca el valor 10 en la lista
it = find( l.begin() , l.end() , 10 );

// busca el primer numero par
it = find_if( l.begin() , l.end() , es_par );
```

STL: <algoritm>

```
// elimina todas las veces que aparece el valor 10
remove( l.begin() , l.end() , 10 );

// elimina todos los números pares de la lista
remove_if( l.begin() , l.end() , es_par );


// reemplaza todos los 15 por 30
replace( l.begin() , l.end() , 15 , 30 );

// reemplaza todos los pares por 0
replace_if( l.begin() , l.end() , es_par , 0 );
```

STL: <algoritm>

```
// cuenta cuantas veces aparece el valor 10
int n1 = count( l.begin() , l.end() , 10 );

// cuenta cuantos números pares hay en la lista
int n2 = count_if( l.begin() , l.end() , es_par );


// suma todo el contenido de la lista
n = accumulate( l.begin() , l.end() , 0 );


// elimina los repetidos de la lista
it = unique( l.begin() , l.end() ); // los coloca al final
l.erase( it , l.end() ); // despues hay que borrarlos
```

STL: <algoritm>

```
bool min_val_abs(const int &x, const int &y)
    { return abs(x)<abs(y); } // simil operator<

vector<int> v;  list<int> l;

// ordena el vector
sort( v.begin() , v.end() );

// ordena el vector por valor absoluto
sort( v.begin() , v.end() , min_val_abs );

// ordena la lista
l.sort(); // orden comun
l.sort( min_val_abs ); // orden por val abs
```

STL: <algoritm>

```
// busca el maximo y minimo
it = max_element( l.begin() , l.end() );
it = min_element( l.begin() , l.end() );

// busca el maximo y minimo valor absoluto
it = max_element( l.begin() , l.end() , min_val_abs );
it = min_element( l.begin() , l.end() , min_val_abs );


// desordena el vector
random_shuffle( v.begin() , v.end() );

// invierte el orden de la lista
reverse( l.begin() , l.end() );
```

STL: <algoritm>

```
// copia los elementos de la lista en el vector
v.resize(l.size());//hay que hacer lugar, copy no inserta!
copy( l.begin() , l.end() , v.begin() );


// muestra la lista en pantalla
copy( l.begin() , l.end() ,
        ostream_iterator<int>(cout," ") );

// lee la lista de un archivo
copy( istream_iterator<int>(archivo) ,
        istream_iterator<int>() ,
        back_inserter(l) );
```

STL: <algoritm>

...y más:

- search, transform, for_each, generate, rotate, merge, fill, ...

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