Estructuras de datos básicas de la STL de C++

Emanuel Lupi

Universidad Nacional de Córdoba
FaMAF

17 de Abril de 2018

Motivación

- Abstracción, Claridad y flexibilidad
- Bugs-Free
- Eficiencia

Que son los templates

Para que son?

Nos permite escribir código genérico que puede ser usado con varios tipos de datos.
Nos permiten parametrizar estas clases para adaptarlas a cualquier tipo de dato

De que manera?

Poniendo el tipo entre <>

ejemplo

```
vector<int> A;
vector<vector<pair<int, string>>> K;
queue<vector<int>> Q;
queue<struct nodo> bfs_Q;
```

Pair

Estructura de datos simple que nos permite crear un par de tipos a determinar. Principales operaciones

- Operador ==

```
pair<int, int> p;
pair<int, string> k;
pair<vector<int>, string> k;
```

Vector

Estructura de datos que nos permite tener elementos en una memoria contigua que se genera dinamicamente en O(1) *amortizado*

Principales operaciones / métodos

- size
- Operador []
- front
- back
- push_back
- pop_back
- insert
- erase
- swap
- clear

Ejemplo Vector

```
vector<int> A;
vector<int> B;
A.push_back(5);
B.swap(A);
cout << B.size() << ", " << B.back == B[0] << endl;

// esto va a mostrar "1, true"
```

Set

Son contenedores que guardan elementos no repetidos en un orden especifico.
Principales operaciones / métodos

- size
- insert
- erase
- swap
- find
- lower_bound
- upper_bound
- clear

Ejemplo Set

```
set<int> ms;
set<int>::iterator itlow,itup;

for (int i=1; i<10; i++) ms.insert(i*10); // 10 20 30 40 50 60 70 80 90

itlow=ms.lower_bound (30);                //       ^
itup=ms.upper_bound (60);                 //                   ^

ms.erase(itlow,itup);                     // 10 20 70 80 90
```

Map

Relaciona una clave con un valor.
Principales operaciones / métodos

- size
- Operador []
- insert
- erase
- swap
- find
- lower_bound
- upper_bound
- clear

Ejemplo Map

```
map<char,int> fm;
map<char,int>::iterator it;

fm['a']=50;
fm['b']=100;
fm['c']=150;
fm['d']=200;          // {'a': 50, 'b': 100, 'c': 150, 'd': 200}

it = fm.find('b');  // iterador a la posicion de b
if (it != fm.end())
  fm.erase (it);  // {'a': 50, 'c': 150, 'd': 200}

cout << fm['d'] << endl;              // 200
```

Stack o pila

Estructura LIFO.
Principales operaciones / métodos

- size
- top
- empty
- push
- pop
- swap
- clear

Ejemplo Stack

```
stack<int> st;

for (int i=0; i<7; ++i)
  st.push(i);

while (!st.empty())
{
    cout << ' ' << st.top();
    st.pop();
}
cout << '\n';
// 6 5 4 3 2 1 0
```

Queue o fila/cola

Estructura FIFO.
Principales operaciones / métodos

- size
- front
- back
- push
- pop
- swap

Ejemplo Queue

```
queue<int> Q;
int myint;

for(int i=0 ; i<7 ; i++)
  Q.push (i);

while (Q.size())
{
   cout << ' ' << Q.front();
  Q.pop();
}
cout << '\n';
```

Una función de gran utilidad

upper_bound, *lower_bound* sobre vectores (si estos están ordenados)