Programação Orientada a Objetos

Prof. Hugo Marcondes
hugo.marcondes@ifsc.edu.br

Aula 04

Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina

STL - Standard Template Library

- STL
  - Conjunto de templates de classes com implementações de estrutura de dados e funcionalidades comum em programação
- Mas antes de entender os “componentes” da STL em si, vamos entender o que é um template de classe !

C++ Templates

- Templates C++ é uma poderosa construção da linguagem que permite a definição de uma classe genérica (um template)
  - Principalmente utilizado na generalização de tipos
- Quando a classe é utilizada (através da declaração/ criação de um objeto) é necessário especificar quais são os tipos de dados que irão ser “preenchidos” na classe.
- Desta forma, classes genéricas são “especializadas” pelo compilador C++ no momento em que o código é gerado.

Template Class

- Uma classe genérica é definida através da declaração “template< typename T >”, antes de sua declaração, conforme abaixo:

```
template<typename T>
class MyClass {
  T _attribute;

public:
  void set_attribute(T parameter);
  T get_attribute();

};
```

STL

- Conjunto de componentes para tratar “problemas” comuns de programação

- “Containers”
  - Componentes utilizados para armazenar um conjunto de objetos (mesma classe)
  - Sequenciais
    - vector
    - deque
    - list
  - Associativos
    - set
    - map

STL

- “Interators”
  - Fornecem uma forma comum para acessar os objetos dentro dos “containers”
  - Cada container define o seu interator

- “Algorithm”
  - Funções genéricas definidas na STL para interargir com os objetos de um container
  - Utilizam os interators para acessar os containers de forma unificada
    - Ex. busca(find), ordenação(sort), etc

Vector

- Suporta o acesso randômico a seus componentes
- Tempo de inserção e remoção constantes no final do vetor
- Tempo linear de inserção/remoção no início e no meio do vetor
- Número de elementos pode variar e o gerenciamento da memória é automático

- Referência completa
  http://www.cplusplus.com/reference/vector/vector/

Vector - Exemplo Hello World :D

```
#include <vector>
#include <iostream>
using namespace std;

char* szHW = "Hello World";

int main(int argc, char* argv[])
{
  vector<char> vec;
  vector<char>::iterator vi;

  char* cptr = szHW;
  while (*cptr != '\0') {
     vec.push_back(*cptr);
     cptr++;
  }
  for (vi=vec.begin(); vi!=vec.end(); vi++) {
      cout << *vi;
  }
  cout << endl;
  return 0;
}
```

Containers Sequenciais

- deque
  - Double ended queue

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

- list
  - Lista (Lista duplamente encadeada)

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

Vamos praticar !

- Acessando a documentação do site http://www.cplusplus.com/, implemente um programa com a seguinte rotina:
  - Solicita para o usuário um conjunto de números inteiros (quantidade arbitrária)
  - Imprime este conjunto na ordem em que o usuário digitou
  - Apresenta qual é o menor número digitado
  - Apresenta qual é o maior número digitado
  - Imprime os números digitados em ordem crescente

Containers Associativos

- Set
  - Coleção de dados ordenados em uma estrutura de árvore binária balanceada
    - Facilita os algoritmos de busca
- Map
  - Coleção de dados associados através do uso de um par de chave e valor
    - Set na realidade é um tipo de Map, onde chave == valor

Exemplo do uso de Set

```
#include <string>
#include <set>
#include <iostream>
using namespace std;

int main(int argc, char* argv[])
{
  set <string> strset;
  set <string>::iterator si;
  strset.insert("laranja");
  strset.insert("maça");
  strset.insert("limão");
  strset.insert("banana");
  strset.insert("uva");
  strset.insert("uva");
   // This one overwrites the previous occurrence
  for (si=strset.begin(); si!=strset.end(); si++) {
      cout << *si << " ";
  }
  cout << endl;
  return 0;
}
```

Exemplo do uso de Map


```
#include <string.h>
#include <iostream>
#include <map>
#include <utility>

using namespace std;

int main()
{
   map<int, string> Estudante;
   map<int,string>::iterator ii;

   Estudante[5234] = "Joao da Silva";
   Estudante[3374] = "Carlos Pereira";
   Estudante[1923] = "Bruna da Silva";
   Estudante[7582] = "Jonas Macedo";
   Estudante[5328] = "Vitor Machado";
   cout << "Estudante[3374]=" << Estudante[3374] << endl << endl;
   cout << "Map size: " << Estudante.size() << endl;
   for(ii=Estudante.begin(); ii!=Estudante.end(); ++ii) {
        cout << (*ii).first << ": " << (*ii).second << endl;
   }
}
```


Referências STL

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

http://www.codeproject.com/Articles/6513/Practical-Guide-to-STL

http://cs.brown.edu/~jak/proglang/cpp/stltut/tut.html

http://www.yolinux.com/TUTORIALS/LinuxTutorialC++STL.html