Conjunto.hpp - (~/Documentos/atual_2023-1/lp/aulas/23_Iteradores/execucao) - GVIM

Arquivo Editar Ferramentas Sintaxe Buffers Janela Ajuda

```
#include <exception>
#include <iostream>

using namespace std;

#include "Conjunto.hpp"

template <typename T>
void imprimir (Conjunto<T> &C)
  {
   cout << "{";

// for (auto it = C.inicio(), fim = C.fim(); it != fim; it++)
// for (auto it = C.begin(), fim = C.end(); it != fim; it++)
//   {
//    cout << ' ' << *it;
//   }
   for (auto elem: C) cout << ' ' << elem;

   cout << " } (nova implementação)\n";
  }

int main ()
  {
   try
     {
      Conjunto<int> C;

      for (int i = 0; i < 5; ++i) C.inserir_novo(i);

      imprimir(C);
     }
   catch (const exception &e)
     {
      cerr << "Exceção: " << e.what() << '\n';  return 1;
     }
  }
```

iteradores.cpp 27,1 Tudo

```
#include <algorithm>
#include <iostream>
#include <ostream>
#include <vector>

using namespace std;

struct Par
  {
   int x,y;

   // a.operator<(b) == operator<(a,b)

   bool operator< (Par b)
     {
      return x < b.x or (x == b.x and y < b.y);
     }
  };

// bool operator< (Par a, Par b)
//   {
//    return a.x < b.x or (a.x == b.x and a.y < b.y);
//   }

ostream& operator<< (ostream &saida, Par p)
  {
   saida << "(" << p.x << "," << p.y << ")";

   return saida;
  }

template <typename T>
void imprimir (vector<T> &v)
  {
   for (auto it = v.begin(), fim = v.end(); it != fim; ++it)
     cout << *it << '\n';

   cout << '\n';
  }

int main ()
  {
   vector<int> vi { 0, -10, 20, 5 };  imprimir(vi);

   sort(vi.begin(), vi.end());        imprimir(vi);

   vector<Par> vp { {1,2}, {-1,2}, {0,3}, {0,2} };

   imprimir(vp);

   sort(vp.begin(), vp.end());        imprimir(vp);
  }
```

operadores_binarios.cpp 10,1 Tudo

```
#ifndef CONJUNTO_HPP
#define CONJUNTO_HPP

// ------------------------------------------------

template <typename T>
class Conjunto
  {
   struct Noh { T elem;  Noh *ant, *prox; };

   Noh sent;

   public:

   class Iterador
     {
      Noh *noh;

      public:

      Iterador (Noh *n) : noh{n} { }

      T operator* () { return noh->elem; }  // (*noh).elem

      bool operator!= (Iterador it)
        {
         return noh != it.noh;
        }

      Iterador& operator++ ()     // ++ Pré-fixo.
        {
         noh = noh->prox;  return *this;
        }

      Iterador operator++ (int)   // ++ Pós-fixo.
        {
         Noh *n = noh;  noh = noh->prox;  return {n};
        }
     };

   Conjunto()
     {
      sent.ant = sent.prox = &sent;
     }

   ~Conjunto()
     {
      for (Noh *n = sent.prox, *prox; n != &sent; n = prox)
        {
         prox = n->prox;  delete n;
        }
     }

   Iterador /* fim */ end () { return {&sent}; }
```

Conjunto.hpp 41,1 Topo

```
   Iterador /* inicio */ begin ()
     {
      return {sent.prox};
     }

   void inserir_novo (T e)
     {
      Noh *n = new Noh {e, &sent, sent.prox};
      n->ant->prox = n;
      n->prox->ant = n;
     }
  };

// ------------------------------------------------

#endif
```

Conjunto.hpp 69,53 Fim