



INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
PARAÍBA
Campus João Pessoa



Projeto Olímpico de Programação

[C++] STL: Container Set

STL: Container Set

- Coleção ordenada de “key” (sem duplicidade);
- Permite acesso sequencial aos elementos armazenados;
- **Complexidade das operações básicas:**

Operação	Complexidade
Inserção e remoção no início, meio e final	$O(1)$
Busca	$O(n)$

- Iteradores são válidos após alteração na lista;
- Inserção no código:

```
#include <set>
```

- Declaração:

```
set<int> numeros;
```

STL: Container Set

insert

```
#include <iostream>
#include <set>
#include <list>

using namespace std;

int main(){
    list<int> lista;
    set<int> dados;
    set<int>::iterator it;

    lista.push_back(10);
    lista.push_back(10);
    lista.push_back(20);
    lista.push_back(20);
    lista.push_back(30);
    lista.push_back(30);
    lista.push_back(10);
    lista.push_back(10);

    dados.insert(lista.begin(), lista.end());
    dados.insert(dados.begin(), 10);
    dados.insert(10);

    for (it = dados.begin(); it != dados.end(); ++it){
        cout << *it << endl;
    }

    return 0;
}
```



STL: Container Set

clear e erase

```
#include <iostream>
#include <set>
#include <list>

using namespace std;

void print_set(set<int> s);

int main(){
    set<int> dados;

    for (int i = 1; i <= 8; ++i){
        dados.insert(i * 10);
    }
    print_set(dados);

    dados.erase(60);
    print_set(dados);

    dados.erase(dados.begin(), +++dados.begin());
    print_set(dados);

    dados.erase(dados.begin());
    print_set(dados);

    return 0;
}

void print_set(set<int> s){
    set<int>::iterator it;

    for (it = s.begin(); it != s.end(); ++it){
        cout << *it << " ";
    }
    cout << endl;
}
```



10 20 30 40 50 60 70 80
10 20 30 40 50 70 80
30 40 50 70 80
40 50 70 80

STL: Container Set

Operadores Relacionais

```
#include <iostream>
#include <set>
#include <list>

using namespace std;

void print_set(set<int> s);

int main(){
    set<int> dados, copia;

    for (int i = 1; i <= 8; ++i){
        dados.insert(i * 10);
        copia.insert(i * 10);
    }
    print_set(dados);
    print_set(copia);

    if (dados == copia)
        cout << "iguais\n";
    else
        cout << "diferentes\n";

    return 0;
}

void print_set(set<int> s){
    set<int>::iterator it;

    for (it = s.begin(); it != s.end(); ++it){
        cout << *it << " ";
    }
    cout << endl;
}
```



STL: Container Set

find

```
#include <iostream>
#include <set>
#include <list>

using namespace std;

void print_set(set<int> s);

int main(){
    set<int> dados;
    set<int>::iterator it;

    for (int i = 1; i <= 8; ++i){
        dados.insert(i * 10);
    }

    for (it = dados.begin(); it != dados.end(); ++it){
        cout << *it << " ";
    }
    cout << endl;

    it = dados.find(60);
    if (it != dados.end()){
        cout << "Encontrei: " << *it << endl;
    }else{
        cout << "Nao encontrei" << endl;
    }

    return 0;
}
```



10 20 30 40 50 60 70 80
Encontrei: 60

STL: Container Set

Desafio!

Escrever um programa, em C++, seis valores $[1,60]$ aleatórios e distintos.

Exibir os valores lidos.

STL: Container Set

Desafio! Resposta

