



Programação Nível Intermediário

Universidade Federal da Bahia
Instituto de Computação
Departamento de Ciência da Computação

AULA 1 - VECTOR, STRUCT, PAIR

Vetores - vector

```
#include <iostream>
```

```
#include <vector>
```

```
using namespace std; // vector -> std
```

```
int main() {
```

```
    int v1[100]; // vetor tradicional com 100
```

```
    elementos
```

```
    vector<int> v2(100); // vector com 100 elementos pré-alocados
```

```
    for(int i=0; i < 100; i++) {
```

```
        cin >> v1[i] >> v2[i];
```

```
    }
```

```
// por que usar vector?
```

Vetores - push_back

```
#include <iostream>
```

```
#include <vector>
```

```
using namespace std;
```

```
int main() {
```

```
    vector<int> v;
```

```
    for(int i=0; i < 1000; i++) {
```

```
        int j;
```

```
        cin >> j;
```

```
        v.push_back(j); // função push_back adiciona elemento no final do
```

```
vetor
```

```
    }
```

```
    // e aloca mais espaço caso necessário
```

Vetores - size

```
#include <iostream>
#include <vector>
using namespace std;
int main() {
    vector<int> v;
    int x;
    while(cin >> x && x >= 0)
        v.push_back(x);
    for(int i=0; i < v.size(); i++) // função size retorna a qtdade de elementos do
vetor
        cout << v[i] << endl; // o acesso ao vetor é realizado normalmente
```

Vetores - clear

...

```
int main() {  
    vector<int> v;  
    int x;  
    while(cin >> x && x >= 0)  
        v.push_back(x);  
    for(int i=0; i < v.size(); i++)  
        cout << v[i] << endl;
```

```
    v.clear();
```

os elementos do vetor

```
    while(cin >> x && x >= 0)  
        v.push_back(x);
```

// função clear remove todos

Vetores - iterator

```
#include <iterator>
```

```
...
```

```
int main() {  
    vector<int> v;
```

```
...
```

```
// iterator é um ponteiro para elementos
```

```
for(vector<int>::iterator it = v.begin();    it != v.end();    it++)  
    cout << *it << endl; // não imprima o iterator, e sim o valor apontado por ele!  
}
```

Vetores - erase

...

```
int main() {  
    vector<int> v;  
    int x;  
    while(cin >> x && x >= 0)  
        v.push_back(x);  
  
    v.erase(v.begin()); // apaga o primeiro elemento  
    v.erase(v.begin()+1); // apaga o segundo elemento  
    v.erase(v.begin()+2, v.end()); // apaga todos os elementos exceto os dois  
    primeiros  
}
```


Vector e Struct

Vector e struct

```
#include <iostream>
#include <vector>
#include <string>

using namespace std;
struct pessoa {
    int id;
    string nome;
};

int main() {
    vector<pessoa> v;
    pessoa eu;

    for (int i=0; i<100 ; i++){
        cin >> eu.id >> eu.nome;
        v.push_back(eu);
    }

    for (int i=0; i<100 ; i++)
        cout << v[i].id << " " << v[i].nome << endl;
}
```

Vetores - vector

Saiba mais em:

<http://www.cplusplus.com/reference/vector/vector/>

PARES DE DADOS

Pares de dados - pair

```
#include <iostream>
```

```
#include <utility>
```

```
#include <string>
```

```
using namespace std; // pair -> std
```

```
int main() {
```

```
    pair<string,double> p; // tipos dos dados do par
```

```
    p.first = "pi"; // acessa primeira parte do par
```

```
    p.second = 3.14159; // acessa a segunda parte do par
```

```
}
```

Vector e pair

```
#include <iostream>
#include <vector>
#include <utility>
#include <string>

int main() {
    vector<pair<int,string>> v;
    pair <int,string> eu;
    for (int i=0; i<100 ; i++){
        cin >> eu.first >> eu.second;
        v.push_back(eu);
    }

    for (int i=0; i<100 ; i++)
        cout << v[i].first << " " << v[i].second << endl;
}
```

Pares de dados - pair

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

int main() {
    pair<int,int> esse = make_pair(10,20);
    pair<int, pair<int, pair<int, int>>>> p;
    p.first = 5;
    p.second.first = 10;
    p.second.second.first = 22;
    p.second.second.second = 30;

    pair<int, pair<int, pair<int, int>>> p2 = {5,{10,{21,30}}};

    if (p2 < p) cout << "Esse P2" << endl;
    else cout << "Esse p1" << endl;

    pair<int, pair<int, pair<int, int>>> p3 = make_pair(3,make_pair(2,make_pair(40,50)));
    cout << p3.second.second.first << endl;

}
```

Pares de dados - pair

// comparação entre pares utiliza a primeira parte e desempata pela segunda, como ilustrado abaixo:

```
bool operator<(pair<string,double> a, pair<string,double> b) {  
    return a.first < b.first || a.first == b.first && a.second < b.second;  
}
```

// você pode redefinir um operador de comparação da seguinte maneira:

```
bool operator<(pair<string,double> a, pair<string,double> b) {  
    return a.second < b.second || a.second == b.second && a.first < b.first;  
}
```


Pares de dados - pair

Saiba mais em:

<http://www.cplusplus.com/reference/utility/pair/>