Vector C++ (STL)

Slide 1

Pertemuan 11 - Dasar Pemrograman

- **Tujuan Pembelajaran:**
  - Memahami limitasi array tradisional
  - Mengenal struktur data dinamis modern: Vector
  - Mampu menggunakan operasi dasar vector
  - Menerapkan vector dalam problem solving
- **Topik Bahasan:**
  - Masalah dengan array fixed-size
  - Pengenalan vector dari STL (Standard Template Library)
  - Operasi-operasi dasar vector

- Perbandingan array vs vector

Masalah dengan Array Tradisional

Slide 2

- **Ukuran Fixed (Tidak Fleksibel)**
  - Harus tentukan ukuran di awal: `int arr[10];`
  - Tidak bisa bertambah atau berkurang saat runtime
- **Manajemen Manual yang Ribet**
  - Tracking ukuran array sendiri dengan variabel counter
  - Rentan error: akses index di luar batas
  - Tidak ada built-in function untuk operasi umum
- **Contoh:**

```
int nilai[5];
int count = 0;

// Input nilai siswa
```

```
for(int i = 0; i < 10; i++) {
    cin >> nilai[i];
}
```

Solusi: Vector

Slide 3

- **Apa itu Vector?**
  - Struktur data array dinamis dari C++ Standard Template Library
  - Ukuran bisa bertambah/berkurang otomatis saat runtime
  - Banyak built-in function untuk operasi umum
- **Setup Penggunaan:**

```
#include <iostream>
#include <vector>  // Wajib include saat pakai vector
using namespace std;

int main() {
    vector<int> angka;     // Vector untuk menyimpan integer
    vector<string> nama;   // Vector untuk menyimpan string
    vector<double> nilai;  // Vector untuk menyimpan double

    return 0;
```

```
}
```

- **Deklarasi dengan Nilai Awal:**

```
vector<int> v1;              // Kosong
vector<int> v2(5);           // 5 elemen, nilai default 0
vector<int> v3(5, 10);       // 5 elemen, semua nilai 10
vector<int> v4 = {1,2,3,4};  // Inisialisasi dengan nilai
```

Operasi: Size, Menambah & Menghapus Elemen

Slide 4

- **size() - Mendapatkan Ukuran Vector**

```
vector<string> buah = {"Apel", "Jeruk", "Mangga"};

cout << "Jumlah buah: " << buah.size();  // Output: 3

// Praktis untuk looping
for(int i = 0; i < buah.size(); i++) {
    cout << buah[i] << endl;
}
```

- **push_back() - Tambah Elemen di Akhir**

```
vector<int> angka;

angka.push_back(10);   // [10]
angka.push_back(20);   // [10, 20]
angka.push_back(30);   // [10, 20, 30]

cout << "Ukuran: " << angka.size();  // Output: 3
```

- **insert(position, value) - Tambah Elemen di posisi tertentu**

```
v.insert(v.begin(), 5);     // Tambah di awal: [5, 10, 20]
v.insert(v.begin() + 1, 7); // Tambah di index 1: [5, 7, 10, 20]
```

Operasi: Hapus/Clear

Slide 5

- **pop_back() - Hapus Elemen Terakhir**

```
vector<int> nilai = {5, 10, 15, 20};

nilai.pop_back();   // [5, 10, 15]
nilai.pop_back();   // [5, 10]

cout << "Ukuran: " << nilai.size();  // Output: 2
```

- **erase() - Hapus Elemen Tertentu**

```
v.erase(v.begin());   // Hapus elemen pertama
v.erase(v.begin() + 2); // Hapus elemen index 2
```

- **clear() - Kosongkan Vector**

```
vector<int> data = {1, 2, 3, 4, 5};
cout << "Sebelum: " << data.size();  // Output: 5
```

```
data.clear();  // Hapus semua elemen
cout << "Sesudah: " << data.size();  // Output: 0
```

- **empty() - Cek Vector Kosong**

```
vector<int> v;

if(v.empty()) {
    cout << "Vector kosong";  // Akan tercetak
}
```

Operasi: Akses Elemen

Slide 6

- **Menggunakan Operator [] (Bracket)**

```
vector<int> nilai = {85, 90, 75, 88};

cout << nilai[0];  // Output: 85 (elemen pertama)
cout << nilai[2];  // Output: 75 (elemen ketiga)

nilai[1] = 95;     // Ubah elemen kedua jadi 95
```

- **Menggunakan .at()**

```
vector<int> skor = {100, 85, 92};

cout << skor.at(0);  // Output: 100
```

- **Cara Lain**

```
v.front();      // Elemen pertama
v.back();       // Elemen terakhir
```

Operasi: Pencarian elemen

Slide 7

- **Menggunakan fungsi FIND**

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

vector<int> vec = {1, 2, 3, 4, 5};
int nilai = 3;

// Mencari nilai dalam vector
auto it = find(vec.begin(), vec.end(), nilai);
```

- **Mencari frekuensi kemunculan elemen tertentu**

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

vector<int> vec = {1, 2, 3, 4, 5};
int nilai = 3;
```

```
cout << count(vec.begin(), vec.end(), nilai);
```

Operasi: Mengurutkan elemen

- **Menggunakan fungsi SORT**

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

vector<int> angka = {5, 3, 7, 1, 9, 2};

// Urutan naik (ascending)
sort(angka.begin(), angka.end());
// Hasil: {1, 2, 3, 5, 7, 9}

// Urutan turun (descending)
sort(angka.begin(), angka.end(), greater<int>());
```

Contoh Praktis: Input Data Dinamis

Slide 9

- **Kasus: Input Nilai Siswa (Jumlah Tidak Diketahui)**

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

int main() {
    vector<int> nilai;
    int input;

    cout << "Masukkan nilai (0 untuk selesai):\n";

    while(true) {
        cin >> input;
        if(input == 0) break;

        nilai.push_back(input);
    }

    // Hitung rata-rata
    int total = 0;
    for(int i = 0; i < nilai.size(); i++) {
        total += nilai[i];
```

```
    }

    cout << "Jumlah nilai: " << nilai.size() << endl;
    cout << "Rata-rata: " << total / nilai.size() << endl;

    return 0;
}
```