TIN ĐẠI CƯƠNG

Bài 9: TEMPLATE, VECTOR

Trần Thị Ngân

Bộ môn Công nghệ phần mềm, Khoa CNTT

Trường đại học Thủy Lợi

Nội dung chính

1. Khuôn mẫu (template)
2. Kiểu dữ liệu vector
3. Khai báo vector
4. Các phương thức
5. Bài tập

Khuôn mẫu (template)

Để tìm số lớn nhất của hai số nguyên, ta viết hàm sau :

```
int timmax(int a, int b)
{
    if (a > b) return a; else return b;
}
```

Tuy nhiên hàm này không áp dụng được cho hai số thực. Ta phải viết thuật toán này cho kiểu dữ liệu thực.

```
double timmax(double a, double b)
{
    if (a > b) return a; else return b;
}
```

→ Nhiều thuật toán có tính tổng quát, có thể áp dụng cho nhiều kiểu dữ liệu khác nhau.

Template

- Ngôn ngữ C++ cho phép chúng ta "tổng quát hoá" các đoạn code tương tự nhau này bằng cách dùng template
- Ví dụ : thuật toán tìm phần tử lớn nhất của hai phần tử

```
template <class T> T timmax(T a, T b)
{
    if (a > b) return a; else return b;
}
```

- Máy tính sẽ tự thay thế kiểu dữ liệu thích hợp cho T trong từng tình huống

```
cout << timmax(5, 10) << endl;
cout << timmax(3.7, 8.4) << endl;
cout << timmax('A', 'B') << endl;
```

Lớp (class)

- Lớp là sự mở rộng của cấu trúc dữ liệu. Lớp không chỉ lưu trữ dữ liệu mà cả các hàm (phương thức)
- Khai báo một đối tượng x thuộc lớp T :

```
T x(danh_sach_tham_so);
```

- Các phương thức của lớp thường được dùng để truy cập đến dữ liệu của đối tượng

```
x.ten_phương_thuc (danh_sach_tham_so);
```

2. Kiểu dữ liệu vector

- Ví dụ :
  - Ngày 20/07/2016, có 3000 sinh viên trúng tuyển đăng kí học trường đại học Thủy Lợi
  - Ngày 21/07/2016, có 200 sinh viên chuyển nguyện vọng sang trường khác
  - Ngày 22/07/2016, có 140 sinh viên ở các trường khác đổi nguyện vọng để sang trường Thủy Lợi

→ Dữ liệu thay đổi theo thời gian. Nếu dùng mảng một chiều sẽ không đáp ứng được nhu cầu.

→ Giải pháp: sử dụng dữ liệu kiểu vector

Vector

- Là kiểu dữ liệu tương tự như mảng nhưng có thể thay đổi kích thước khi chèn hoặc loại bỏ phần tử (cấu trúc dữ liệu mảng động)
- Ví dụ:
  - Dãy các số thực: vector<float>
  - Dãy các giá trị logic: vector<bool>
  - Dãy các dãy số nguyên (vector của vector): vector<vector<int>>
- Có rất nhiều hàm hỗ trợ, chẳng hạn kiểm tra số phần tử, thêm hay xoá các phần tử

3. Khai báo vector

- Khai báo thư viện vector trước khi sử dụng :

  #include<vector>

- Cú pháp :
  - vector<kiểu dữ liệu> tên_vector ;
  - vector<kiểu dữ liệu> tên_vector(kích_thước) ;
  - vector<kiểu dữ liệu> tên_vector(kích_thước, giá_trị) ; với giá_trị là giá trị khởi tạo cho các phần tử
- Ví dụ :
  - vector<int> A ; //vector A kiểu nguyên, không có phần tử nào
  - vector<bool> B(10) ; //vector B có 10 phần tử kiểu logic
  - vector<float> C(8, 2.0) ; //vector C có 8 phần tử kiểu thực với giá trị khởi gán là 2.0

Sử dụng vector

- Cách sử dụng vector giống như mảng một chiều

  - Dùng chỉ số để truy cập đến các phần tử trong vector

  Ví dụ : A[i] hoặc A.at(i)

  - Thao tác với từng phần tử của vector tương tự như thao tác với một biến thông thường

- Các thao tác cơ bản :

  - Nhập, xuất dữ liệu

  - Thêm hoặc xóa phần tử, tìm số phần tử của vector

  và rất nhiều phương thức khác

4. Các phương thức

- Rất nhiều hàm có sẵn trong thư viện vector, tham khảo http://www.cplusplus.com/reference/vector/vector
- Một số hàm hay sử dụng
  - v.size() : trả về số phần tử của vector v
  - v.resize(m) : thay đổi cỡ của vector v thành m phần tử
  - v.pop_back() : xoá phần tử cuối cùng của vector v
  - v.push_back(e) : thêm phần tử có giá trị e vào cuối vector v
  - v.back() : tham chiếu đến phần tử cuối cùng của vector v
  - v.front() : tham chiếu đến phần tử đầu tiên của vector v
  - v.clear() : làm rỗng vector v (kích thước của v sẽ là 0)
  - v.empty() : trả về true nếu vector v rỗng

Nhập dữ liệu cho vector

Cách 1 : Nhập số phần tử rồi khai báo vector

```
int n;
cout << "Nhap so phan tu cua vector: ";
cin >> n;
vector<double> A(n);
cout << "Nhap du lieu cho vector\n";
for (int i = 0; i < n; i++)
{
    cout << "A[" << i << "] = ";
    cin >> A[i];
}
```

Nhập dữ liệu cho vector

Cách 2 : Khai báo vector, nhập số phần tử rồi chỉnh lại kích thước của vector

```
vector<double> A;
int n;
cout << "Nhap so phan tu cua vector: ";
cin >> n;
A.resize(n);
cout << "Nhap du lieu cho vector\n";
for (int i = 0; i < n; i++)
{
    cout << "A[" << i << "] = ";
    cin >> A[i];
}
```

Nhập dữ liệu cho vector

Cách 3 : Khai báo vector, dùng vòng lặp để nhập giá trị của các phần tử

```
vector<double> A;
double x;
char traloi;
cout << "Ban co muon nhap khong (y/n)? ";
cin >> traloi;
while (traloi == 'y')
{
    cout << "Nhap mot gia tri ";
    cin >> x;
    A.push_back(x);
    cout << "Ban co nhap tiep khong (y/n)? ";
    cin >> traloi;
}
```

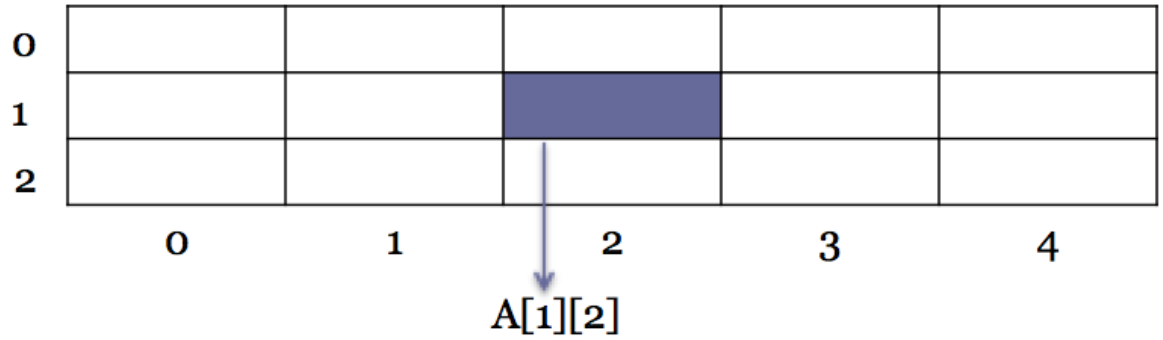
In vector ra màn hình

```
cout << "Xuat du lieu cua vector\n";
for (int i = 0; i < A.size(); i++) cout << A[i] << " ";
cout << endl;
```

Vector của vector

- khai báo một vector các số thực : vector<float> hang(5);

- khai báo một vector có các thành phần là vector số thực : vector<vector<float>> matran(3, hang);



- tương tự như mảng hai chiều

5. Bài tập

Bài 1: Nhập số nguyên dương n và một dãy n số thực. Tạo ra một dãy số mới gồm các số thực dương trong dãy và in ra màn hình dãy số mới đó.

Bài 2: Nhập một dãy số thực, đảo ngược dãy số và in dãy số mới ra màn hình.

Bài 3: Nhập số nguyên dương n và một dãy A có n số thực. Nhập một số nguyên k. Xóa đi k phần tử cuối cùng của dãy A, nếu k ≥ n thì giữ nguyên dãy A. In ra dãy số mới và trung bình cộng của các phần tử trong dãy đó.