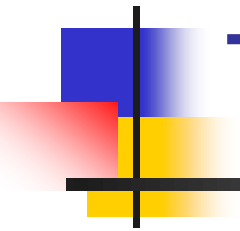
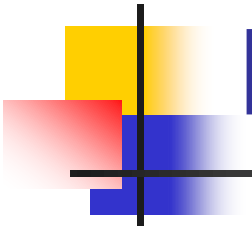


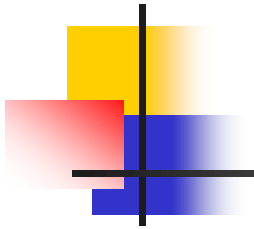
TIN ĐẠI CƯƠNG

Bài 6: Xử lý dữ liệu

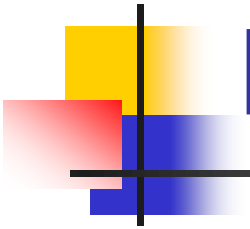


Bài 6: Xử lý dữ liệu

- Một chút về các kiểu dữ liệu
- Phạm vi và vòng đời của biến
- Các kiểu dữ liệu tự tạo
- Kiểu chuỗi (string)
- Các mẫu (template)



Một chút về các kiểu dữ liệu

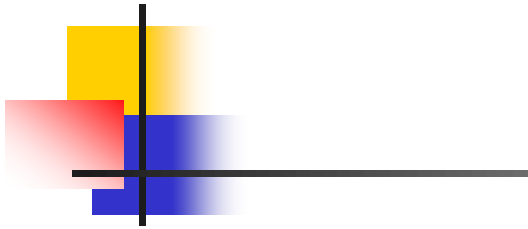


Một chút về các kiểu dữ liệu

- Số nguyên:
  - int (có dấu)
  - unsigned int (không dấu)
  - char (kiểu kí tự, ví dụ: 'a', '9',…)
- Logic: bool
- Số thực: double, float

Một chút về các kiểu dữ liệu

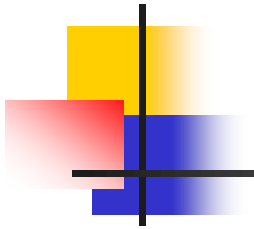
- Khai báo hằng số:
  - const <kiểu> <tên hằng số> = <giá trị>;
  - const int z = 0;
  - const bool b = false;
- Khai báo tham chiếu:
  - <kiểu>& <tên biến> = <tên biến>;
  - int & n = m;
  - double & x = y;



Phạm vi và vòng đời của biến

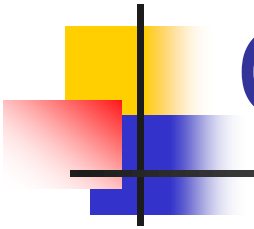
Phạm vi và vòng đời của biến

- Phạm vi: Khối chương trình có thể sử dụng biến đó
- Vòng đời: Khoảng thời gian có thể sử dụng biến đó



Các kiểu dữ liệu tự tạo

Các kiểu dữ liệu tự tạo



- Kiểu dữ liệu: Hầu hết các kiểu dữ liệu trong máy tính đều phỏng theo các "loại" dữ liệu mà con người thường sử dụng
- Các ngôn ngữ lập trình cung cấp một số kiểu dữ liệu cơ bản (số nguyên, số thực, logic,…)
- Cho phép người dùng tổ hợp một số loại dữ liệu cơ bản thành các loại phức tạp hơn. Ví dụ:
  - Phân số: tử số (số thực) + mẫu số (số thực)
  - Sinh viên: tên (chuỗi kí tự) + địa chỉ (chuỗi kí tự) + điểm trung bình học tập (số thực)

Kiểu chuỗi (string)

Kiểu chuỗi (string)

- Dãy các kí tự liên tiếp, viết trong cặp nháy kép: "How are you?","x","",…
- Kiểu dữ liệu mới: string
- Cần: #include <string>
- Khai báo biến:
  - string str;
  - string w("Hello");
  - string s = "Hello";

Kiểu chuỗi (string)

- Các hàm cơ bản: Xem Bảng 5.1 (trang 240)
- Giới thiệu thêm:
  - s.substr(<v>, <k>): Tạo ra chuỗi mới là chuỗi con của s từ vị trí <v> và lấy <k> kí tự
  - s.insert(<v>, s2): Chèn chuỗi s2 vào s từ vị trí <v>

Kiểu chuỗi

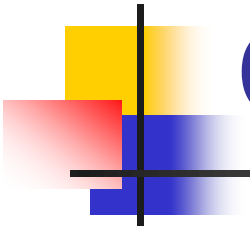
- Bản chất: Dãy các kí tự
- Cần: #include <string>
- Khai báo:
  - string str;
  - string w("Hello");
  - string s = "Hello";
- Các hàm cơ bản: Tham khảo giáo trình
- Nhập liệu: getline(cin, str);

Sử dụng chỉ mục với chuỗi

Dữ liệu	**H**	**A**	**P**	**P**	**Y**		**N**	**E**	**W**
Chỉ mục	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**...**

- Chuỗi s = "HAPPY NEW YEAR"
- s[0] ➔ 'H' s[1] ➔ 'A' s[2] ➔ 'P'
- Cách dùng:
  - Lấy ra: **cout << s[1] << endl;**
  - Ghi vào: **s[4] = 'I';**
  - Kết hợp: **for (int i = 0; i < 10; i++) cout << s[i] << endl;**

Các hàm liên quan

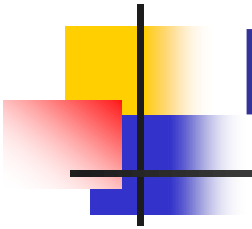


- Có thể coi string là một vector đặc biệt, vector của các kí tự (char)
- Hàm **length()** hoặc **size()**: Trả về chiều dài của string
- Hàm **append(str)**: Thêm str vào cuối string hiện tại (có thể dùng +=)
- Hàm **push_back(c)**: Thêm kí tự c vào cuối string

Các hàm liên quan

- Hàm **insert(v, str)**: Chèn chuỗi str vào vị trí v trong string hiện tại
- Hàm **erase(v, k)**: Xóa k kí tự bắt đầu từ vị trí v
- Hàm **find(str)**: Tìm vị trí xuất hiện đầu tiên của str trong string hiện tại

Bài tập về xử lý chuỗi



- Rất nhiều và phong phú
- Là phần quan trọng của môn học
- Là phần xử lý cần thiết đối với nhiều phần mềm
- Các dạng cơ bản:
  - Kiểm tra, chuẩn hóa chuỗi
  - Tìm, thay thế chuỗi

Bài tập về xử lý chuỗi

1. Nhập vào chuỗi S, in ra màn hình chuỗi vừa nhập và thông tin về chuỗi đó
2. Kiểm tra xem chuỗi có chứa toàn các chữ số hay không?
3. Nhập và đếm số từ trong chuỗi S
4. Hãy xóa mọi kí tự A trong chuỗi W nhập từ bàn phím

Các mẫu (template)

Các mẫu (template)

- Cách viết chương trình không phụ thuộc vào kiểu biến

```
template<T>
T tong(T a, T b) {
        return a+b;
}
```

Bài kiểm tra

Nhập xâu kí tự W

- Đếm xem trong W có bao nhiêu kí tự là những chữ số
- Hãy xóa đi tất cả những kí tự là chữ số (tức là xóa đi tất cả những kí tự '0', '1', '2', '3', '4', '5', '6', '7', '8', '9' khỏi W). Sau đó in ra W