Chinh phục Stack trong C++17

Triết lý Learning How to Learn

Slide Learning CPP

Ngày 15 tháng 1 năm 2026

Lời chào từ người đồng hành

Triết lý Learning How to Learn

Chào mừng bạn đến với hành trình xây dựng những "khối kiến thức"(chunks) vững chắc về Stack. Chúng ta sẽ biến những dòng lệnh khô khan thành những thực thể sống động trong tư duy.

- Không chỉ học code, mà học cách tư duy.
- Hiểu bản chất thông qua các ẩn dụ thực tế.

Lộ trình khám phá: 4 Trạm dừng chân

1. **Chương 1: Chiếc ống Pringles thần kỳ** - Bản chất LIFO.
2. **Chương 2: Bộ lệnh điều khiển** - Các thao tác `push`, `pop`, `top`.
3. **Chương 3: Cuộc dạo chơi trong ngăn xếp** - Duyệt Stack bằng `while` và C++17.
4. **Chương 4: Khi Stack kết hợp cùng Algorithms** - Biến hóa sức mạnh.

Chương 1: Chiếc ống Pringles thần kỳ

Ẩn dụ cốt lõi

Hãy tưởng tượng **Stack** giống hệt như một **ống khoai tây chiên Pringles**.

- Miếng khoai bỏ vào **cuối cùng** $\rightarrow$ ăn **đầu tiên**.
- Miếng khoai ở **đáy ống** $\rightarrow$ phải đợi ăn hết các miếng bên trên.
- **Nguyên lý LIFO:** Last In, First Out (Vào sau cùng, ra đầu tiên).

Tại sao chúng ta cần Stack?

Ứng dụng thực tế

Stack cực kỳ "kỷ luật", ép bạn làm việc theo thứ tự nhất định:

- Đảo ngược một chuỗi.
- Nút **Undo** trong trình soạn thảo văn bản.
- Kiểm tra tính đúng đắn của các cặp dấu ngoặc.

Cấu trúc cơ bản trong C++17

Để gọi "ống Pringles"này ra, ta dùng thư viện <stack>.

```cpp
#include <iostream>
#include <stack> // Thu vien chua Stack
using namespace std;

int main() {
    stack<int> s; // Tao mot ngan xep chua cac so nguyen
    return 0;
}

```

Kiểm tra "một chút"kiến thức

Câu hỏi

Nếu bỏ lần lượt 3 cuốn sách: "Toán", "Văn", "Anh"vào một cái thùng hẹp (Stack). Thứ tự nhận được khi lấy ra từng cuốn là gì?

Kiểm tra "một chút"kiến thức

Câu hỏi

Nếu bỏ lần lượt 3 cuốn sách: "Toán", "Văn", "Anh"vào một cái thùng hẹp (Stack). Thứ tự nhận được khi lấy ra từng cuốn là gì?

Đáp án

Thứ tự: **Anh** $\rightarrow$ **Văn** $\rightarrow$ **Toán**. *Giải thích: Cuốn "Anh"vào sau cùng nên nằm trên cùng (LIFO).*

Ẩn dụ Cánh tay Robot

Chỉ có một cánh tay duy nhất ở miệng ống thực hiện 3 hành động:

- `push(giá_trị)`: Đặt một miếng khoai mới lên đỉnh.
- `pop()`: Gắp miếng trên cùng ra và vứt đi (không trả về giá trị).
- `top()`: Nhìn lướt xem miếng trên cùng là gì nhưng vẫn để lại trong ống.

Các lệnh phụ trợ

Lệnh	**Công dụng**
`empty()`	Kiểm tra ống còn miếng nào không (True/False)
`size()`	Đếm tổng số miếng khoai trong ống

Code thực tế: Vận hành ống Pringles

```cpp
stack<string> pringles;
pringles.push("Vi Cay");
pringles.push("Vi Pho Mai");
pringles.push("Vi Tao Bien");

cout << pringles.top() << endl; // Ket qua: Vi Tao Bien

pringles.pop(); // An mieng tren cung

cout << pringles.top() << endl; // Ket qua: Vi Pho Mai
cout << "Con: " << pringles.size() << " mieng.";

```

Tình huống

Trong trò chơi, người chơi đi: Trái $\rightarrow$ Lên. Khi ấn nút **Undo**, lệnh nào được thực hiện?

Tình huống

Trong trò chơi, người chơi đi: Trái $\rightarrow$ Lên. Khi ấn nút **Undo**, lệnh nào được thực hiện?

Giải đáp

Lệnh `pop()` được thực hiện. Nó gỡ bỏ hành động vừa xảy ra nhất để đưa bạn về trạng thái ngay trước đó.

Chương 3: Cuộc dạo chơi trong ngăn xếp

Thách thức: Cỗ máy quét bí ẩn

Stack không cho phép dùng mắt quét từ đầu đến cuối như Vector. Nó là một "kẻ kín tiếng".

- Để xem bên trong, bạn phải dùng "camera nội soi".
- Có hai cách chính: Kiểu truyền thống (Phá hủy) và Kiểu C++17 hiện đại.

Duyệt Stack kiểu truyền thống

```cpp
stack<int> s;
// ... push 10, 20, 30 ...

while (!s.empty()) {
    auto topValue = s.top(); // C++17 dung auto
    cout << "Dang xem mieng: " << topValue << endl;
    s.pop(); // Bat buoc phai bo ra moi xem duoc mieng tiep theo
}

```

Lưu ý quan trọng

Sau vòng lặp này, Stack sẽ **trống rỗng**. Bạn phải ăn hết ống khoai mới biết bên trong có gì!

Duyệt mà không làm mất dữ liệu

```
stack<int> goc;
for(int x : {1, 2, 3, 4, 5}) goc.push(x);

// Tao mot ban sao de bao ve du lieu goc
stack<int> ban_sao = goc;

while (!ban_sao.empty()) {
    cout << ban_sao.top() << " ";
    ban_sao.pop();
}
// goc.size() van con nguyen 5 phan tu!

```

Phong cách C++17: Vòng lặp "Bán tự động"

Tận dụng cấu trúc của vòng lặp `for` để code gọn gàng hơn.

```
stack<string> s;
for(auto x : {"S", "T", "A", "C", "K"}) s.push(x);

// for ( ; Dieu kien con hang ; Day hang cu ra )
for (; !s.empty(); s.pop()) {
    auto& item = s.top(); // Dung auto& de lay tham chieu nhanh
    cout << item << " ";
}

```

Tại sao Stack không có Iterator?

Ẩn dụ: Chiếc giếng hẹp và sâu

Stack là một cái giếng hẹp, chỉ vừa để thả gàu xuống.

- **Iterator (Thước kẻ):** Bạn muốn đứng ngoài chỉ trỏ vào từng món đồ mà không nhấc chúng lên $\rightarrow$ **Không thể**.
- **Triết lý:** Stack ép bạn tuân thủ LIFO để bảo vệ tính kỷ luật của dữ liệu. Nếu cần duyệt nhiều, hãy dùng `vector`.

So sánh Vector và Stack

Tính năng	**Vector (Cái khay)**	**Stack (Cái ống)**
Truy cập	Bất cứ vị trí nào	Chỉ ở đỉnh (`top()`)
`foreach`	Hỗ trợ (**Có**)	**Không** (Bảo vệ LIFO)
Tốc độ	Chậm ở đầu	Cực nhanh ở đỉnh
Ẩn dụ	Danh sách đi chợ	Ống Pringles

Kỹ thuật "Đổ ra đĩa"

Vì Stack không có Iterator, ta "đổ"dữ liệu ra vector để xử lý.

```cpp
// Buoc 1: Do ra "dia" Vector
vector<int> plate;
while(!s.empty()) {
    plate.push_back(s.top());
    s.pop();
}
// Buoc 2: Sap xep tang dan bang Algorithms
sort(plate.begin(), plate.end());

// Buoc 3: Dong goi lai vao Stack
for(auto x : plate) s.push(x);

```

Bài toán cuối khóa: Đảo ngược chuỗi

Đề bài

Nhập: "HOC SINH CAP HAI"→ Xuất: "IAH PAC HNIS COH".

```
string text = "HOC SINH CAP HAI";
stack<char> sentence;
for (auto x : text) sentence.push(x);

for (; !sentence.empty(); sentence.pop()) {
    cout << sentence.top();
}

```

Tổng kết

Bạn đã làm chủ Stack từ bản chất LIFO đến kỹ thuật duyệt C++17 chuyên nghiệp!