

# NGĂN XẾP



# Mục tiêu

---

- Hiểu được khái niệm cấu trúc dữ liệu ngăn xếp
- Biết cách cài đặt ngăn xếp bằng mảng và con trỏ
- Biết cách vận dụng ngăn xếp để giải một số bài toán

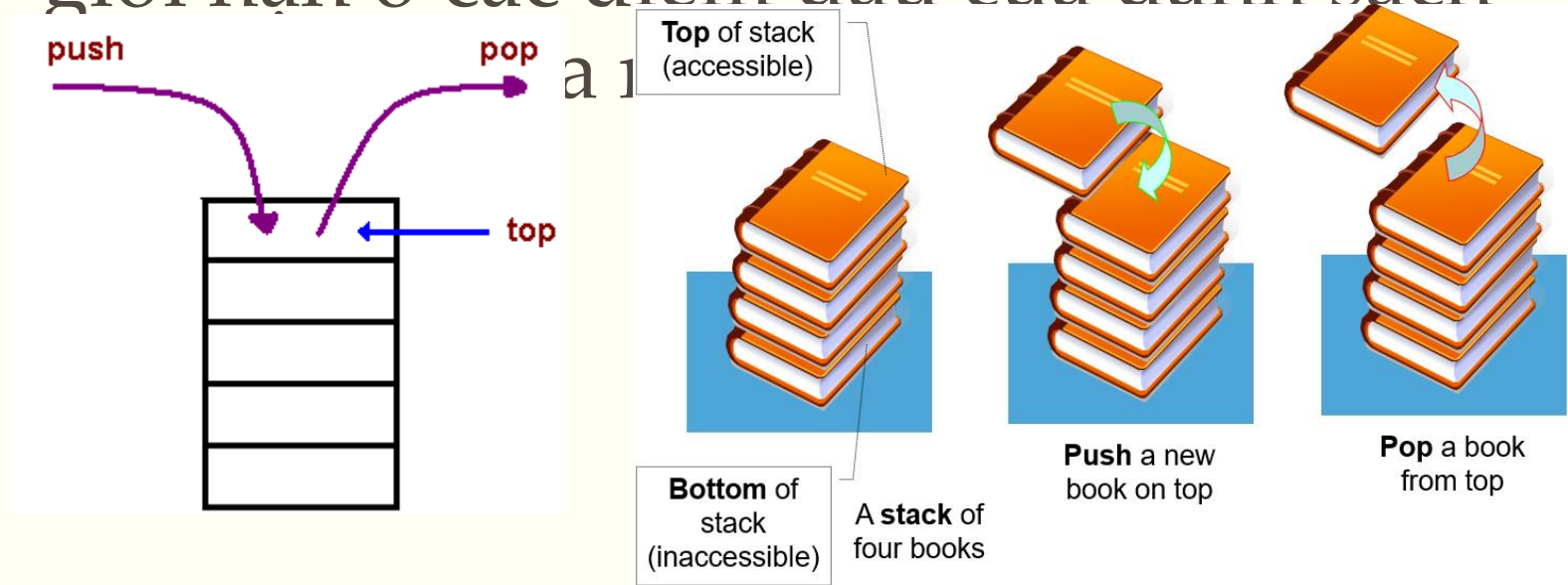
# Nội dung chính

---

- Khái niệm
- Các phép toán
- Cài đặt bằng mảng
- Cài đặt bằng con trỏ

# Khái niệm

- Ngăn xếp là một danh sách
- Các phép toán chèn, xóa phần tử được giới hạn ở các điểm đầu của danh sách



# Các phép toán cơ bản

---

- Khởi tạo ngăn xếp rỗng
- Kiểm tra ngăn xếp rỗng hay không?
- Kiểm tra ngăn xếp đầy hay không?
- Chèn phần tử vào đỉnh ngăn xếp
- Tìm và lấy ra phần tử ở đỉnh ngăn xếp

# Cài đặt ngăn xếp bằng mảng


---

```
5  template <class Item>
6  class Stack {
7      Item element[::Max];
8      static int top;
```

# Cài đặt ngăn xếp bằng mảng (tiếp...)

---

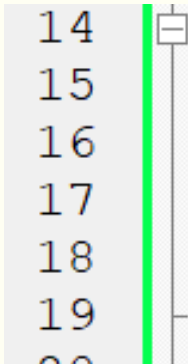
- Khởi tạo ngăn xếp rỗng



```
public:
    Stack() {
        top=0;
    }
```

# Cài đặt ngăn xếp bằng mảng (tiếp...)

- Kiểm tra ngăn xếp rỗng ?
- Hàm isEmpty :
  - Input: ngăn xếp Q
  - Output:
    - 1: Nếu ngăn xếp rỗng
    - 0: ngăn xếp đã có phần tử



```
bool isEmpty() {  
    if(top==0)  
        return true;  
    else  
        return false;  
}
```



# Cài đặt ngăn xếp bằng mảng (tiếp...)

---

- Kiểm tra ngăn xếp đầy?
- Xây dựng hàm isFull:
  - Input: ngăn xếp Q
  - Output:
    - 1: ngăn xếp đầy
    - 0: ngược lại

```
21 bool IsFull() {  
22     if (top==::Max)  
23         return true;  
24     else  
25         return false;  
26 }
```

# Cài đặt ngăn xếp bằng mảng (tiếp...)

---

## ■ Thêm phần tử vào ngăn xếp

```
38 void Push(Item x) {  
39  
40     if (IsFull()) {  
41         cout<<"Ngan xep day khong chen duoc "<<endl;  
42         return;  
43     }  
44     else {  
45         element[top+1]=x;  
46         top++;  
47     }  
48  
49 }
```

# Cài đặt ngăn xếp bằng mảng (tiếp...)

---

- Tìm và lấy ra phần tử ở đỉnh ngăn xếp

```
27 Item Pop() {  
28     if (IsEmpty()) {  
29         cout<<"Ngan xep rong "<<endl;  
30         return 0;  
31     } else {  
32         Item x= element[top];  
33         top--;  
34         return x;  
35     }  
36 }
```

# Cài đặt ngăn xếp bằng con trỏ

---

```
1  #include <iostream>
2  using namespace std;
3  template <class Item>
4
5  class StackP {
6      private:
7          struct Node {
8              Item data;
9              struct Node* next;
10         };
11         struct Node* top;
```

# Cài đặt ngăn xếp bằng con trỏ

---

- Khởi tạo ngăn xếp rỗng

```
12      public:  
13          StackP () {  
14              top = NULL;  
15          }
```

# Cài đặt ngăn xếp bằng con trỏ

---

- Thêm phần tử vào đỉnh ngăn xếp

```
20 void Push(Item x) {  
21     Node* temp = new Node;  
22     temp->data = x;  
23     temp->next = NULL;  
24  
25     temp->next = top;  
26     top = temp;  
27 }
```

# Cài đặt ngăn xếp bằng con trỏ

---

## ▪ Lấy phần tử từ đỉnh ngăn xếp

```
29  Item Pop() {  
30      Item ret;  
31      if (!this->IsEmpty()) {  
32          ret = top->data;  
33          Node* temp = top;  
34          top = top->next;  
35          delete temp;  
36      }  
37      return ret;  
...
```

# Cài đặt ngăn xếp bằng con trỏ

---

- Duyệt các phần tử của ngăn xếp

```
39 void Display() {  
40     while (!this->isEmpty()) {  
41         cout << top->data << endl;  
42         top = top->next;  
43     }  
44 }
```



# Bài tập

---

- Đổi cơ số thập phân sang nhị phân
- Kiểm tra sâu đối xứng
- Đảo ngược sâu
- Cho chuỗi ký tự (a-z), đếm số lượng các ký tự liên tiếp nhau
  - Input: aaabbaaac
  - Output: a3b2a3c1

# Tổng kết

---

- Ngăn xếp là danh sách thao tác thêm, loại bỏ ở một phía của danh sách
- Thứ tự “Vào sau ra trước”

# Tiếp theo ...

---

- Hàng đợi