# CHƯƠNG 2: NỘI DUNG CÔNG VIỆC

## 2.1. Ngăn xếp (STACK):

### 2.1.2. Ðịnh nghĩa cấu trúc dữ liệu ngắn xếp và hàng đợi

Stack là một vật chứa (container) các đối tượng làm việc theo cơ chế LIFO (Last In First Out) nghĩa là việc thêm một đối tượng vào stack hoặc lấy một đối tượng ra khỏi stack được thực hiện theo cơ chế "Vào sau ra trước". Các đối tượng có thể được thêm vào stack bất kỳ lúc nào nhưng chỉ có đối tượng thêm vào sau cùng mới được phép lấy ra khỏi stack. Thao tác thêm 1 đối tượng vào stack thường được gọi là "Push". Thao tác lấy 1 đối tượng ra khỏi stack gọi là "Pop". Trong tin học, CTDL stack có nhiều ứng dụng: khử đệ qui, tổ chức lưu vết các quá trình tìm kiếm theo chiều sâu và quay lui, vét cạn, ứng dụng trong các bài toán tính toán biểu thức, . Một hình ảnh một stack

Ta có thể định nghĩa CTDL stack như sau: stack là một CTDL trừu tượng (ADT) tuyến tính hỗ trợ 2 thao tác chính: Push(o): Thêm đối tượng o vào đầu stack Pop(): Lấy đối tượng ở đầu stack ra khỏi stack và trả về giá trị của nó. Nếu stack rỗng thì lỗi sẽ xảy ra. Ngoài ra, stack cũng hỗ trợ một số thao tác khác: isEmpty(): Kiểm tra xem stack có rỗng không. Top(): Trả về giá trị của phần tử nằm ở đầu stack mà không hủy nó khỏi stack. Nếu stack rỗng thì lỗi sẽ xảy ra. Các thao tác thêm, trích và huỷ một phần tử chỉ được thực hiện ở cùng một phía của Stack do đó hoạt động của Stack được thực hiện theo nguyên tắc LIFO (Last In First Out - vào sau ra trước). Ðể biểu diễn Stack, ta có thể dùng mảng 1 chiều hoặc dùng danh sách liên kết.

### 2.1.3. Biểu diễn stack dùng mảng

*-* Ta có thể tạo một stack bằng cách khai báo một mảng 1 chiều với kích thước tối đa là N (ví dụ, N có thể bằng 1000). Như vậy stack có thể chứa tối đa N phần tử đánh số từ 0 đến N -1. Phần tử nằm ở đầu stack sẽ có chỉ số t (lúc đó trong stack

Tạo stack S và quản lý đỉnh stack bằng biến t: Data S [N]; Int t; Việc cài đặt stack thông qua mảng một chiều đơn giản và khá hiệu quả. Tuy nhiên, hạn chế lớn nhất của phương án cài đặt này là giới hạn về kích thước của stack N. Giá trị của N có thể quá nhỏ so với nhu cầu thực tế hoặc quá lớn sẽ làm lãng phí bộ nhớ.

### 2.1.4. Biểu diễn stack dùng danh sách:

- Ta có thể tạo một stack bằng cách sử dụng một danh sách liên kết đơn (danh sách liên kết). Có thể nói, danh sách liên kết có những đặc tính rất phù hợp để dùng làm stack vì mọi thao tác trên stack đều diễn ra ở đầu stack.

- Sau đây là các thao tác tương ứng cho list-stack:

**Tạo Stack S rỗng**

LIST \* S;

Lệnh S.pHead=l.pTail= NULL tạo ra một Stack S rỗng.

**Kiểm tra stack rỗng :**

char IsEmpty(LIST &S)

{

if (S.pHead == NULL) // stack rỗng return 1;

else return 0;

}

**Thêm một phần tử p vào stack S**

void Push(LIST &S, Data x)

{

InsertHead(S, x);

}

**Trích huỷ phần tử ở đỉnh stack S**

Data Pop(LIST &S)

{ Data x;

if(isEmpty(S)) return NULLDATA; x = RemoveFirst(S);

return x;

}

**Xem thông tin của phần tử ở đỉnh stack S**

Data Top(LIST &S)

{ if(isEmpty(S)) return NULLDATA; return l.Head->Info;

}

### 2.1.5. Ứng dụng của Stack:

- Cấu trúc Stack thích hợp lưu trữ các loại dữ liệu mà trình tự truy xuất ngược với trình tự lưu trữ, do vậy một số ứng dụng sau thường cần đến stack : - Trong trình biên dịch (thông dịch), khi thực hiện các thủ tục, Stack được sử dụng để lưu môi trường của các thủ tục. - Trong một số bài toán của lý thuyết đồ thị (như tìm đường đi), Stack cũng thường được sử dụng để lưu dữ liệu khi giải các bài toán này. Ngoài ra, Stack cũng còn được sử dụng trong trường hợp khử đệ qui đuôi.

## 2.2. Kế hoạch phân công công việc:

*Biểu đồ Gantt*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| TT | Nhiệm Vụ | 22/09/2022 | 23/09/2022 | 24/09/2022 | 27/09/2022 | 5/10/2022 |
| 1 | Tổ chức phân chia công việc |  |  |  |  |  |
| 2 | Chọn một chương trong CSLD & GT |  |  |  |  |  |
| 3 | Tìm kiếm tài liệu tham khảo |  |  |  |  |  |
| 4 | Soạn thảo chương 1 |  |  |  |  |  |
| 5 | Hoàn chỉnh công việc trên Github |  |  |  |  |  |
| 6 | Soạn báo cáo chương 2,3 |  |  |  |  |  |
| 7 | Chuẩn bị báo cáo slides |  |  |  |  |  |
| 8 | Kiểm tra và đánh giá |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  | Lê Ly Ta | |  |  |
|  |  |  |  |  |  |  |
|  |  |  | Thạch Nia Thy | |  |  |

## 2.3. Kết quả:

* **Link GitHub:**
* [**https://github.com/NiaThy12345/HelloWord.git**](https://github.com/NiaThy12345/HelloWord.git)
* **Lịch sử commit:**

# 

*Danh sách commit*

# CHƯƠNG 3: KẾT LUẬN

* **3.1.Kết quả đạt được**
* Trong bài tập lớn này nhóm em đã rèn luyện được tầm quan trọng của làm việc nhóm:
* Sẽ mang lại kết quả tốt cho công việc khi mà công việc đó từng cá nhân không thể làm được hoặc làm được mà hiệu quả không cao.
* Tận dụng được những điểm mạnh và điểm yếu của mỗi cá nhân để công việc được hiệu quả hơn.
* Học hỏi lẫn nhau giữa các thành viên trong nhóm.
* **3.2 Bài học rút ra**
* Tìm ra ý tưởng hoặc giải pháp để giải được quyết vấn đề.
* Học hỏi và chia sẽ được nhiều kinh nghiệm.
* Chú trọng trong công việc.

# TÀI LIỆU THAM KHẢO

* Về Git và Github: <https://csc.edu.vn/lap-trinh-va-csdl/tin-tuc/kien-thuc-lap-trinh/Git-la-gi--Nhung-khai-niem-co-ban-khi-lam-viec-tren-Git-4133> (truy cập ngày 10/5/2022)
* Tài liệu giảng dạy học phần ... (tên thầy biên soạn – năm)
* Tài liệu giảng dạy....