Trường: **ĐH CNTP TP.HCM** Khoa: **Công nghệ thông tin** Bộ môn: **Công nghệ phần mềm.** MH: **TH Cấu trúc dữ liệu & giải thuật**	**BÀI 3. STACK**	

A. MỤC TIÊU:

- Định nghĩa, cài đặt được cấu trúc Stack, Queue sử dụng danh sách liên kết đơn
- Vận dụng được Stack, Queue trong một số bài toán cụ thể.

B. DỤNG CỤ - THIẾT BỊ THÍ NGHIỆM CHO MỘT SV:

STT	Chủng loại – Quy cách vật tư	Số lượng	Đơn vị	Ghi chú
1	Computer	1	1	

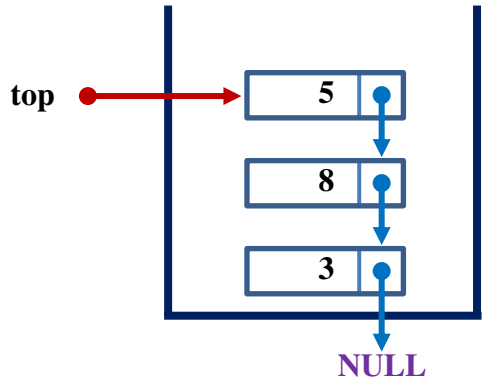
C. NỘI DUNG THỰC HÀNH

I. Tóm tắt lý thuyết

1. Mô tả Stack

– Một stack là một cấu trúc dữ liệu mà việc thêm vào và loại bỏ phần tử chỉ được thực hiện ở một đầu duy nhất (*gọi là đỉnh* – top của stack).

– Là một loại cấu trúc dữ liệu hoạt động theo nguyên tắc: vào sau ra trước – **LIFO** (Last In First Out).



2. Thao tác trên Stack sử dụng DSLK đơn

a. Khai báo kiểu dữ liệu Stack

```
struct SNode
{
	<Data> Info;
	SNode *next;
};
```

```
struct Stack
{
	SNode* top;
};
```

b. Tạo nút mới cho Stack: Tương tự như tạo Snode trong danh sách liên kết đơn.

c. Khởi tạo Stack

Sau khi khởi tạo thì Stack phải rỗng, nên con trỏ top sẽ trỏ NULL.

d. Kiểm tra Stack rỗng.

Kiểm tra Stack có rỗng hay không? Nếu rỗng nghĩa là con trỏ top có giá trị NULL, sẽ trả về 1, ngược lại trả về 0.

e. Thêm phần tử mới p có giá trị x vào Stack (Push): tương tự phương thức thêm một phần tử và đầu danh sách liên kết

- **Bước 1:** Nếu phần tử muốn thêm p không tồn tại thì không thực hiện.
- **Bước 2:** Nếu Stack rỗng thì
  + top = p;
- **Bước 3:** Ngược lại
  + p→next = top;
  + top = p;

f. Lấy giá trị và hủy phần tử đầu Stack (Pop): tương tự phương thức xóa phần tử đầu của dánh sách liên kết

- **Bước 1:** Nếu ngăn xếp rỗng thì: Không thực hiện gì cả.
- **Bước 2:** Khi (top != NULL) thì: thực hiện các công việc sau:
  + **Bước 2.1:** p = top;
  + **Bước 2.2:** top = top→next;
  + **Bước 2.3:** Lưu lại thông tin nút bị xóa vào một tham số x;
  + **Bước 2.4:** delete p; *//Hủy nút do p trỏ đến (con trỏ p)*

g. Lấy giá trị của phần tử ở đầu Stack

- **Bước 1:** Nếu Stack rỗng thì: Không thực hiện gì cả.

- **Bước 2:** Khi (top != NULL) thì: thực hiện công việc sau:
  + Lấy thông tin của nút ở đỉnh của Stack gán vào một tham số x để sử dụng.

h. Xem nội dung của Stack

- **Bước 1:** Nếu ngăn xếp rỗng thì: Thông báo ngăn xếp rỗng và không thực hiện.
- **Bước 2:** Gán p = top; *//p lưu địa chỉ của phần tử đỉnh Stack*
- **Bước 3:**

  Lặp lại trong khi (con trỏ p còn khác NULL) thì thực hiện:
  + Xuất nội dung của phần tử tại con trỏ p.
  + p = p→next; *//Xét phần tử kế*

II. Bài tập hướng dẫn mẫu

Bài 1. Đổi cơ số cho 1 số nguyên n từ hệ 10 sang hệ a (*với* $2 \le a \le 9$)?

Bước 1: Tạo một Project mới → đặt tên: Stack_DoiCoSo_<Họ tên sinh viên>

Bước 2: Khai báo thêm các thư viện cơ bản cho chương trình.

Bước 3: Khai báo cấu trúc dữ liệu Stack cho chương trình.

Bước 4: Viết các hàm cần thiết cho chương trình như sau:

- Viết hàm tạo mới (cấp phát) một nút thông tin x.
- Viết hàm khởi tạo stack (làm rỗng stack).
- Viết hàm kiểm tra stack có rỗng hay không.
- Viết hàm thêm một nút p vào vị trí đỉnh của stack.
- Viết hàm lấy ra và hủy một nút ở vị trí đỉnh của stack.
- Viết hàm ứng dụng stack để đổi cơ số của một số nguyên dương n từ hệ 10 sang hệ a (a=2, 8, 16).

Bước 5: Viết hàm main để thực thi chương trình.

```
//*************************************************
void main()
{
 int n, coso;
 do
 {
    printf("Hay nhap mot so he 10: ");
    scanf("%d", &n);
 }while(n <= 0);
```

```
do
{
   printf("Hay nhap co so muon doi: ");
   scanf("%d", &coso);
}while(coso <= 0);
Stack_DoiCoSo(n, coso);
getch();
}
```

III. Bài tập ở lớp

Bài 1. Đổi cơ số cho 1 số nguyên n từ hệ 10 sang hệ 16

Bài 2. Viết chương trình minh họa ứng dụng Stack để lưu các thao tác undo và redo trong chương trình MS Word.

- Xây dựng Stack để lưu nội dung thao tác MS Word: delete char, delete selected words, typing char, bold, italic, underline.
- Nhập/đọc vào Stack1 một số thao tác đã thực hiện; Stack2: danh sách tạm lưu các tham tác bị undo, dùng phục hồi khi redo.
- Chương trình chạy có 3 lựa chọn:
    - Do: thực hiện 1 thao tác → đưa thao tác đó vào Stack1
    - Undo: hủy thao tác, lấy ra khỏi stack1 (lưu Stack2 để phục vụ cho redo).
    - Redo: lấy thao tác từ Stack2 phục hồi vào Stack1.

Sau khi nhập/đọc thao tác vào Stack1. Cho người dùng thực hiện nhiều lần một trong ba thao tác (do, undo, redo). Mỗi thao tác hãy xuất ra nội dung thực hiện của thao tác đó.

Bài 3. Ứng dụng Stack (*cài đặt theo dạng danh sách liên kết đơn*) để viết chương trình thực hiện việc kiểm tra sự hợp lệ của các cặp dấu ngoặc.

Mỗi dấu "(", "{", hoặc "[" đều phải có một dấu đóng tương ứng ")", "}", hoặc "]".

Ví dụ:

- Đúng: ( )(( )){([( )])}
- Sai:)(( ))
- Sai: ({[])}

Viết giải thuật nhận một chuỗi chứa các ký tự mở, đóng ngoặc. Kiểm tra chuỗi đã cho có hợp lệ về việc mở và đóng ngoặc không?

Bài 4. Tìm dấu ngoặc đơn trùng lặp (dư thừa) trong một biểu thức

Đưa ra một biểu thức cân bằng có thể chứa dấu ngoặc mở và đóng, hãy kiểm tra xem biểu thức có chứa bất kỳ dấu ngoặc đơn trùng lặp nào hay không

Ví dụ:

Input: ((x+y))+z

Output: Trùng lặp () được tìm thấy trong biểu thức con ((x + y))

Input: (x+y)

Output: Không thấy trùng lặp ()

Input: ((x+y)+((z)))

Output: Trùng lặp () được tìm thấy trong biểu thức con ((z))

IV. Bài tập về nhà

Bài 5.

Đảo ngược 1 **chuỗi ký tự** bất kỳ được nhập từ bàn phím bằng cách dùng Stack. Kiểm tra chuỗi có đối xứng không?

Ví dụ: chuỗi nhập *"Nice to meet you!"* đưa vào Stack.

Chuỗi xuất ra từ Stack là: *!uoy teem ot eciN*.

Bài 6. Giải bài toán Tính giai thừa bằng Stack.

Gợi ý: dùng stack lưu các giá trị giai thừa từ lớn đến bé. Sau đó lần lượt lấy các giá trị giai thừa trong stack ra từ bé đến lớn. Giá trị giai thừa lớn tính dựa vào giá trị giai thừa bé trước đó.

Bài 7. Bài tập nâng cao:

Ứng dụng ngăn xếp Stack để chuyển từ biểu thức trung tố (infix) sang hậu tố (postfix) và tính giá trị biểu thức.

Hướng dẫn:

Biểu thức hậu tố (Postfix) là biểu thức được biểu diễn bằng cách đặt các toán tử ra sau các toán hạng.

Một vài ví dụ minh hoạ:

Infix	Postfix
A / B – C * D	A B / C D * +
A / (B – C * D)	A B C D * - /

Yêu cầu bài toán được tách thành 2 bước:

- Bước 1: chuyển biểu thức trung tố thành biểu thức hậu tố
- Bước 2: từ biểu thức hâu tố thu được ở bước 1, thực hiện tính toán và in kết quả cho người dùng.

--- HẾT ---