KỸ THUẬT LẬP TRÌNH C

Chương 5: Ngăn xếp

1. Giới thiệu
2. Cài đặt ngăn xếp theo kiểu kế tiếp
3. Cài đặt ngăn xếp theo kiểu liên kết
4. Ứng dụng ngăn xếp

1. Giới thiệu

Ngăn xếp là một kiểu danh sách đặc biệt mà thao tác thêm nút và thao tác xoá nút luôn luôn được thực hiện ở một đầu gọi là đỉnh (top).

Chúng ta thấy trên ngăn xếp nút thêm vào sau lại được lấy ra trước nên cấu trúc ngăn xếp còn được gọi là cấu trúc LIFO (Last in – First out).

2. Cài đặt ngăn xếp theo kiểu kế tiếp

Cài đặt ngăn xếp như một danh sách kề bằng cách dùng mảng một chiều: phần tử 0 của mảng xem như đáy ngăn xếp, phần tử cuối của mảng là đỉnh ngăn xếp.

a. Khai báo cấu trúc ngăn xếp

```
#define MAX 50
struct STACK{
    int top;          //Đỉnh ngăn xếp
    int nodes[MAX]; /* Mỗi phần tử của mảng là một nút
                       của ngăn xếp */
};
```

b. Các thao tác trên ngăn xếp

- Khởi động ngăn xếp

```
void Initialize(STACK *stack) {
    stack->top = -1;
}
```

- Kiểm tra ngăn xếp có rỗng hay không

```
int Empty(STACK *stack) {
    if (stack->top == -1)
        return 1;
    return 0;
}
```

- Kiểm tra ngăn xếp có đầy hay không

```
int Full(STACK *stack) {
        if (stack->top == MAX - 1)
                return 1;
        return 0;
}
```

- Thêm nút mới vào đỉnh ngăn xếp

```
void Push (STACK * stack, int x) {
        if (Full(stack))
                printf("Ngăn xếp đầy\n");
        else  stack->nodes[++stack->top] = x;
}
```

- Xoá nút tại đỉnh ngăn xếp

```
int Pop (STACK * stack) {
        if (Empty(stack)) {
                printf("Ngăn xếp rỗng\n");
                <Có thể trả về một gía trị qui ước>
        }
        else  return (stack->nodes[stack->top--]);
}
```

- Truy xuất nút đang ở đỉnh ngăn xếp

```
int StackTop(STACK *stack) {
        if (Empty(stack)) {
                printf("Ngăn xếp rỗng\n");
                <Có thể trả về một gía trị qui ước>
        }
        else  return (stack->nodes[stack->top]);
}
```

3. Cài đặt ngăn xếp theo kiểu liên kết

Cài đặt ngăn xếp như một danh sách liên kết: con trỏ đầu danh sách liên kết là pstack chỉ nút tại đỉnh ngăn xếp, nút tại đỉnh chỉ nút thứ hai, …, nút cuối danh sách xem như đáy ngăn xếp.

a. Khai báo cấu trúc một nút của ngăn xếp

```
struct node {
    int info; //Chứa nội dung nút
    node *next; //Con trỏ chỉ nút kế tiếp
}

typedef node *NODEPTR;
```

b. Các thao tác trên ngăn xếp

- Cấp phát biến động làm nút ngăn xếp

```
NODEPTR GetNode() {
    NODEPTR p;
    p = (NODEPTR) malloc(sizeof(node));
    return p;
}
```

- Giải phóng biến động đã cấp phát trước đó

```
void FreeNode(NODEPTR p) {
    free(p);
}
```

- Khởi động ngăn xếp

```
void Initialize(NODEPTR *pstack) {
    *pstack = NULL;
}
```

- Kiểm tra ngăn xếp có rỗng hay không

```
int Empty(NODEPTR pstack) {
    if (pstack == NULL)
            return 1;
    return 0;
}
```

- Thêm nút mới vào đỉnh ngăn xếp

```
void Push (NODEPTR *pstack, int x) {
        NODEPTR p;
        p = GetNode();
        p->info = x;
        p->next = *pstack;
        *pstack = p;
}
```

- Xoá nút tại đỉnh ngăn xếp

```
int Pop (NODEPTR *pstack) {
    NODEPTR p;
    int x;
    if(Empty(*pstack)) {
            printf("Ngăn xếp rỗng\n");
            <Có thể trả về một gía trị qui ước>
    }
    else {
            p = *pstack;
            x = p->info;
            *pstack = p->next;
            FreeNode(p);
            return x;
    }}
```

- Truy xuất nút đang ở đỉnh ngăn xếp

```
int StackTop(NODEPTR pstack) {
        if (Empty(pstack)) {
                printf("Ngăn xếp rỗng\n");
                <Có thể trả về một gía trị qui ước>
        }
        else  return (pstack->info);
}
```

4. Ứng dụng ngăn xếp

Ngăn xếp thương được dùng để giải quyết các vấn đề có cơ chế LIFO như:

- giải quyết các vấn đề trong các trình biên dịch của các ngôn ngữ lập trình (Kiểm tra cú pháp của các câu lệnh, xử lý các biểu thức toán học, xử lý việc gọi các chương trình con, xử lý việc gọi các hàm đệ qui),
- chuyển một giải thuật đệ quy sang một giải thuật không đệ quy (khử đệ quy), …

Hết