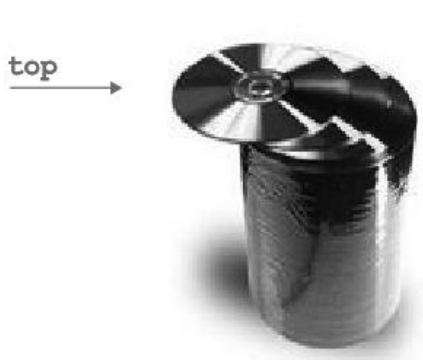
สแตค (Stack)

สแตคคือข้อมูลที่เรียงลำดับซ้อนกัน ข้อมูลที่อยู่ด้านบนสุดเรียกว่า top

การเพิ่มข้อมูลเข้าสแตคเรียกว่า push ส่วนการดึงข้อมูลออกเรียกว่า pop

สแตคมีความหมายอีกอย่างหนึ่งว่า Last In First Out (LIFO) นั่นคือข้อมูลสุดท้ายที่ถูกเพิ่มเข้ามาจะเป็นข้อมูลแรกที่ถูกดึงออก



สแตคของแผ่นซีดี

สแตคของหนังสือ

Static Implementations

กำหนดสัญลักษณ์ค่าคงที่ SIZE ให้มีค่าเท่ากับ 5
ประกาศตัวแปร top เป็นตัวแปรโกลบอลที่จะเก็บเลขจำนวนเต็ม
ประกาศตัวแปร Items เป็นตัวแปรโกลบอลแบบอะเรย์ที่จะเก็บตัวอักษรมีขนาดเท่ากับ SIZE

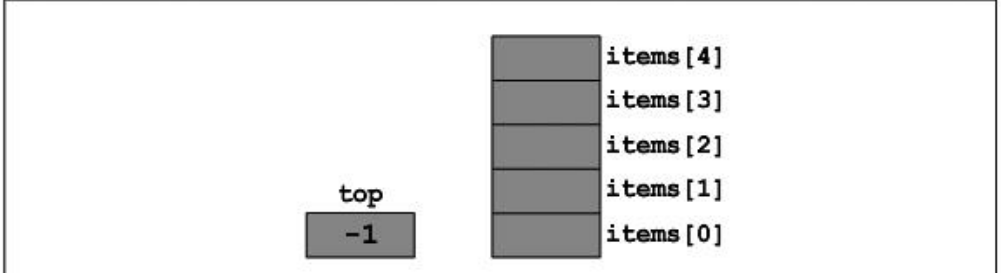
```
#define SIZE 5
int top;
char items[SIZE];
```

ฟังก์ชัน initialize_stack()

ฟังก์ชัน initialize_stack() ใช้สำหรับกำหนดรูปแบบและค่าเริ่มต้นของสแตค

ตัวอย่าง การกำหนดรูปแบบและค่าเริ่มต้นของสแตค

```
#define SIZE 5
int top;
char items[SIZE];
void initialize_stack(){
    top = -1;
}
```

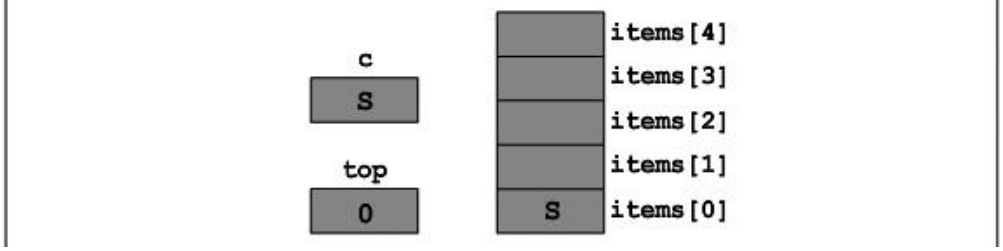


ฟังก์ชัน push_stack()

ฟังก์ชัน push_stack() ใช้สำหรับเพิ่มข้อมูลเข้าสู่สแตค

ตัวอย่าง การเพิ่มข้อมูลเข้าสู่สแตค

```
void push_stack(char c){
    top++;
    items[top] = c;
}
```

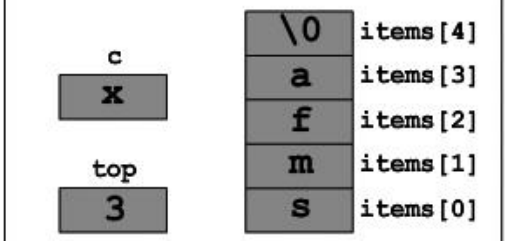


ฟังก์ชัน pop_stack()

ฟังก์ชัน pop_stack() ใช้สำหรับดึงข้อมูล ลบข้อมูล และเปลี่ยนตำแหน่งของตัวแปร top ในสแตค

ตัวอย่าง การดึงข้อมูล ลบข้อมูล และเปลี่ยนตำแหน่งของตัวแปร top ในสแตค

```
char pop_stack(){
    char c;
    c = items[top];
    items[top]='\0';
    top--;
    return c;
}
```

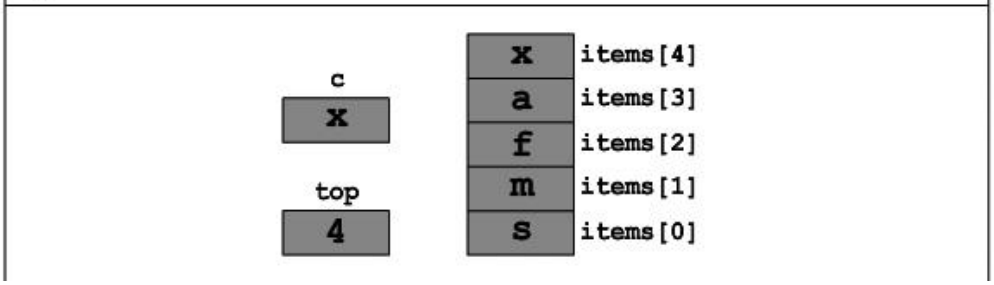


ฟังก์ชัน stack_top()

ฟังก์ชัน stack_top() ใช้สำหรับทำสำเนาข้อมูลที่อยู่ตำแหน่ง top ในสแตค โดยไม่ลบข้อมูลนั้น

ตัวอย่าง การทำสำเนาข้อมูลจากตำแหน่ง top ในสแตค โดยไม่ลบข้อมูล

```
char stack_top(){
    char c;
    c = items[top];
    return c;
}
```

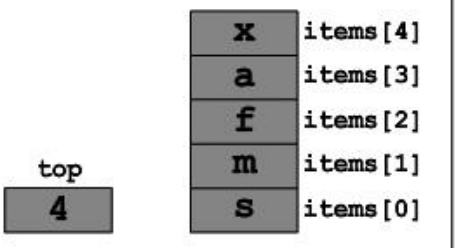


ฟังก์ชัน empty_stack()

ฟังก์ชัน empty_stack() ใช้สำหรับตรวจสอบสแตคว่าเป็นสแคตว่าง (ไม่มีข้อมูล) หรือไม่?

ตัวอย่าง การตรวจสอบสแตคว่าเป็นสแตคว่าง (ไม่มีข้อมูล) หรือไม่?

```
int empty_stack(){
    if(top == -1)
        return 1;
    else
        return 0;
}
```

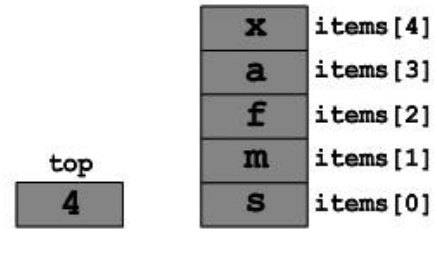


ฟังก์ชัน full_stack()

ฟังก์ชัน full_stack() ใช้สำหรับตรวจสอบสแตคว่าเป็นสแคตที่มีข้อมูลเต็มหรือไม่?

ตัวอย่าง การตรวจสอบสแตคว่าเป็นสแตคที่มีข้อมูลเต็มหรือไม่?

```
int full_stack(){
    if(top == SIZE-1)
        return 1;
    else
        return 0;
}
```

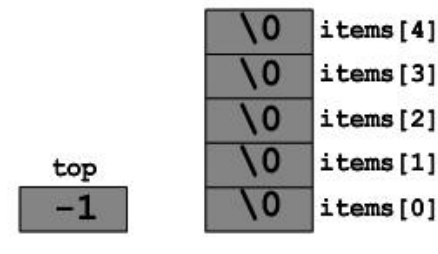


ฟังก์ชัน destroy_stack()

ฟังก์ชัน destroy_stack() ใช้สำหรับลบข้อมูลในสแตค

ตัวอย่าง การตรวจสอบสแตคว่าเป็นสแตคที่มีข้อมูลเต็มหรือไม่?

```
void destroy_stack(){
  while(top != -1){
    items[top]='\0';
    top--;
  }
}
```



Dynamic Implementations

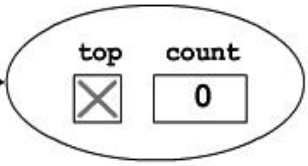
ประกาศโครงสร้างข้อมูลชื่อว่า node ประกอบไปด้วยสมาชิก 2 ตัวคือ
- c เป็นตัวแปรที่จะเก็บตัวอักษร 1 ตัว
- next เป็นตัวแปรพอยน์เตอร์ที่จะชี้ไปที่โครงสร้างข้อมูลแบบ node

ประกาศตัวแปร top เป็นตัวแปรพอยน์เตอร์โกลบอลแบบ node และกำหนดค่าเริ่มต้นให้เท่ากับ NULL
ประกาศตัวแปร count เป็นตัวแปรโกลบอลที่จะเก็บเลขจำนวนเต็มและกำหนดค่าเริ่มต้นให้เท่ากับ 0

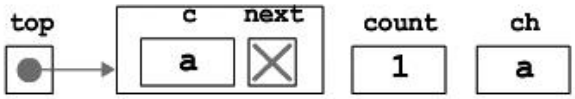
```
typedef struct nd{
    char c;
    struct nd *next;
}node;
node *top = NULL;
int count = 0;
```

Push Stack

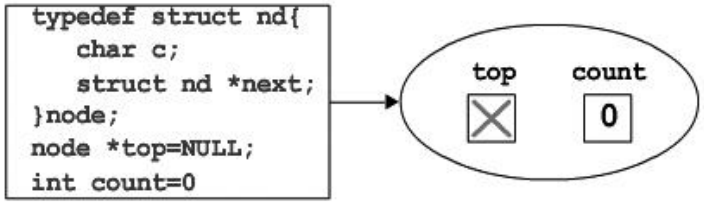
```
typedef struct nd{
   char c;
   struct nd *next;
}node;
node *top=NULL;
int count=0
```



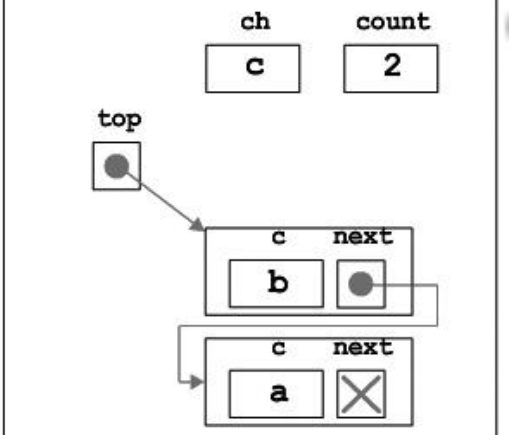

```
void push(char ch){
     node *n = malloc(sizeof(node));
     n->next = top;
     top = n;
     n->c = ch;
     count++;
}
```



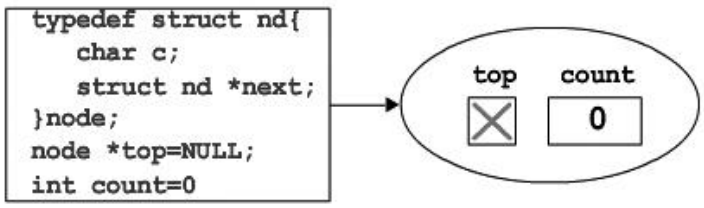
Pop Stack



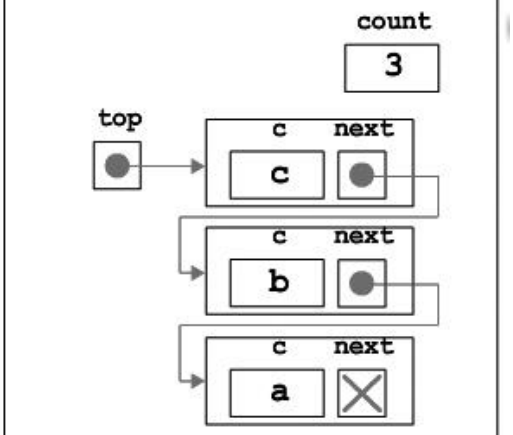

```
char pop(){
  char ch;
  node *n;
  n = top;
  top = top->next;
  ch = n->c;
  free(n);
  count--;
  return ch;
}
```



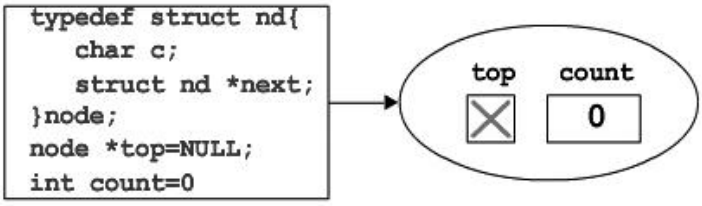
Stack Top



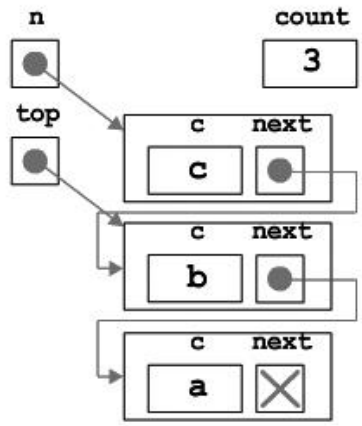

```
char stacktop() {
  if (top == NULL)
    printf("Empty");
  else
    return top->c;
}
```



Destroy Stack




```
void destroy(){
  node *n;
  while(top != NULL){
    n = top;
    top = top->next;
    free(n);
    count--;
  }
}
```



Stack Applications with Dynamic Implementations

- **Decimal to Binary**

ประกาศโครงสร้างข้อมูลชื่อว่า node ประกอบไปด้วยสมาชิก 2 ตัวคือ
num เป็นตัวแปรที่จะเก็บเลขจำนวนเต็ม
next เป็นตัวแปรพอยน์เตอร์ที่จะชี้ไปที่โครงสร้างข้อมูลแบบ node
ประกาศตัวแปร top เป็นตัวแปรพอยน์เตอร์โกลบอลแบบ node และกำหนดค่าเริ่มต้นให้เท่ากับ NULL
ประกาศตัวแปร count เป็นตัวแปรโกลบอลที่จะเก็บเลขจำนวนเต็มและกำหนดค่าเริ่มต้นให้เท่ากับ 0

```
typedef struct nd{
    int num;
    struct nd *next;
}node;
node *top = NULL;
int count = 0;
```

```c
typedef struct nd{
  int num;
  struct nd *next;
}node;
node *top = NULL;
int count=0;
```

1

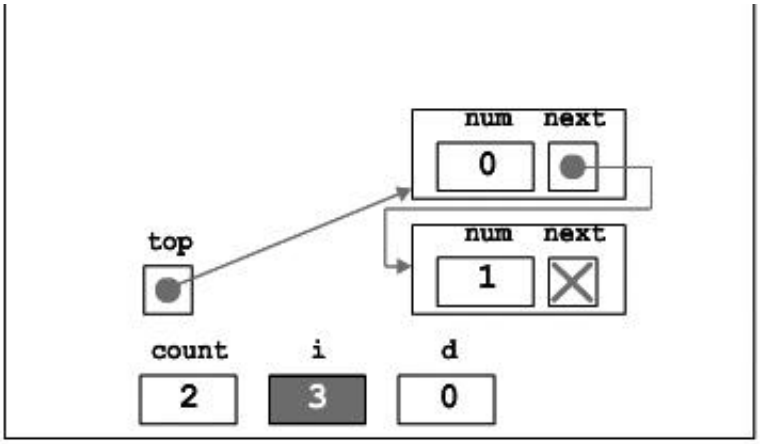
```c
void push(int x){
  node *n=malloc(sizeof(node));
  n->next = top;
  top = n;
  n->num = x;
  count++;
}
```

2

```c
void main(){
  int i,d;
  printf("Enter decimal number ");
  scanf("%d",&i);
  while(i>1){
    d=i%2;
    i=i/2;
    push(d);
  }
  push(i);
  printf("\nBinary number is ");
  while(top!=NULL)
    printf("%d ",pop());
}
```

```c
char pop(){
  int b;
  node *n;
  n = top;
  top = top->next;
  b = n->num;
  free(n);
  count--;
  return b;
}
```

3



- **Infix to Postfix**

ประกาศโครงสร้างข้อมูลชื่อว่า node ประกอบไปด้วยสมาชิก 2 ตัวคือ
- c เป็นตัวแปรที่จะเก็บตัวอักษร 1 ตัว
- next เป็นตัวแปรพอยน์เตอร์ที่จะชี้ไปที่โครงสร้างข้อมูลแบบ node

ประกาศตัวแปร top เป็นตัวแปรพอยน์เตอร์โกลบอลแบบ node และกำหนดค่าเริ่มต้นให้เท่ากับ NULL

```
typedef struct nd{
    char c;
    struct nd *next;
}node;
node *top = NULL;
```

```
#include<stdio.h>
#include<stdlib.h>
typedef struct nd{
  char c;
  struct nd *next;
}node;
node *top = NULL;

void push(char x){
  node *n = malloc(sizeof(node));
  n->next = top;
  top = n;
  n->c = x;
}

char pop(){
  char p;
  node *n;
  n = top;
  top = top->next;
  p = n->c;
  free(n);
  return p;
}
char stacktop() {
  if(top == NULL)
    return NULL;
  else
    return top->c;
}

int checkpr(char temp){
  int pr;
  if(temp=='*' || temp=='/')
    pr=2;
  else
    pr=1;
  return pr;
}

void checkoper(char ck){
  if(stacktop()==NULL)
    push(ck);
  else{
    if(checkpr(ck)<=checkpr(stacktop()))

while((checkpr(ck)<=checkpr(stacktop()))&&
      (stacktop()!=NULL))
```

```
        printf("%c ",pop());
    push(ck);
  }
}

void main(){
  char ch;
  while((ch = getchar()) != '\n'){
    if(isdigit(ch)||isalpha(ch))
      printf("%c ",ch);
    else
      checkoper(ch);
  }
  while(stacktop()!=NULL)
    printf("%c ",pop());
}
```