

# DATA STRUCTURES AND ALGORITHMS

## CẤU TRÚC NGĂN XẾP (STACK)

# Nội dung

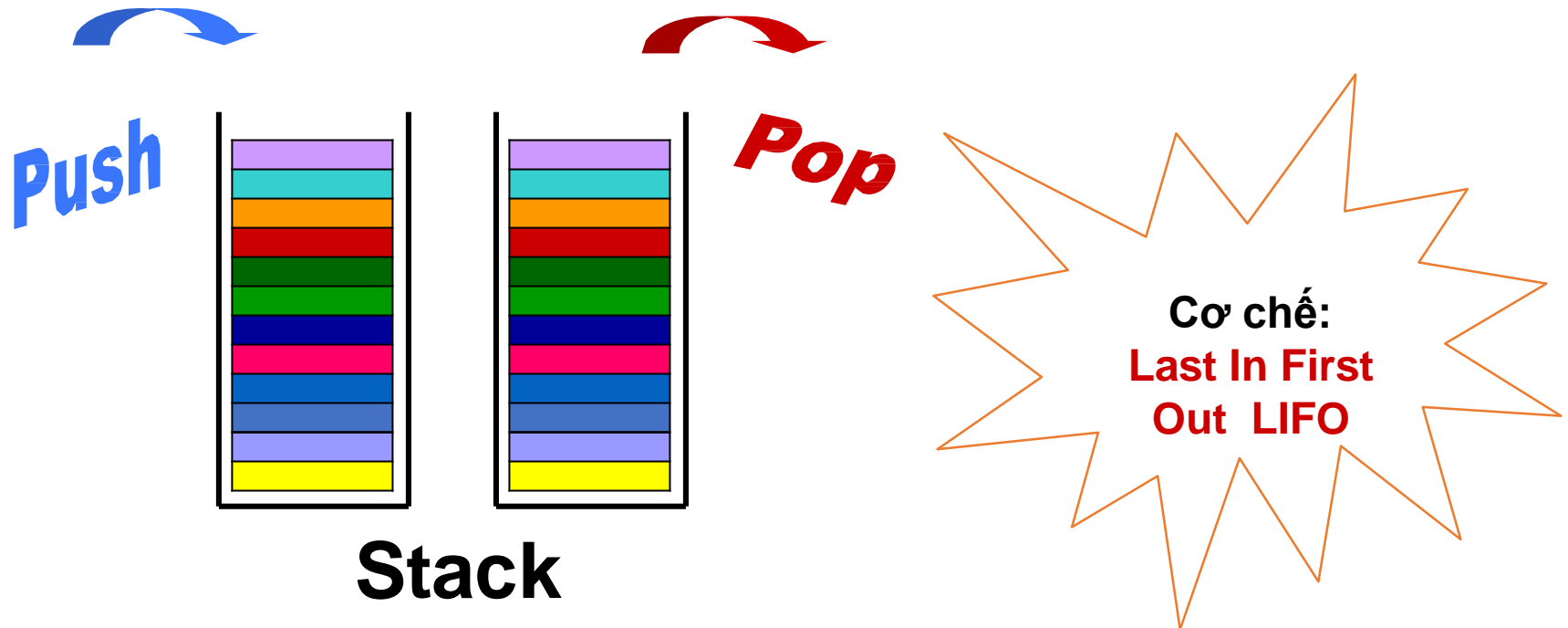
---

- Giới thiệu về stack
- Cài đặt stack
- Ứng dụng

# GIỚI THIỆU VỀ STACK

# Stack là gì?

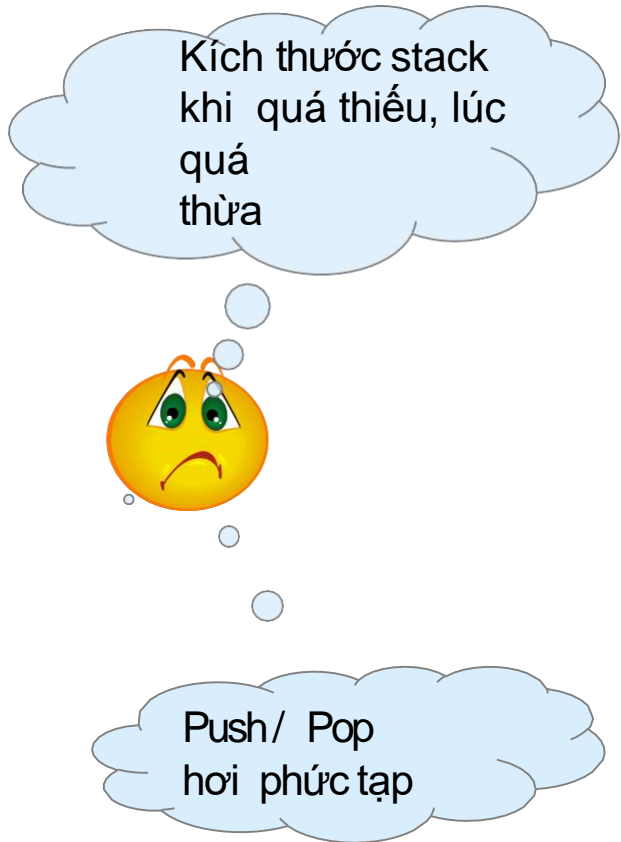
- Stack là một danh sách đặc biệt mà việc thêm vào và loại bỏ được thực hiện tại một đầu (gọi là đỉnh – top của stack).
- Hoạt động theo cơ chế LIFO (Last In First Out)



CÀI ĐẶT STACK

# Phương pháp cài đặt

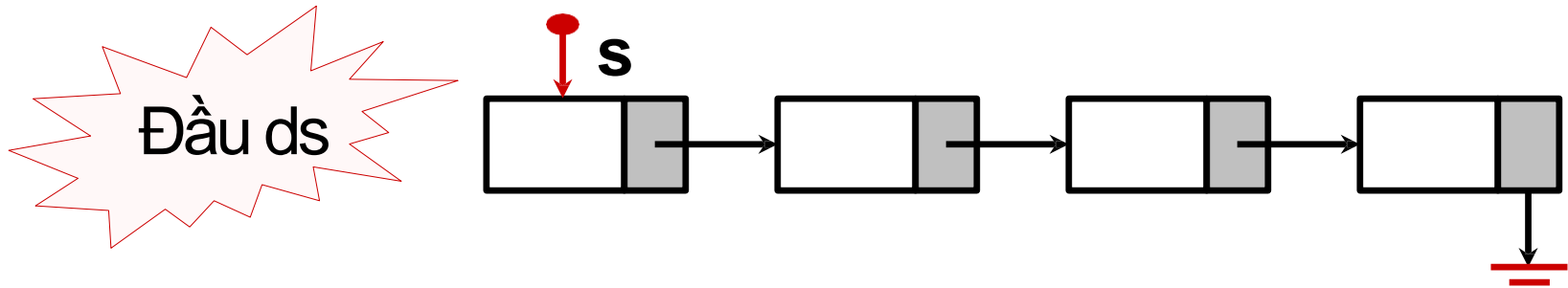
## Mảng 1 chiều



## Danh sách liên kết đơn



# Cài đặt stack dùng DSLK đơn



## Khai báo

```
struct node
```

```
{
```

```
    DataType data;
```

```
    node *next;
```

```
};
```

```
typedef node* stack;
```

## Sử dụng

```
stack s;
```

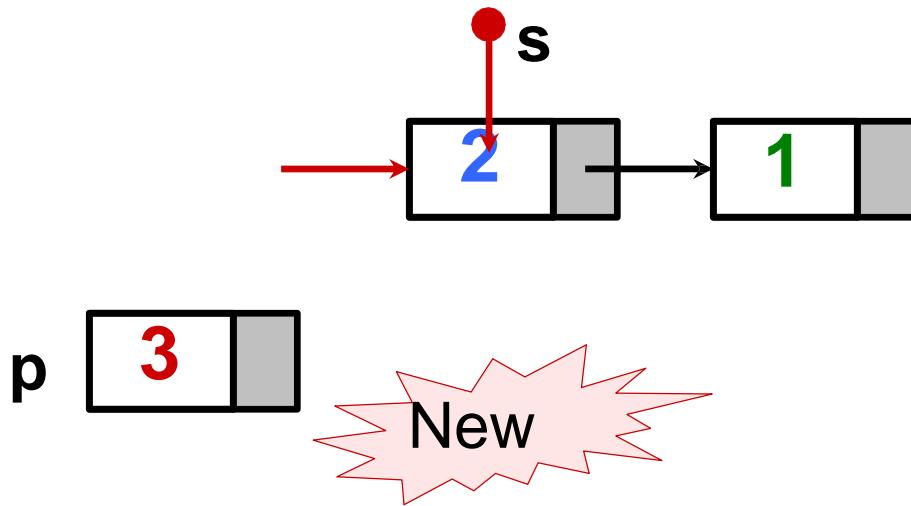
# Các thao tác trên stack

---

- Khởi tạo: `init`
- Kiểm tra rỗng: `isEmpty`
- Thêm một phần tử vào stack: `push`
- Lấy một phần tử ra khỏi stack: `pop`
- Xem nội dung của phần tử đầu tiên trong stack: `peek`

# Thêm phần tử x vào stack- push

- Tạo nút mới có dữ liệu là x
- Thêm nút vừa tạo vào đầu stack



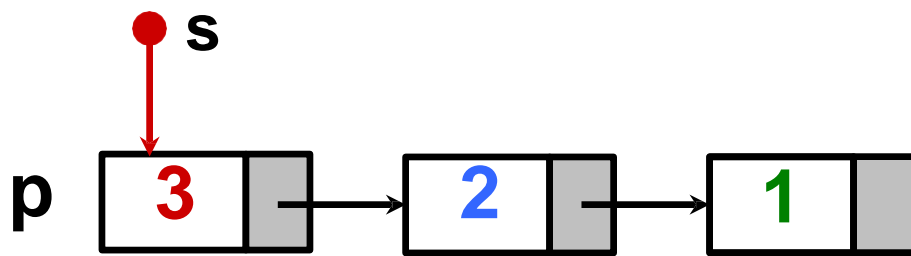
```
void Push(stack &s, DataType x) {...}
```

```
// khai bao cau truc stack
struct node
{
    int data;
    node *next;
};
typedef node* stack;

void push(stack &s, int x)
{
    node* p = new node;
    p->data = x;
    p->next = s;
    s = p;
}
```

# Lấy một phần tử ra khỏi stack - pop

- Lấy ra phần tử đầu danh sách
- Trả về nội dung và giải phóng nút



X  
=

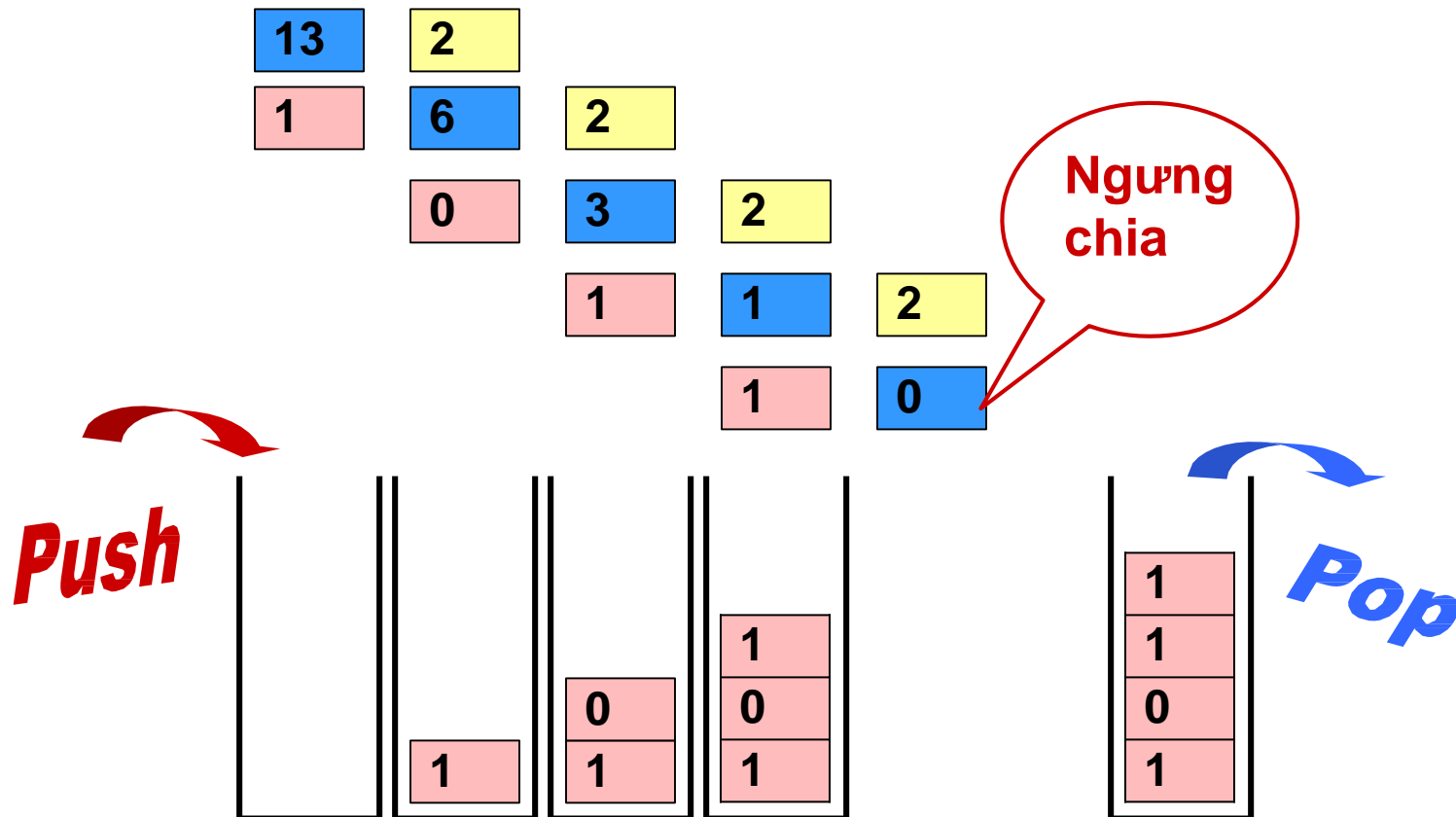
```
int Pop(stack &s, DataType &x) {...}
```

```
int pop(stack &s, int &x)
{
    if (isEmpty(s))
        return 0;
    node *p = s;
    x = s->data;
    s = s->next;
    delete p;
    return 1;
}
```

ỨNG DỤNG

# Bài toán chuyển đổi hệ đếm

- Phát biểu bài toán: Viết chương trình đổi một số thập phân sang cơ số nhị phân vận dụng stack



# Bài toán chuyển đổi hệ đếm

---

```
stack convertBase(int x, int b)
{
    int r; // lưu phần dư của phép chia
    stack s; // lưu kết quả chuyển đổi
    init(s);

    do
    {
        r = x % b;
        push(s, r);
        x = x / b;
    } while (x != 0);

    return s;
}
```

# Bài toán chuyển đổi dạng của biểu thức

---

- Viết chương trình cài đặt bài toán chuyển biểu thức trung tố sang hậu tố và tiền tố
- Sau đó tính giá trị biểu thức hậu tố và biểu thức tiền tố.

# Trung tố, tiền tố, và hậu tố

---

- Trung tố (infix), hậu tố (postfix, ký pháp Ba Lan ngược), tiền tố (prefix, ký pháp Ba Lan) là các cách biểu diễn khác nhau của biểu thức toán học.
- Trung tố: cách biểu diễn thông dụng
  - Toán tử được viết giữa các toán hạng.
  - Phép toán có thứ tự ưu tiên.
  - Thứ tự thực hiện phép toán từ trái qua phải nếu phép toán có cùng thứ tự ưu tiên.
  - Sử dụng dấu ngoặc đơn ( ) để thay đổi thứ tự thực hiện của phép toán.
  - Ví dụ:  $A * (B + C) / D$

# Trung tố, tiền tố, và hậu tố

---

- Tiền tố (Ký pháp Ba Lan)
  - Toán tử được viết trước toán hạng.
  - Thứ tự thực hiện phép toán từ trái qua phải.
  - Không sử dụng dấu ngoặc đơn () để thay đổi thứ tự thực hiện của phép toán.
  - Phép toán được thực hiện trên các toán hạng nằm ngay sau phép toán.
  - Ví dụ:  $/ * A + B C D$

# Trung tố, tiền tố, và hậu tố

---

- Hậu tố (Ký pháp Ba Lan ngược)
  - Toán tử được viết sau các toán hạng.
  - Thứ tự thực hiện phép toán từ trái qua phải
  - Không sử dụng dấu ngoặc đơn () để thay đổi thứ tự thực hiện của phép toán.
  - Phép toán được thực hiện trên các toán hạng nằm ngay trước phép toán.
  - Ví dụ:  $ABC + * D /$

# Ví dụ về trung tố, tiền tố, và hậu tố

Trung tố	Tiền tố (ký pháp Ba Lan)	Hậu tố (ký pháp Ba Lan ngược)
$A * B + C / D$	$+ * A B / C D$	$A B * C D / +$
$A * (B + C) / D$	$/ * A + B C D$	$A B C + * D /$
$A * (B + C / D)$	$* A + B / C D$	$A B C D / + *$

**THUẬT TOÁN CHUYỂN BIỂU**  
**THỨC DẠNG TRUNG TỔ**  
**SANG HẬU TỔ**

# Thuật toán chuyển biểu thức dạng **trung tố sang hậu tố**

1. Khởi tạo:
  - Biến `output = []` dùng để chứa biểu thức kết quả
  - Stack `s`
2. Duyệt biểu thức từ trái qua phải.
3. Nếu ký tự đang xét là toán hạng, thêm ký tự đó vào `output`.
4. Nếu ký tự đang xét là '(', thêm ký tự đó vào stack `s`.
5. Nếu ký tự đang xét là ')', lấy một phần tử ra khỏi stack `s` và thêm phần tử đó vào `output` cho đến khi gặp dấu '('.  
Loại bỏ dấu '(' vừa gặp khỏi stack.

## Thuật toán chuyển biểu thức dạng **trung tố** sang **hậu tố**

6. Nếu ký tự đang xét là toán tử:
  1. Trong khi đỉnh **Stack** là toán tử và có độ ưu tiên  $\geq$  độ ưu tiên của toán tử đang xét thì lấy toán tử trong **Stack** ra và thêm toán tử này vào output. *(Lặp lại 6.1 tới khi phần tử đầu của stack không thỏa mãn điều kiện vừa nêu)*
  2. Thêm ký tự đang xét vào stack s.
7. Lặp lại bước 3-6 cho tới khi duyệt tới phần tử cuối cùng của biểu thức.
8. Sau khi duyệt hết biểu thức, nếu trong stack còn phần tử thì lấy lần lượt các phần tử ra và thêm vào output.

## Thuật toán chuyển biểu thức dạng trung tố sang hậu tố

Ví dụ: Chuyển biểu thức  $\text{exp} = A*(B+C)$  sang dạng hậu tố

Khởi tạo:  $\text{output} = []$

$\text{stack s} = \text{NULL}$

i	exp[i]	output	S
0	A	A	NULL
1	*	A	*
2	(	A	(*
3	B	AB	(*
4	+	AB	+(*
5	C	ABC	+(*
6	)	ABC+	*

Bước 8: **output =ABC+\***

## Thuật toán chuyển biểu thức dạng trung tố sang hậu tố

Ví dụ: Chuyển biểu thức  $\text{exp} = A * B ^ C + D$  sang dạng hậu tố

Khởi tạo:  $\text{output} = []$

$\text{stack } s = \text{NULL}$

i	exp[i]	output	S
0	A	A	NULL
1	*	A	*
2	B	AB	*
3	^	AB	^ *
4	C	ABC	^ *
5	+	ABC ^ *	+
6	D	ABC ^ * D	+

Bước 8: **output = ABC ^ \* D +**

## Thuật toán chuyển biểu thức dạng trung tố sang hậu tố

Ví dụ: Chuyển biểu thức  $\text{exp} = A*(B+C*D)-E$  sang dạng hậu tố

Đáp án: **output = ABCD\*+\*E-**

## Thuật toán chuyển biểu thức dạng trung tố sang hậu tố

Ví dụ: Chuyển biểu thức  $\text{exp} = A * (B + C * D) - E$  sang dạng hậu tố

[illegible]

**THUẬT TOÁN CHUYỂN BIỂU**  
**THỨC DẠNG TRUNG TỐ**  
**SANG TIỀN TỐ**

# Thuật toán chuyển biểu thức dạng **trung tố sang tiền tố**

1. Đảo ngược biểu thức trung tố.
2. Thay thế '(' bằng ')' và ')' bằng '(' trong biểu thức đảo ngược.
3. Thực hiện các bước tìm biểu thức hậu tố của biểu thức đảo ngược thu được ở bước 2 như sau:
  1. Khởi tạo:
    - Biến `output = []` dùng để chứa biểu thức kết quả
    - Stack `s`
  2. Duyệt biểu thức từ trái qua phải.
  3. Nếu ký tự đang xét là toán hạng, thêm ký tự đó vào `output`.

# Thuật toán chuyển biểu thức dạng **trung tố sang tiền tố**

## Bước 3 (tiếp):

4. Nếu ký tự đang xét là '(', thêm ký tự đó vào stack s.
5. Nếu ký tự đang xét là ')', lấy một phần tử ra khỏi stack s và thêm phần tử đó vào output cho đến khi gặp dấu '('. Loại bỏ dấu '(' vừa gặp khỏi stack.
6. Nếu ký tự đang xét là toán tử:
  1. Lấy phần tử đầu ra khỏi stack s và thêm phần tử đó vào output nếu phần tử ở đỉnh stack là toán tử và toán tử đó có độ ưu tiên *lớn hơn* toán tử đang xét. (*Lặp lại 3.6.1 tới khi phần tử đầu của stack không thỏa mãn điều kiện vừa nêu*).
  2. Thêm ký tự đang xét vào stack s.

# Thuật toán chuyển biểu thức dạng **trung tố** sang **tiền tố**

## Bước 3 (tiếp):

7. Lặp lại bước 3.3-3.6 cho tới khi duyệt tới phần tử cuối cùng của biểu thức.
  8. Sau khi duyệt hết biểu thức, nếu trong stack còn phần tử thì thấy lần lượt các phần tử ra và thêm vào output.
- 
4. Đảo ngược biểu thức hậu tố thu được ở bước 3.
  5. Kết quả của bước 4 là biểu thức tiền tố cần tìm.

# Thuật toán chuyển biểu thức dạng trung tố sang tiền tố

Ví dụ: Chuyển biểu thức  $\text{exp} = A*(B+C)$  sang dạng tiền tố

Bước 1:  $\text{rev} = )C+B(*A$

Bước 2:  $\text{rev} = (C+B)*A$

i	exp[i]	output	S
0	(	[]	(
1	C	C	(
2	+	C	+(
3	B	CB	+(
4	)	CB+	
5	*	CB+	*
6	A	CB+A	*

Bước 3: thu được biểu thức hậu tố:  $CB+A*$

Bước 4: **biểu thức tiền tố cần tìm =  $*A+BC$**

**Thuật toán chuyển biểu thức dạng trung tố sang tiền tố**

Ví dụ: Chuyển biểu thức  $exp = A*(B+C)$  sang dạng tiền tố

Bước 1: rev =

Bước 2: rev =


Bước 3: thu được biểu thức hậu tố:

Bước 4: **biểu thức tiền tố cần tìm =**

## Thuật toán chuyển biểu thức dạng trung tố sang tiền tố

Ví dụ: Chuyển biểu thức  $\text{exp} = A*B+C/D$  sang dạng tiền tố

Bước 1:  $\text{rev} = D/C+B*A$

Bước 2:  $\text{rev} = D/C+B*A$

i	exp[i]	output	S
0	D	D	NULL
1	/	D	/
2	C	DC	/
3	+	DC/	+
4	B	DC/B	+
5	*	DC/B	*+
6	A	DC/BA	*+

Bước 3: thu được biểu thức hậu tố:  $DC/BA*+$

Bước 4: **biểu thức tiền tố cần tìm =  $+*AB/CD$**

## Thuật toán chuyển biểu thức dạng trung tố sang tiền tố

**Ví dụ:** Chuyển biểu thức  $\text{exp} = (A-B/C)*(A/K-L)$  sang dạng tiền tố

Đáp án:  $*-A/BC-/AKL$

**Ví dụ:** Chuyển biểu thức  $\text{exp} = A*(B+C*D)-E$  sang dạng tiền tố

Đáp án:  $-*A+B*CDE$

# Một số ví dụ

---

- $a / (b - c + d) * (e - a) * c$
- $a / b - c + d * e - a * c$
- $a + b * c - d / e * f$
- $(a + b * c - d) / (e * f)$
- $(7 / (5 - 3) * 9 + 2)$

$$P = (7 / (5 - 3) * 9 + 2)$$

[illegible]

# TÍNH GIÁ TRỊ BIỂU THỨC HẬU TỔ

# Thuật toán

---

1. Khởi tạo stack  $s = \text{NULL}$ .
2. Duyệt biểu thức hậu tố từ trái qua phải.
3. Nếu phần tử đang xét là toán hạng, thêm phần tử đó vào stack  $s$ .
4. Nếu phần tử đang xét là toán tử, lấy phần tử đầu tiên (a) và phần tử thứ 2 (b) ra khỏi stack và thực hiện phép toán giữa b và a (kq). Sau đó thêm kết quả của phép toán vào stack  $s$ .
5. Lặp lại bước 3 và 4 cho tới khi duyệt tới phần tử cuối cùng của biểu thức.
6. Kết quả của biểu thức hậu tố chính là phần tử đầu tiên trong stack  $s$  thu được sau bước 5.

# Ví dụ: Tính giá trị biểu thức $2578^*+^*9-$

Khởi tạo stack  $s = \text{NULL}$

i	exp[i]	a	b	Kết quả phép toán giữa b và a	S
0	2	-	-	-	2
1	5	-	-	-	5 2
2	7	-	-	-	7 5 2
3	8	-	-	-	8 7 5 2
4	*	8	7	$7 * 8 = 56$	56 5 2
5	+	56	5	$5 + 56 = 61$	61 2
6	*	61	2	$2 * 61 = 122$	122
7	9	-	-	-	9 122
8	-	9	122	$122 - 9$	<b>113</b>

# Ví dụ: Tính giá trị biểu thức $72*93/+$

Khởi tạo stack  $s = \text{NULL}$

i	exp[i]	a	b	Kết quả phép toán giữa b và a	S
0	7	-	-	-	7
1	2	-	-	-	2 7
2	*	2	7	$7 * 2 = 14$	14
3	9	-	-	-	9 14
4	3	-	-	-	3 9 14
5	/	3	9	$9 / 3 = 3$	3 14
6	+	3	14	$14 + 3 = 17$	<b>17</b>

# TÍNH GIÁ TRỊ BIỂU THỨC TIỀN TỔ

# Thuật toán

---

1. Khởi tạo stack  $s = \text{NULL}$ .
2. Duyệt biểu thức tiền tố từ phải qua trái.
3. Nếu phần tử đang xét là toán hạng, thêm phần tử đó vào stack  $s$ .
4. Nếu phần tử đang xét là toán tử, lấy phần tử đầu tiên (a) và phần tử thứ 2 (b) ra khỏi stack và thực hiện phép toán giữa a và b. Sau đó thêm kết quả của phép toán vào stack  $s$ .
5. Lặp lại bước 3 và 4 cho tới khi duyệt tới phần tử đầu tiên của biểu thức.
6. Kết quả của biểu thức tiền tố chính là phần tử đầu tiên trong stack  $s$  thu được sau bước 5.

# Ví dụ: Tính giá trị biểu thức $-*2+5*789$

Khởi tạo stack  $s = \text{NULL}$

i	exp[i]	a	b	Kết quả phép toán giữa a và b	S
8	9	-	-	-	9
7	8	-	-	-	8 9
6	7	-	-	-	7 8 9
5	*	7	8	$7 * 8 = 56$	56 9
4	5	-	-	-	5 56 9
3	+	5	56	$5 + 56 = 61$	61 9
2	2	-	-	-	2 61 9
1	*	2	61	$2 * 61$	122 9
0	-	122	9	$122 - 9 = 113$	<b>113</b>

# Ví dụ: Tính giá trị biểu thức $+^*72/93$

Khởi tạo stack  $s = \text{NULL}$

i	exp[i]	a	b	Kết quả phép toán giữa a và b	S
6	3	-	-	-	3
5	9	-	-	-	9 3
4	/	9	3	$9 / 3 = 3$	3
3	2	-	-	-	2 3
2	7	-	-	-	7 2 3
1	*	7	2	$7 * 2 = 14$	14 3
0	+	14	3	$14 + 3 = 17$	<b>17</b>

# Ví dụ: Tính giá trị biểu thức $+^*72/93$

---

Khởi tạo stack  $s = \text{NULL}$
