3.2 顺序栈

- 顺序栈是一个静态结构，需要提前分配好空间位置
- 进栈：MAXNUM是栈中最大元素个数，当栈中已经有这么多元素时，再进行进栈操作，就会产生溢出，称为上溢（overflow）
- 出栈：对出栈运算时，需要判断是否为空栈，否则同样会产生溢出，称为下溢（underflow）
- 写相应运算的算法时，注意判断栈满或栈空

栈的实现：顺序表示

```
1  typedef int DataType;
2  struct Stack
3  {
4          int MAX;  //最大容量
5          int top;      //栈顶指针
6          DataType *elem;   //存放元素的数组起始指针
7  };
8  typedef struct Stack *SeqStack //定义顺序栈类型
```

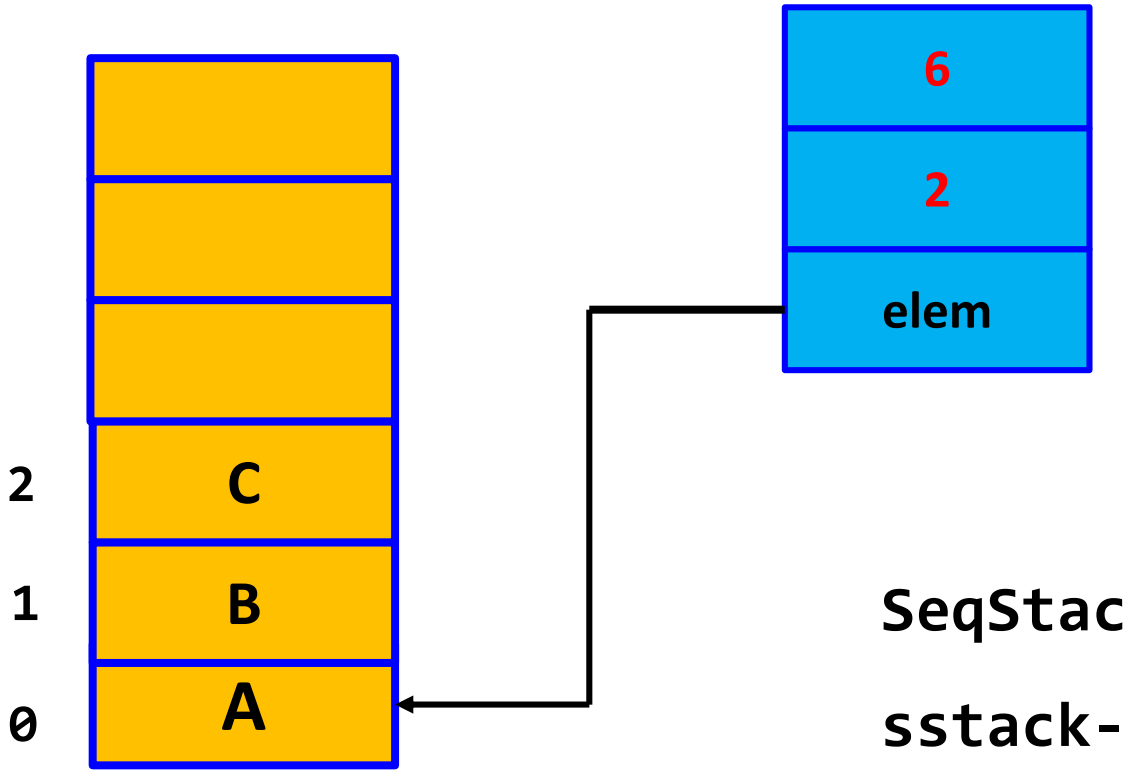
3.2.1 创建空栈

算法3-1

算法思路： 创建空的顺序栈就是为顺序栈分配一个预先定义的数组空间，并将顺序栈的栈顶top成员变量设置为-1。

```
1   SeqStack SetNullStack_Seq(int m)//创建空顺序栈，m是分配的最大空间
2   {
3           SeqStack sstack = (SeqStack)malloc(sizeof(struct SeqStack));
4           if(sstack!=NULL)
5           {
6                   sstack->elem = (int*)malloc(sizeof(int)*m);
7                   if(sstack->elem!=NULL)
8                    {
9                       sstack->MAX = m; //顺序栈最大容量
10                      sstack->top = -1; //设置栈顶初值为-1
11                      return(sstack);
12                  }
13                  else
14                 {
15                      free(sstack);  return NULL;
16                  }
17          }
18          else
19          {
20                  printf("out of space"); return NULL;
21           }
22  }
```

栈的实现：顺序表示



sstack->Max = 6

sstack->top = 2

SeqStack sstack　定义变量sstack

sstack->top　表示指向栈顶的变量

sstack->elem　表示存放栈中元素的数组起始地址

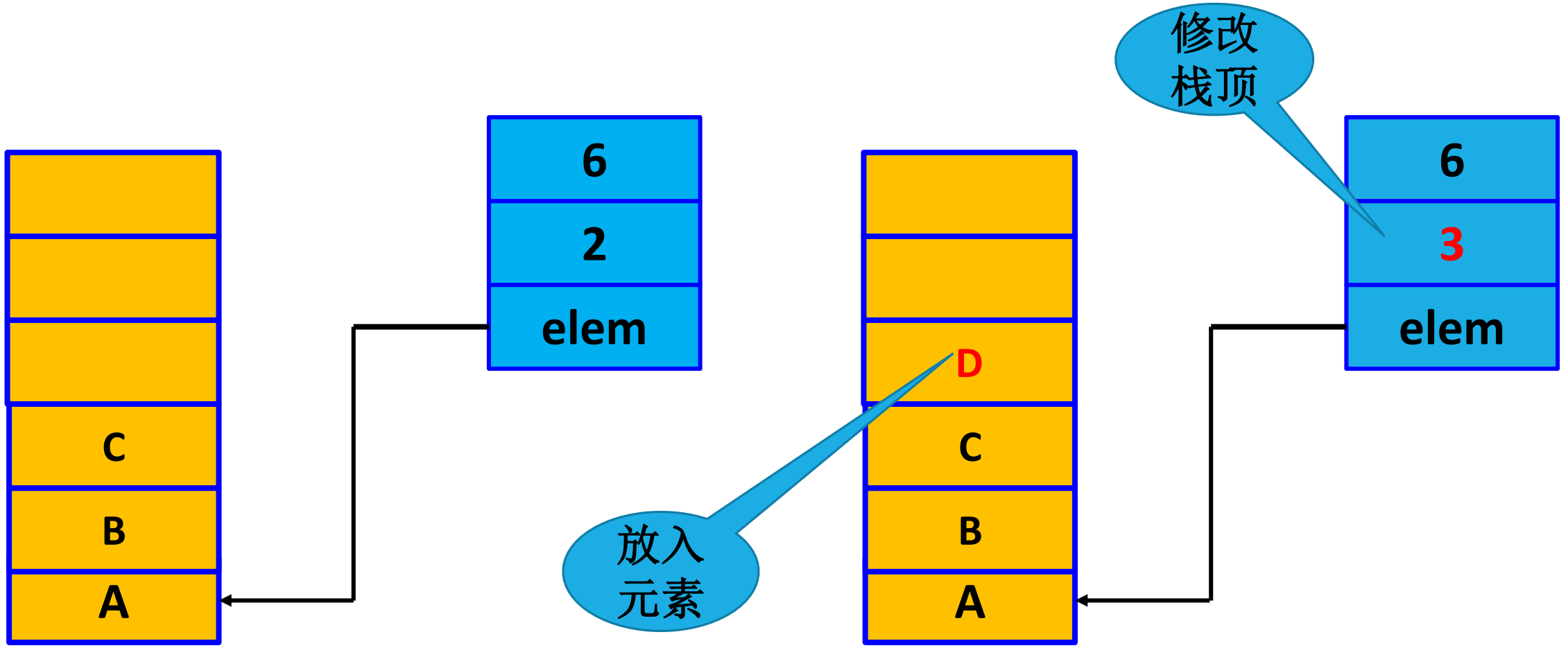
sstack->elem[sstack->t]　表示当前的栈顶元素

3.2.2 判断栈是否为空　算法3-2

算法思路：顺序栈的判空是检查栈顶指针是否等于初始化的-1，如果是-1，则返回1，否返回0

```
1  int IsNullStack_seq(SeqStack sstack) //判断一个栈是否为空
2  {
3          return(sstack->top==-1); //检查栈顶top
4  }
```

3.2.3 进栈

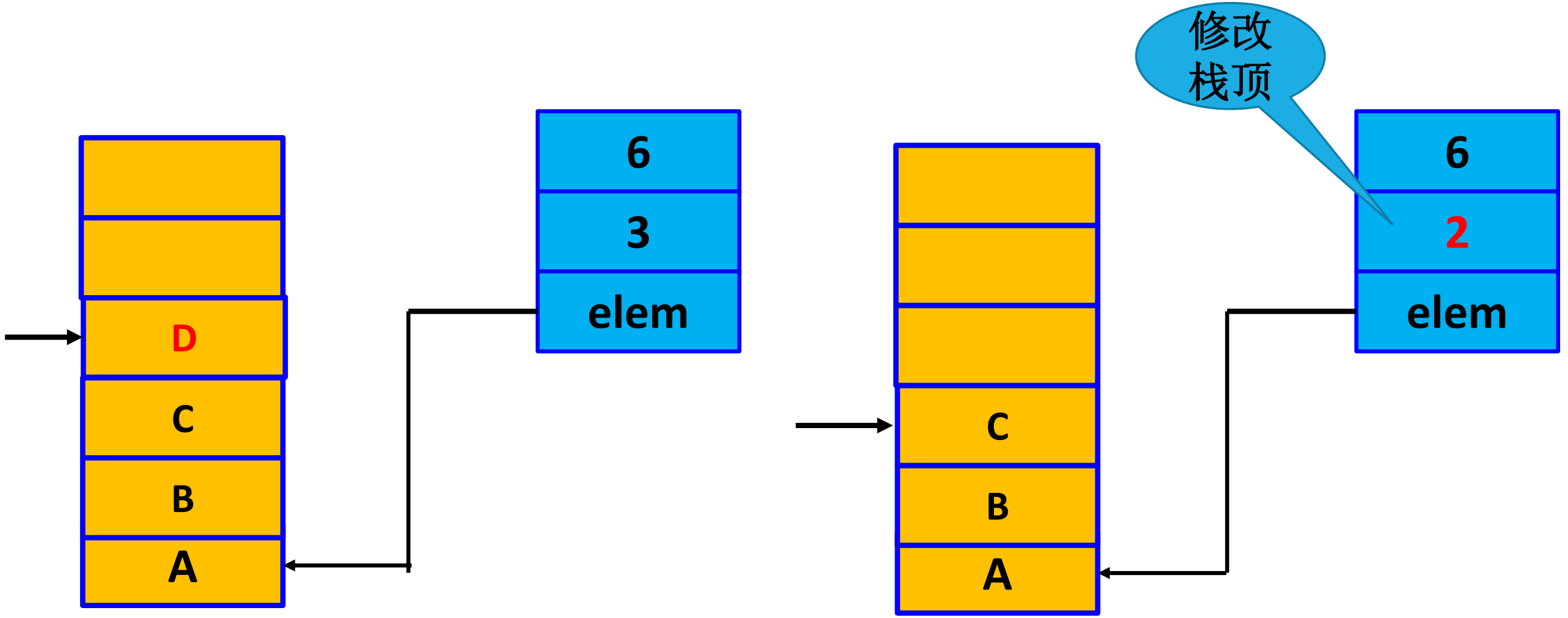


3.2.3 进栈 算法3-3

算法思路：首先检查栈是否满了，也就是检查栈顶是否已经达到了最大值，如果是，则不能再进行进栈操作，否则能够进栈。进栈时需要先修改栈顶，然后将元素压入栈中。

```
1  void Push_seq(SeqStack sstack,int x) //入栈
2  {
3          if( sstack->top>=(sstack->MAX-1))  //检查栈是否满
4                  printf( "overflow! \n" );
5          else
6          {
7                  sstack->top ++; //若不满，先修改栈顶变量
8                  sstack->elem[sstack->top] = x; //把元素x放到栈顶变量的位置中
9           }
10 }
```

3.2.4 出栈

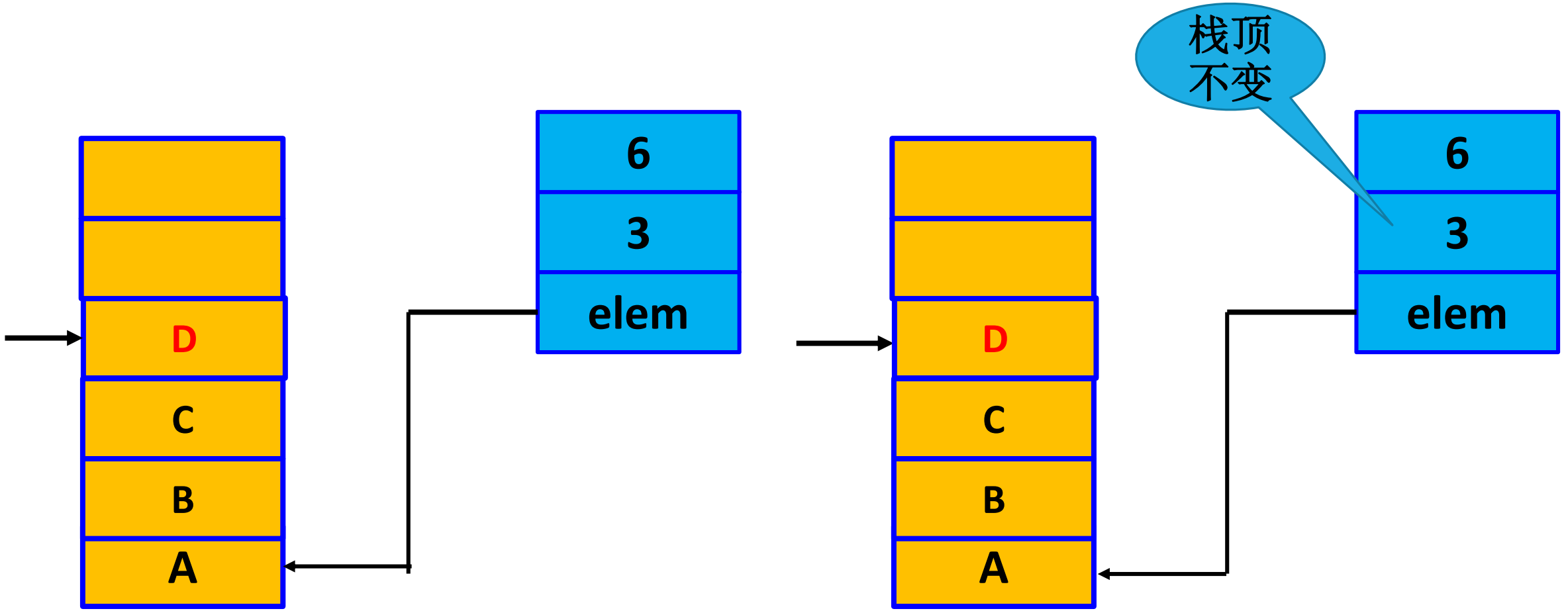


3.2.4 出栈　　算法3-4

算法思路：首先检查栈是否为空，如果是空栈，输出提示信息，不空，则栈顶指针减1。

```
1  void Pop_seq(SeqStack sstack)    //出栈
2  {
3          if (IsNullStack_seq(sstack))  //判断栈是否为空，调用算法3-2
4                  printf("Underflow!\n");
5          else
6                  sstack->top = sstack->top-1; //栈顶减1
7  }
```

3.2.5 取栈顶元素



3.2.5 取栈顶元素　算法3-5

算法思路：取顺序栈的栈顶元素时，首先检查是否为空栈，如果是空栈，输出提示信息；否则，返回栈顶元素。

```
1  DataType Top_seq(SeqStack sstack) //求栈顶元素的值
2  {
3    if (IsNullStack_seq(sstack)) //判断sstack所指的栈是否为空栈，调用算法3-2
4            printf("it is empty stack");
5   else
6      return sstack->elem[sstack->top];
7  }
```