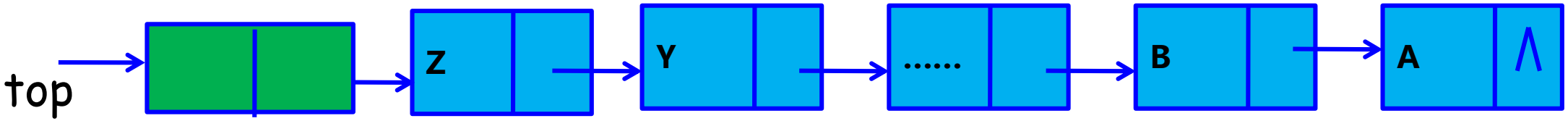
3.3 链栈

```
1  typedef int DataType;
2  struct Node
3  {
4          DataType        data;
5          struct Node*  next;
6  };
7  typedef struct Node  *PNode; //结点类型
8  typedef struct Node  *top, *LinkStack; //栈顶和链栈类型
```

注意：指针从栈顶开始，往栈底方向链接，有头结点



3.3.1 创建空栈　　算法3-6

算法思路：创建空的链栈，需要申请struct Node结构空间，并设置top->next为空，注意这里是带有头结点的空链栈。

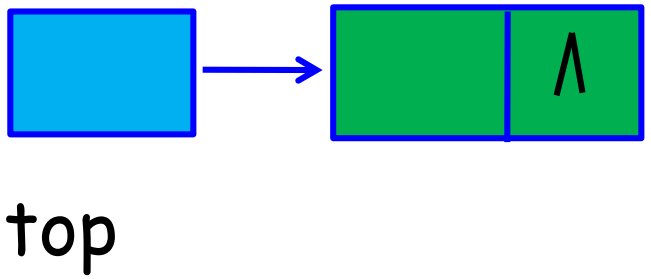
```
1  LinkStack SetNullStack_Link() //创建空链栈
2  {
3          LinkStack top = (LinkStack)malloc(sizeof(struct Node));
4          if (top!= NULL)
5              top->next = NULL;
6          else
7              printf("Alloc failure");
8          return top;    //返回栈顶指针
9  }
```

3.3.2 判断栈是否为空　算法3-7

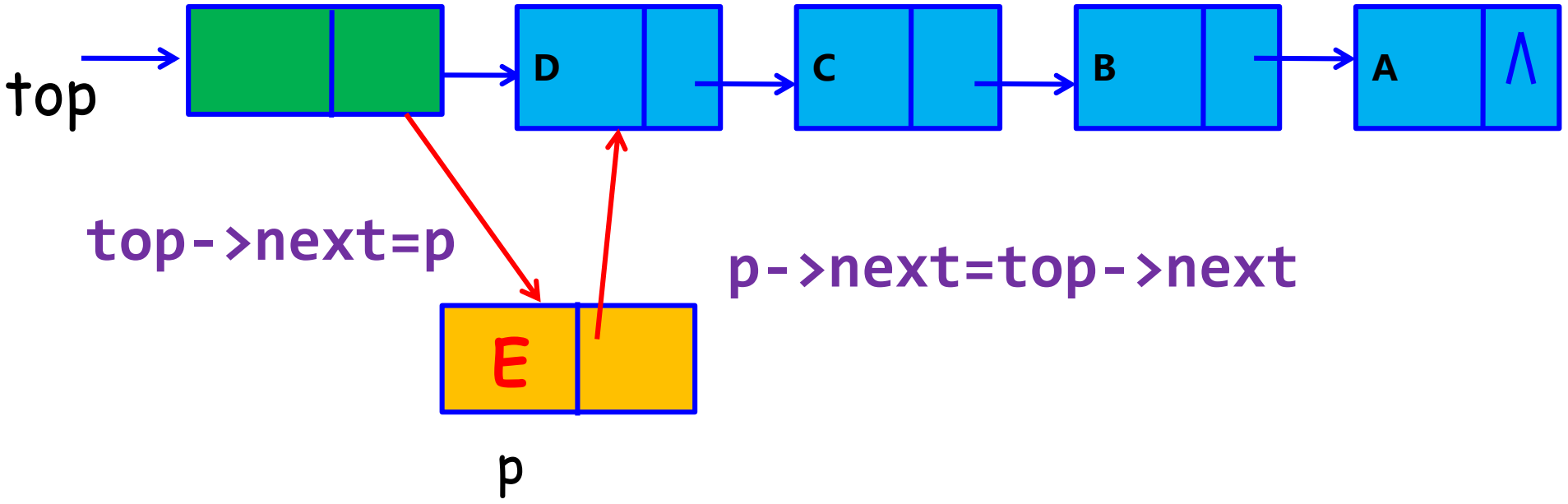
算法思路：判断链栈是否为空只需要栈顶结点的后继指针是否为空，空则返回1，否则返回0。

```
1  int IsNullStack_link(LinkStack top)//判断一个链栈是否为空
2  {
3          if (top->next == NULL)
4                  return 1;
5          else
6                  return 0;
7  }
```

3.3.3 进栈



3.3.3 进栈

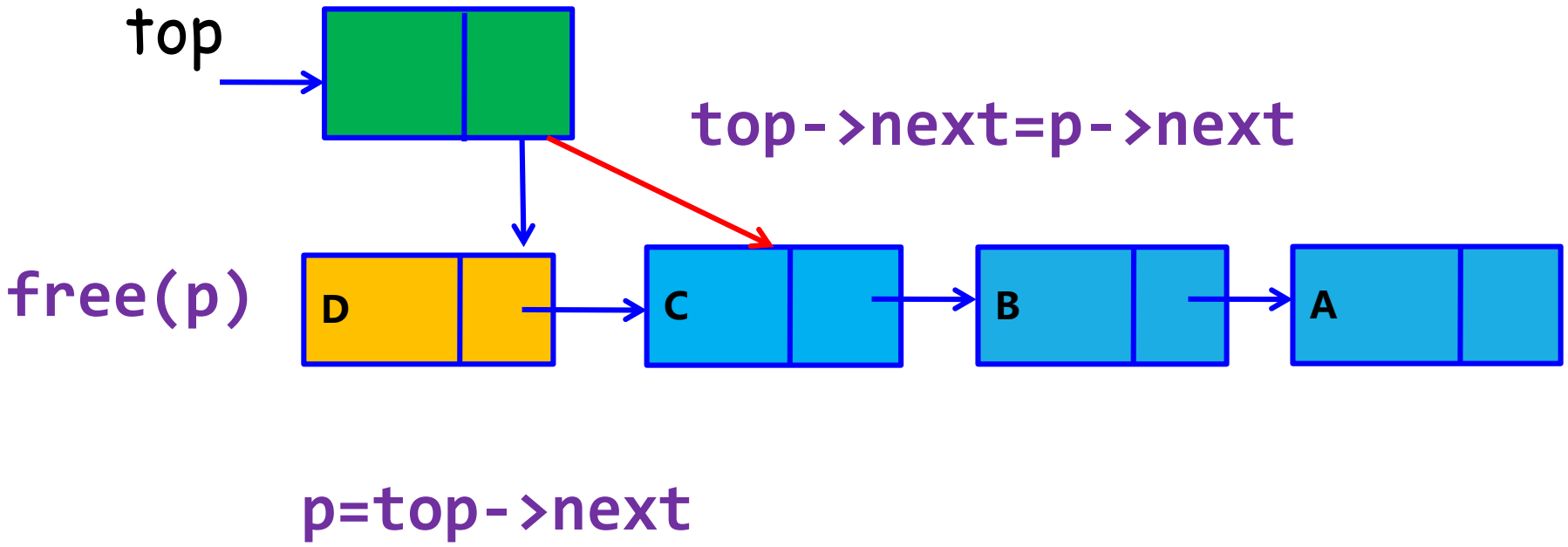


3.3.3 进栈　算法3-8

算法思路：将一个元素压入链栈中，首先要申请结点空间，然后给数据域和指针域赋值，并修改栈顶top的后继指针，将其赋值为新插入的结点。

```
1   void Push_link(LinkStack top, DataType x) //进栈
2   {
3           PNode p;
4           p = (PNode)malloc(sizeof(struct Node)); //申请结点空间
5           if (p == NULL)
6                   printf("Alloc failure");
7           else
8           {
9                   p->data = x;          //数据域赋值
10                  p->next = top->next;  //指针域赋值
11                  top->next = p;        //修改栈顶
12          }
13  }
```

3.3.4 出栈



3.3.4 出栈　　算法3-9

算法思路：出栈操作时，首先判断栈是否空，如果栈不空，修改栈顶指针，释放结点空间。

```
1   void Pop_link(LinkStack top)  // 删除栈顶元素
2   {
3           PNode p;
4           if (IsNullStack_link(LinkStack top))  //判断栈是否为空
5                   printf("it is empty stack!");
6           else
7           {
8                   p = top->next;          //p指向待删除结点
9                   top->next = p->next;  //修改栈顶指针
10                  free(p);                //释放删除结点空间
11          }
12  }
```

3.3.5 取栈顶元素

算法思路：先判断栈是否为空，如果栈不空，取栈顶元素的值，并返回。操作过程中栈结构保持不变。

```
1  DataType Pop_seq_return(LinkStack top)// 删除栈顶元素
2  {
3          if (IsNullStack_link(LinkStack top))  //判断栈是否为空
4                  printf("it is empty stack!");
5          else
6                  return top->next->data;
7  }
```