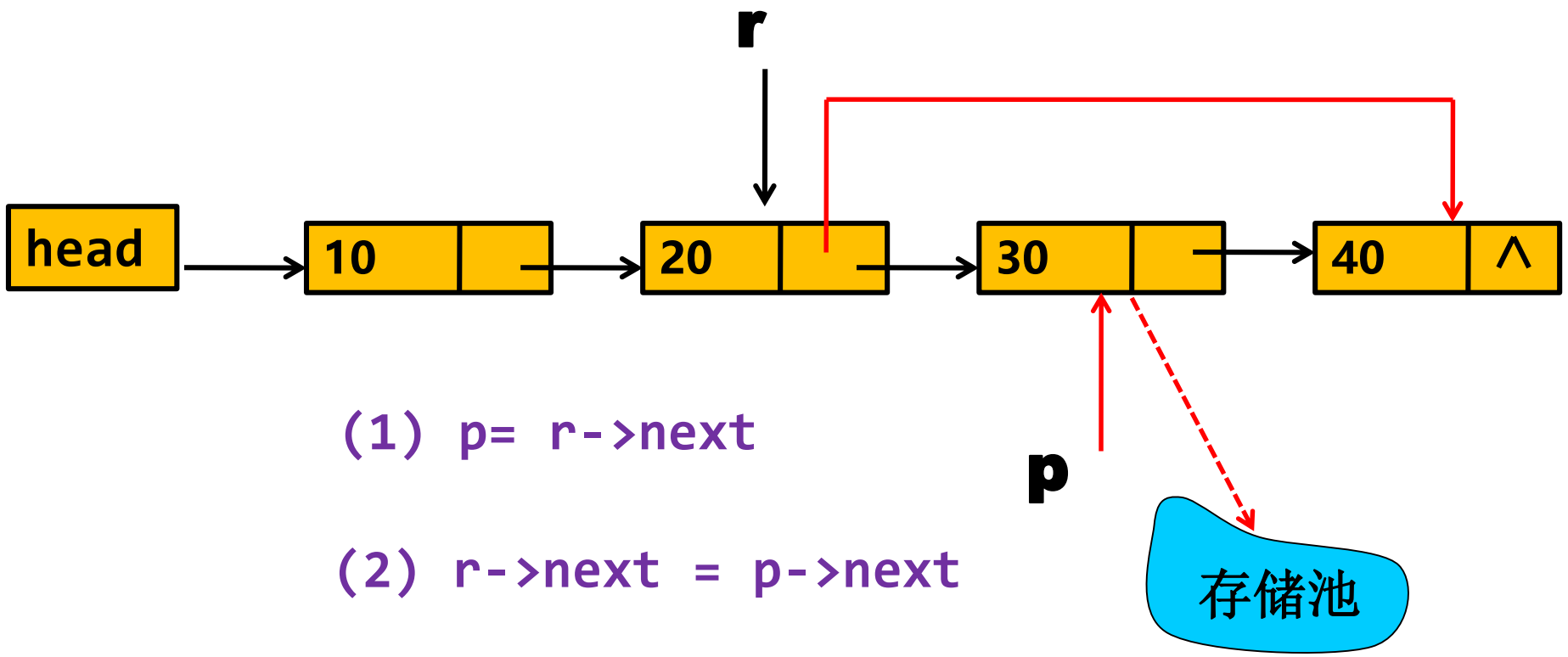
2.8 单链表删除：删除r的后继

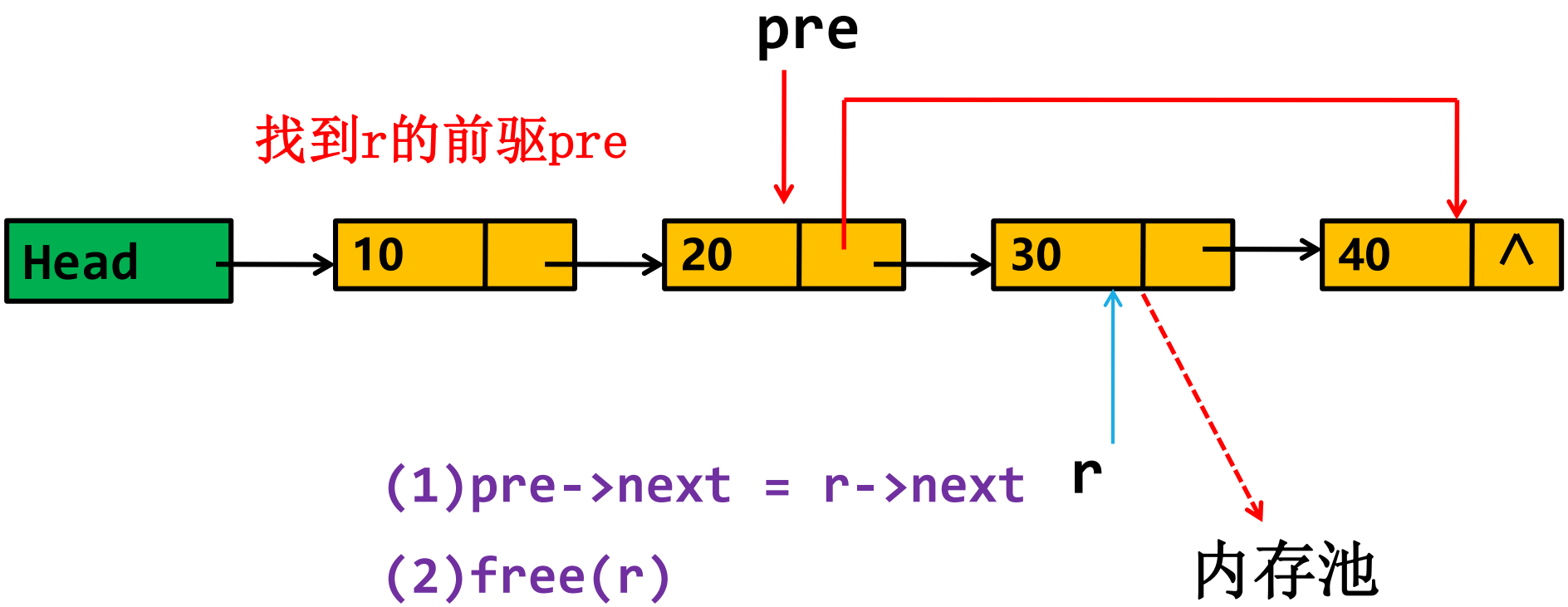



```
(1) p= r->next

(2) r->next = p->next

(3) free(p)
```

2.8 单链表删除：删除r本身




```
(1)pre->next = r->next
(2)free(r)
```

2.8 单链表删除：删除r本身

算法2-19

```
1   void DelPostion_Link(LinkList head, PNode r)
2   {//删除r指针所指结点
3           PNode pre=head;
4           while (pre->next != r) //定位r的前驱结点*/
5           {
6                   pre = pre->next;
7           }
8          pre->next = r->next;
9          free(r);
10  }
```

2.8.1 单链表删除：删除r的后继 算法2-20

```
1  //删除r指针所指结点的后继结点
2  void DelPostionNext_Link(LinkList head, PNode r)
3  {
4          PNode p;
5          if(r->next) p = r->next;
6             r->next = p->next;
7          free(p);
8  }
```

2.8.2 单链表删除：按值删除

算法2-21

```
1   //删除第一个与输入参数data相等的值的结点
2   void DelValue_Link(struct Node *head,int data)
3   {
4           struct Node* p=head->next;  struct Node* beforeP=head;
5           while (p!=NULL) {
6                 if (p->data==data)  //找到相等的进行删除
7                 {
8                           beforeP->next=p->next;
9                           free(p); break;
10                }
11              else    //否则继续向后移动查找
12                {
13                          beforeP=p; p=p->next;
14              }
15          }
16  }
```