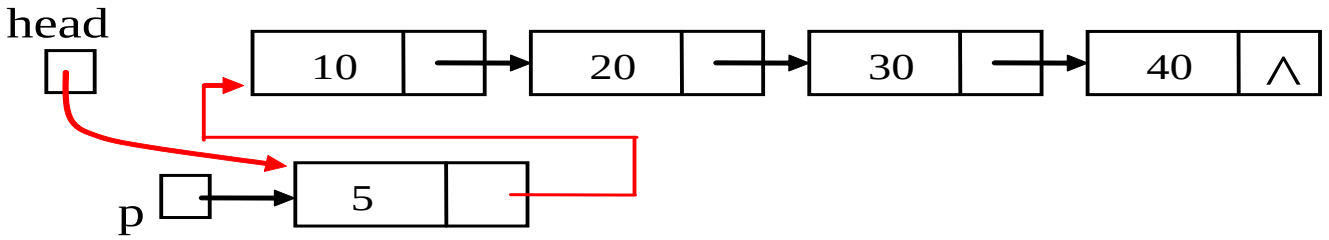
学习目标和要求

1.能够写出构造有序链表的算法。

问题分析

- 首先要找到插入位置，有三种情况：

1）小于表头结点，插在表头处；

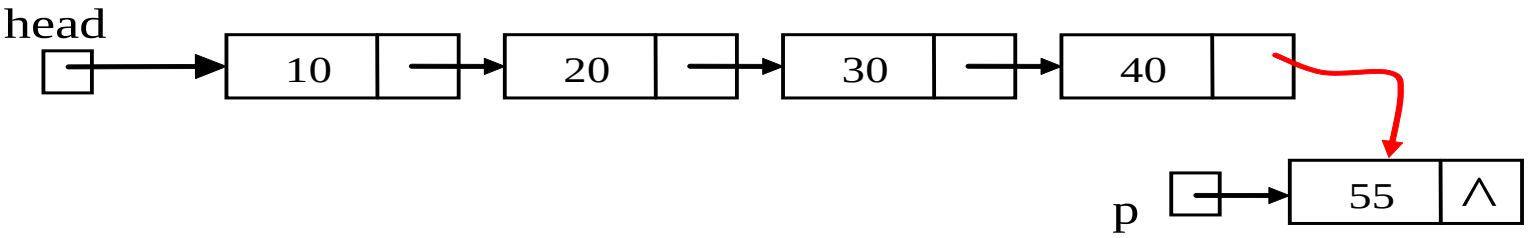


问题分析

- 首先要找到插入位置，有三种情况：

1）小于表头结点，插在表头处；

2）大于表尾结点，插在表尾处；



问题分析

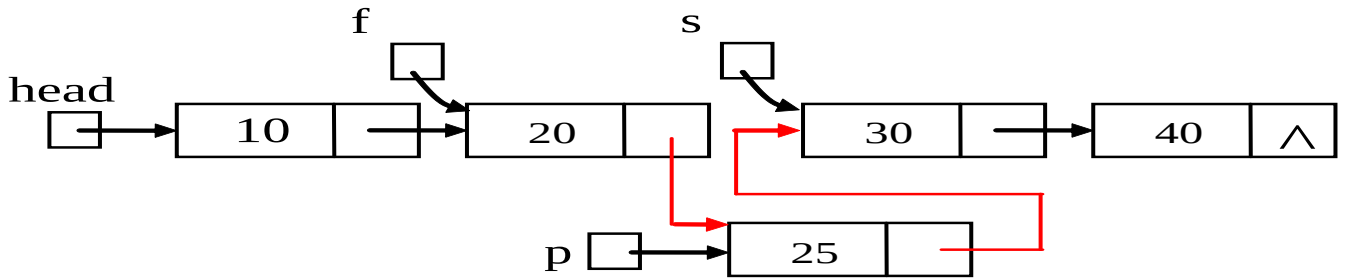
- 首先要找到插入位置，有三种情况：

1）小于表头结点，插在表头处；

2）大于表尾结点，插在表尾处；

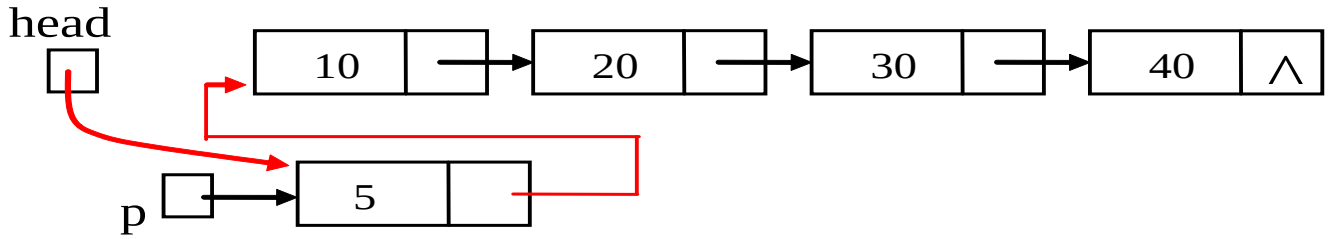
3）第三种情况就是插在表“中间”，某结点p，满足：

$$f->data<p->data\leq s->data$$

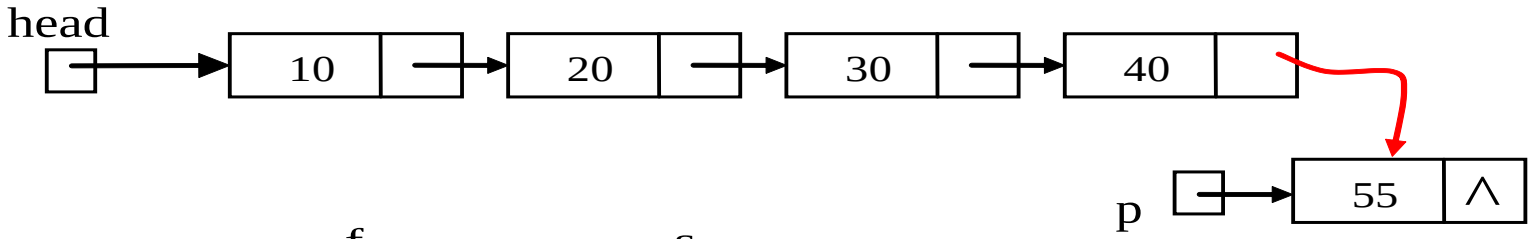


问题分析

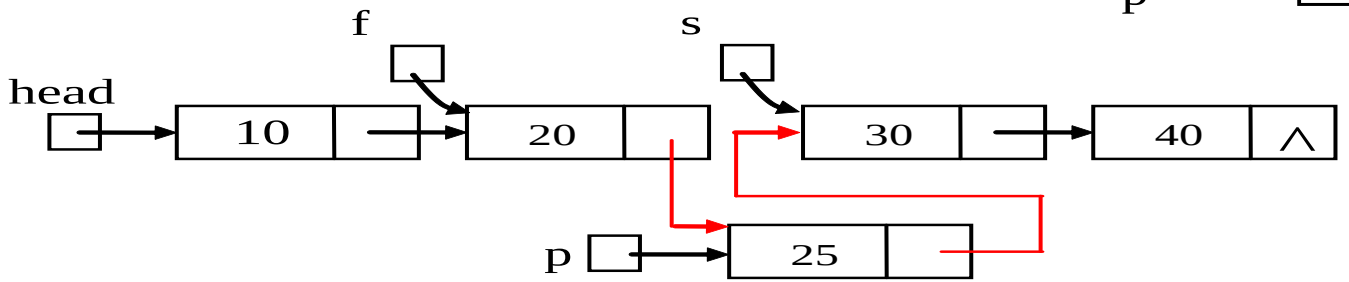
表头插入：



表尾插入：



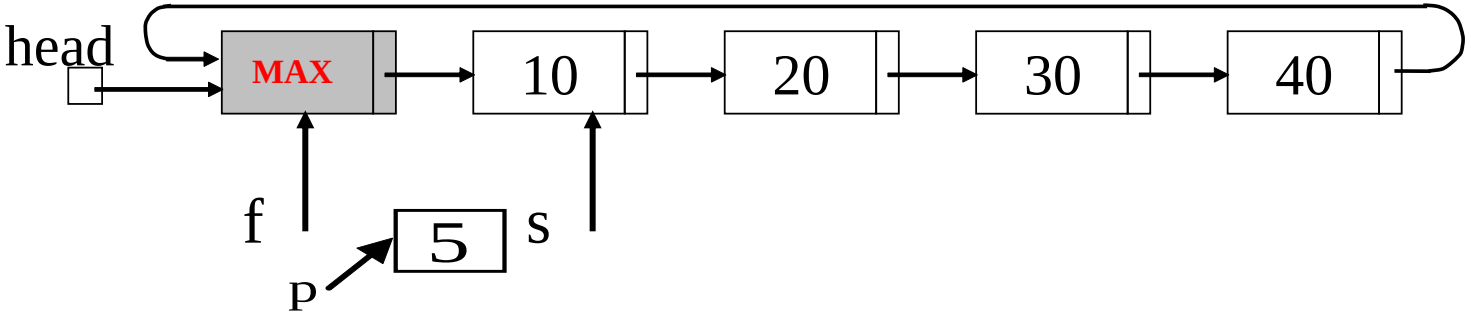
表中插入：



问题分析

加头有序循环链表：

表头插入元素5

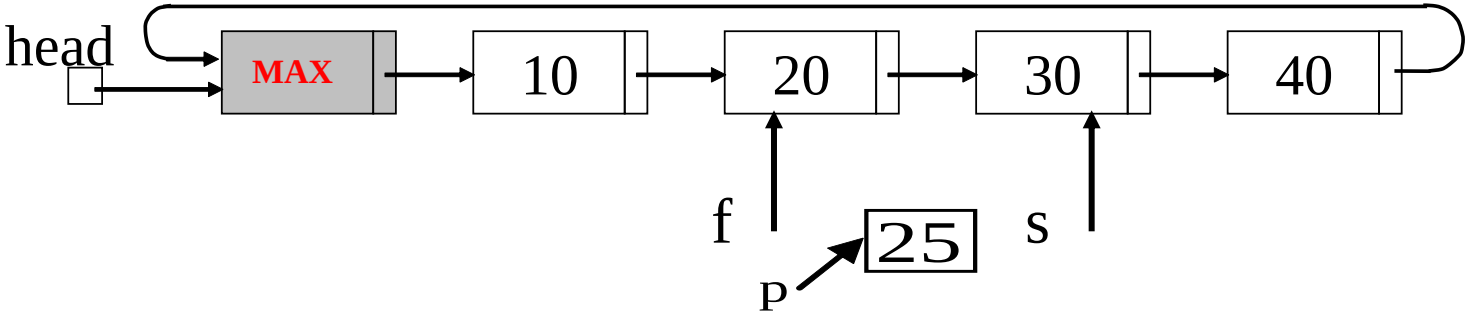



```
6.      p=new snode;      p->data=x;
7.      f=head, s=f->next; ;                    //置搜索指针初值
8.      while(s->data<x)  f=s,s=s->next;        //有序搜索
9.      f->next=p, p->next=s;                   //有序插入
```

问题分析

加头有序循环链表：

表中插入元素25

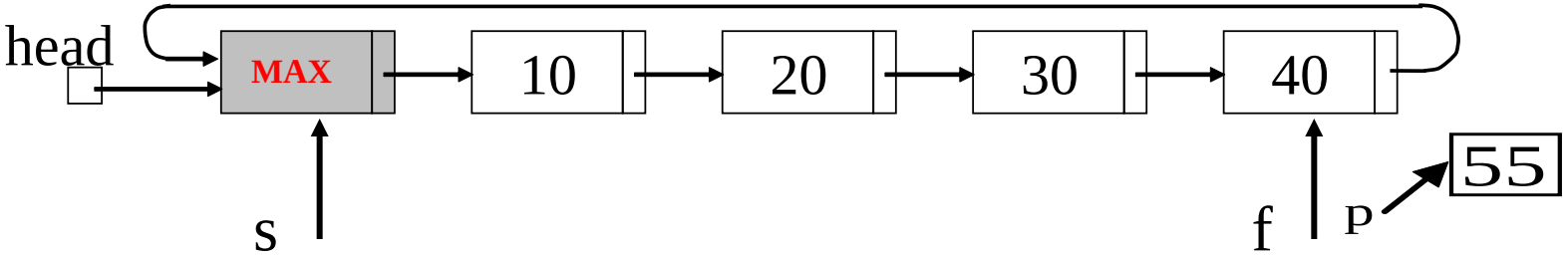



```
6.      p=new snode;      p->data=x;
7.      f=head, s=f->next; ;                    //置搜索指针初值
8.      while(s->data<x)  f=s,s=s->next;        //有序搜索
9.     f->next=p, p->next=s;                    //有序插入
```

问题分析

加头有序循环链表：

表尾插入元素55




```
6.      p=new snode;      p->data=x;
7.      f=head, s=f->next; ;                    //置搜索指针初值
8.      while(s->data<x)  f=s,s=s->next;        //有序搜索
9.     f->next=p, p->next=s;                    //有序插入
```

有序插入法构造加头有序循环链表的算法

```
ptr  creatlinkedBC( )
{  ptr  head, f,s,p;   element_type  x ;
1.   head=new snode;
2.   head->data=MAX; //MAX公共监督元
3.   head->next=head; //构造空链表
4.    scanf("%d", &x);
5.   while (x!=End_elm)
6.      {  p=new snode;        p->data=x;
7.         f=head, s=f->next;                          //置搜索指针初值
8.          while(s->data<x)f=s,s=s->next;    //有序搜索
9.         f->next=p, p->next=s;                       //有序插入
10.         scanf("%d", &x); //读入下一个元素
       }
11.   return(head);
}
```

$O(n^2)$