فایل پنجم درس ساختمان های داده

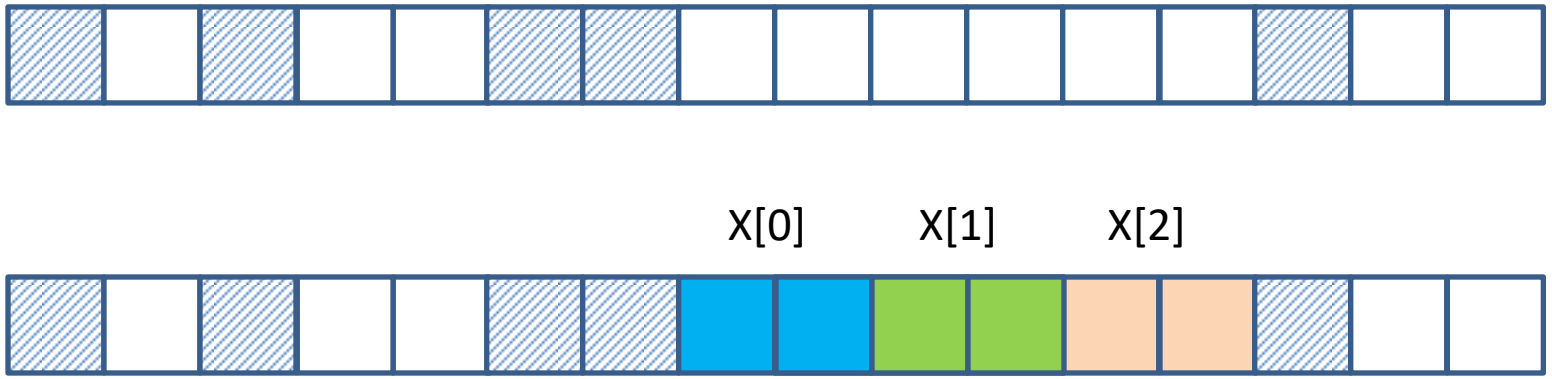
- فایل آماده شده برای جلسه معرفی ساختار لیست پیوندی و کاربردهای آن و صرفا برای کلاس آنلاین در این ترم می باشد.

همواره سلامت و موفق باشید- اسبقی

لیست پیوندی(Linked List)

- برای درک بهتر لیست پیوندی، لازم است ساختار آرایه مجددا بررسی شده و تفاوت های آن با لیست پیوندی مشخص گردد:
- هنگام تعریف یک آرایه داریم:

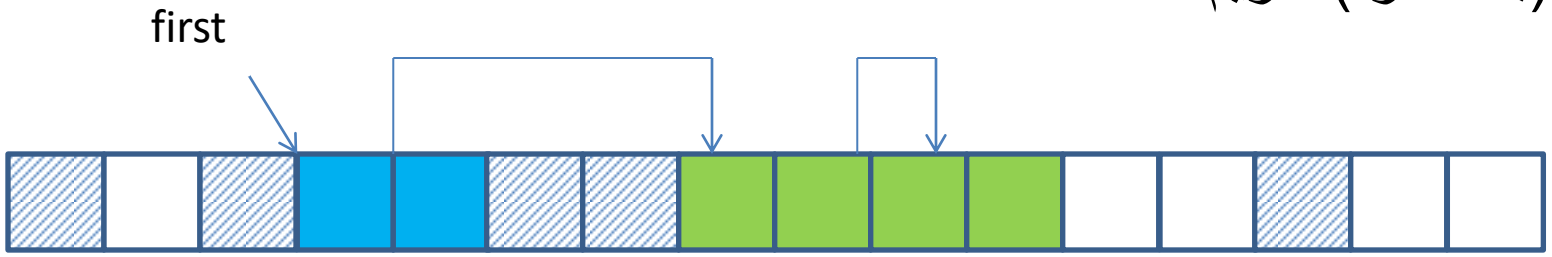
```
int   x[3];
```



آرایه 3 عنصری x بصورت خطی در حافظه ذخیره می شود.

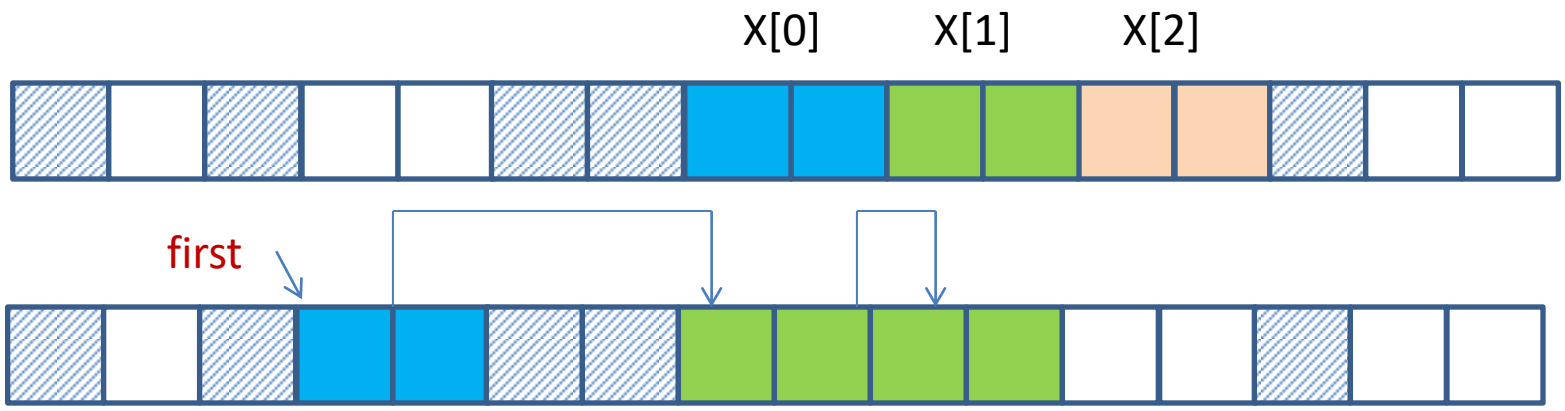
لیست پیوندی

- حال با فرض حافظه با همان شرایط برای ذخیره یک لیست پیوندی 3 عنصری (با 4 گره) داریم:



- همانطور که مشاهده می شود، لیست پیوندی بصورت غیر خطی در حافظه ذخیره می شود. بدین معنی که عناصر یا گره های لیست پیوندی، در حافظه لزوما پشت سر هم نبوده و هر کدام ممکن است در آدرس متفاوت و هرکجای حافظه قرار گرفته باشند. آنچه که در این ساختار متفاوت است، سایز متغیر آن می باشد، بطوریکه با هر عمل درج یا حذف، سایز آن تغییر می کند.

مقایسه دو ساختار آرایه و لیست پیوندی



آرایه	لیست پیوندی
ساختار خطی	ساختار غیر خطی
سایز ثابت	سایز متغیر
ساختار استاتیک(ایستا)	ساختار دینامیک(پویا)
دسترسی تصادفی/مستقیم	دسترسی ترتیبی

لیست پیوندی

لیست پیوندی بعنوان یک ADT:

- تعریف:

مجموعه ای مرتب از عناصر

- عملیات:

-ایجاد یک لیست خالی

-درج عنصر در لیست()- در هر کجای لیست

-حذف عنصر از لیست()-از هر کجای لیست

-پیمایش لیست()

گره لیست پیوندی

یک لیست پیوندی بصورت مجموعه ای از گره هایی ایجاد می شود که با استفاده از اشاره گر به هم مرتبط می گردند.

یک گره لیست پیوندی بصورت زیر تعریف می شود:

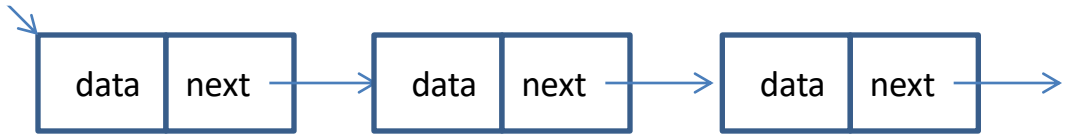
node




```
struct   node{
 int        data;
 node     *next;};
```

نکته: یک اشاره گر از نوعی تعریف می شود که به آن اشاره می کند.

مثالی از یک لیست پیوندی:



تعریف گره لیست پیوندی

تعریف متغیر دینامیک:

```
int     *p;
P = (int  *)malloc(size of (int));
……..
free   p;
```

تعریف گره دینامیک:

```
node   *p;
P = (node  *)malloc(size of (struct node));
…….
free  p;
```

مثالی برای ایجاد گره های لیست پیوندی

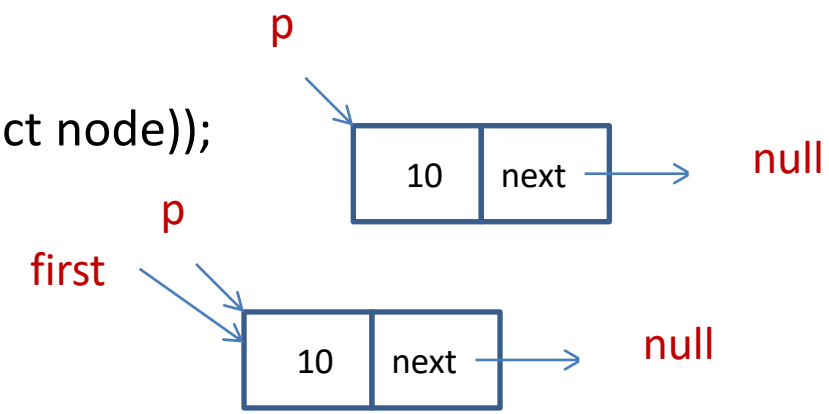
ایجاد لیست خالی:

```
node   *first;
first = (node  *)malloc(size of (struct node));
first = null;
```



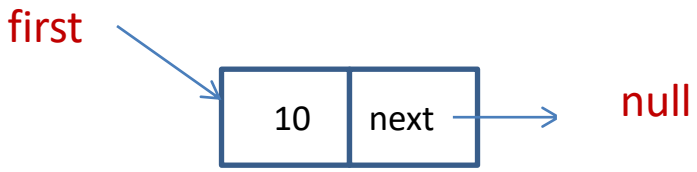
درج یک گره در لیست:

```
node   *p;
p = (node  *)malloc(size of (struct node));
p->data = 10;
p->next = null;
first = p;
```

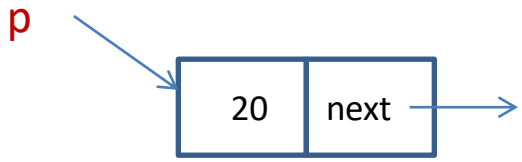


درج عنصر در لیست پیوندی

- بعنوان مثال اگر لازم باشد یک گره جدید در لیست پیوندی حاصل ، اضافه شود، دو حالت داریم:

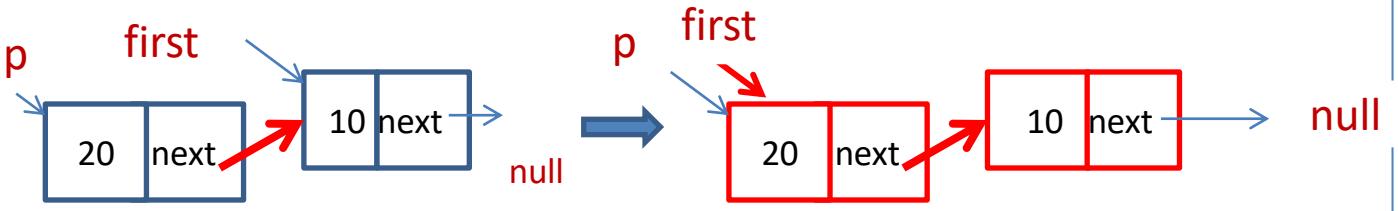



```
node   *p;
p = (node  *)malloc(size of (struct node));
p->data = 20;
```



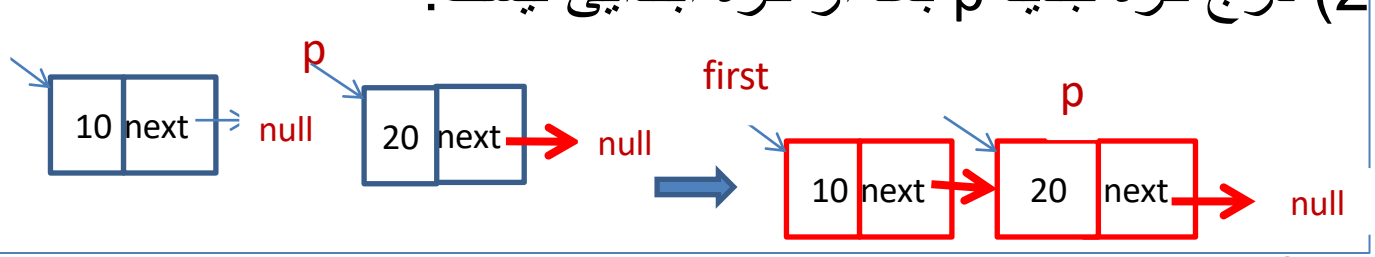
1) درج گره جدید p قبل از گره ابتدایی لیست:

```
p-> next = first;
first = p;
```



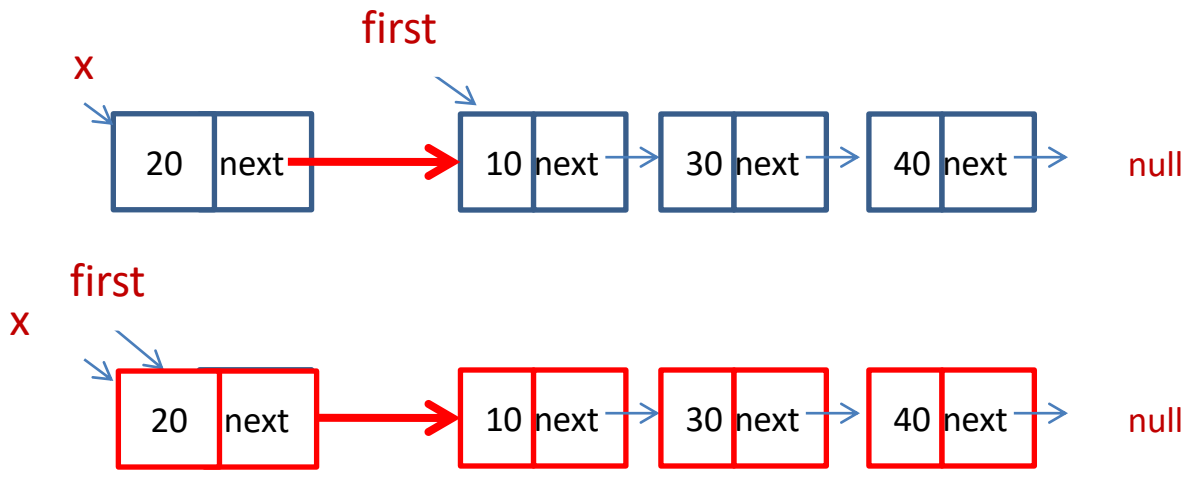
2) درج گره جدید p بعد از گره ابتدایی لیست:

```
p->next = null,
first ->next = p;
```



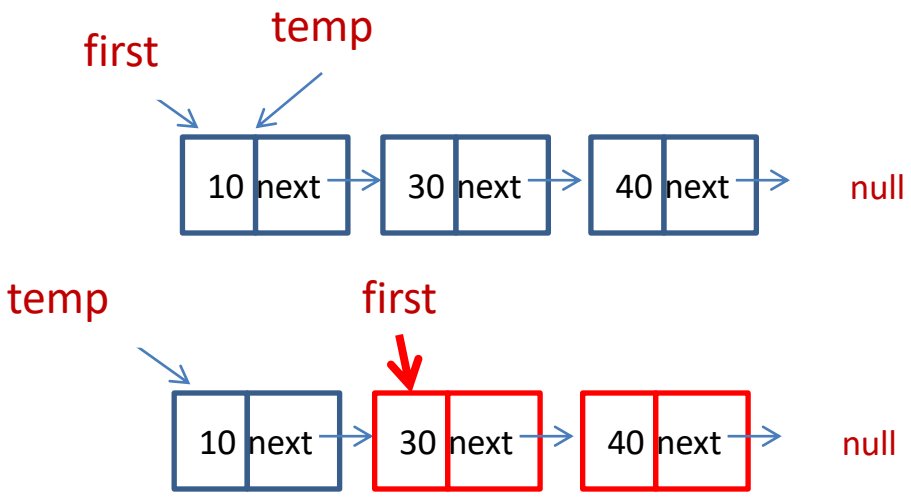
درج عنصر در ابتدای لیست پیوندی

```
void   insert_node(node  *first , node  *x)
{ x->next = first;
  first  = x;
}
```



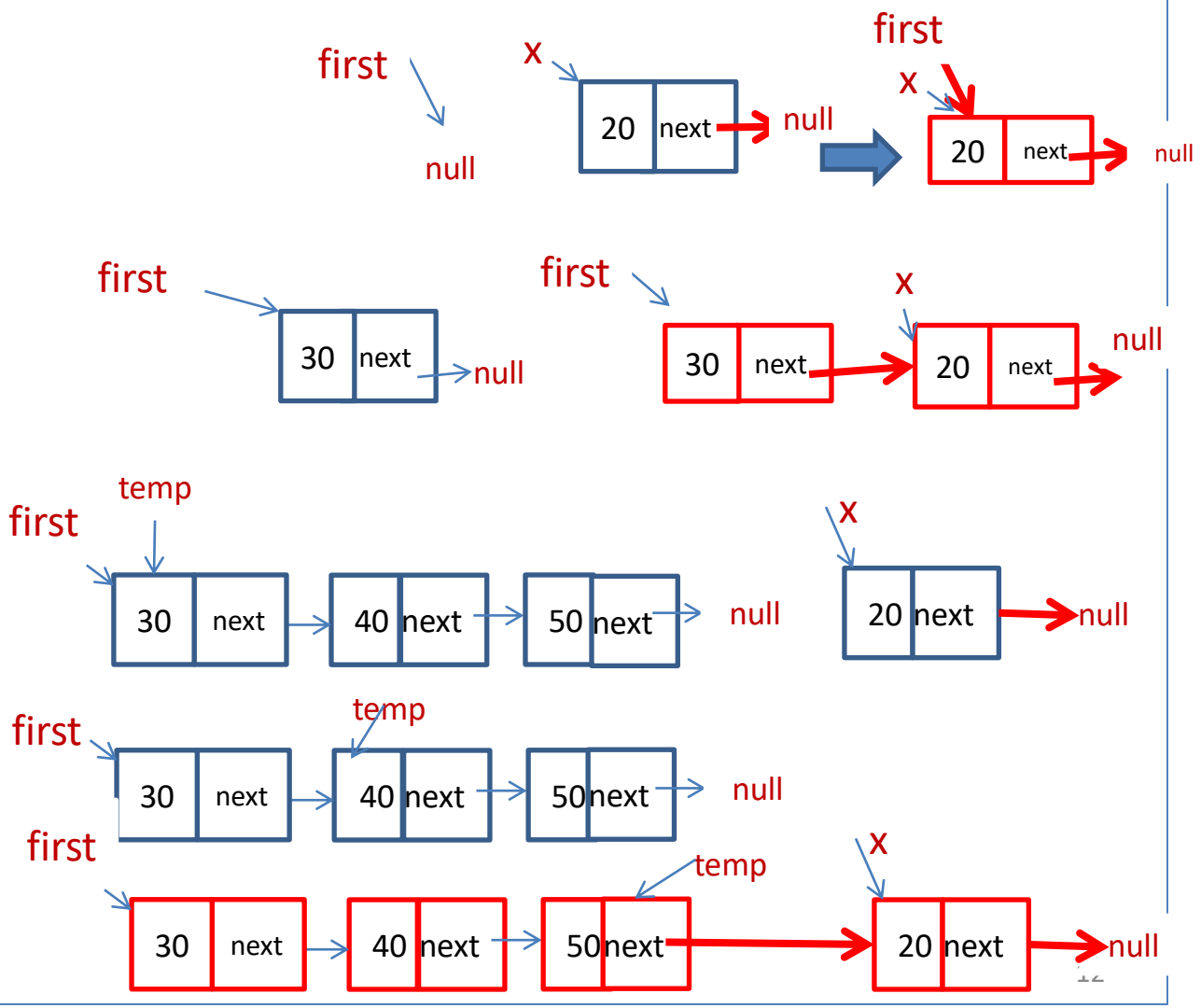
حذف عنصر از ابتدای لیست پیوندی

```
void  delete_node(node  *first){
if(first == null)
   exit(1);
node   *temp = first;
first = first ->next;
free(temp);}
```



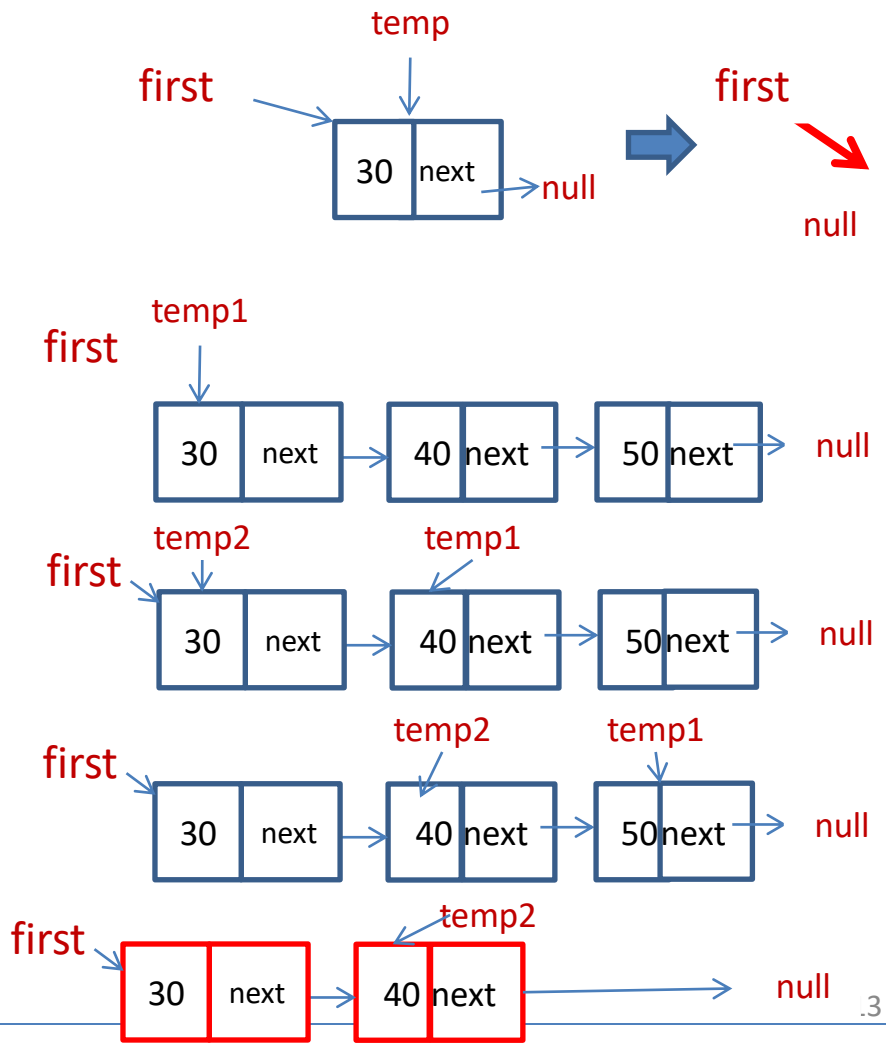
درج عنصر در انتهای لیست پیوندی

```
void  inset_node(node  *first,node  *x){
x->next = null;
if( first == null)
    first = x;
elseif(first->next ==null)
   first->next = x;
else{ node  *temp = first;
   while(temp->next!=null)
        temp = temp->next;
  temp->next = x;}
}
```



حذف عنصر از انتهای لیست پیوندی

```
void  delete_node(node  *first){
if( first == null)
    exit(1);
elseif(first->next ==null)
 {node  *temp =first;
   first  = null;
   free(temp);}
else{ node  *temp 2,*temp1 = first;
   while(temp1->next!=null){
         temp2 = temp1;
         temp1 = temp1->next;}
  temp2 ->next = null;
  free(temp1);}
}
```

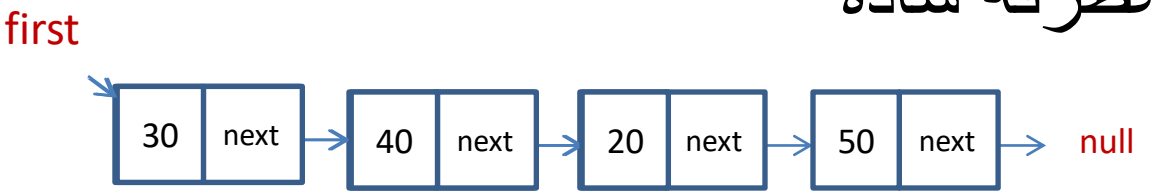


انواع لیست پیوندی

- لیست پیوندی یکطرفه ساده
- لیست پیوندی یکطرفه حلقوی
- لیست پیوندی با گره رأس
- لیست پیوندی دوطرفه ساده
- لیست پیوندی دوطرفه حلقوی

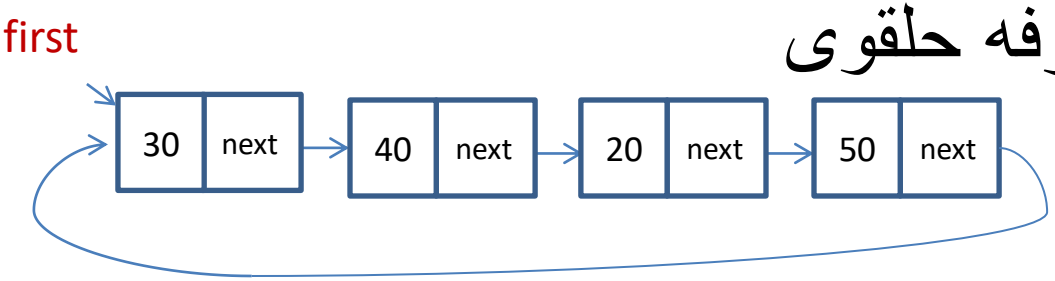
لیست پیوندی یک طرفه

- لیست پیوندی یکطرفه ساده



if(temp->next ==null)

- لیست پیوندی یکطرفه حلقوی

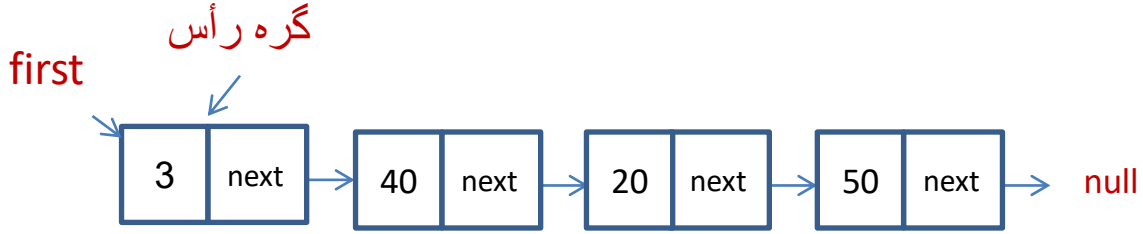


if(temp->next ==first)

لیست پیوندی با گره رأس

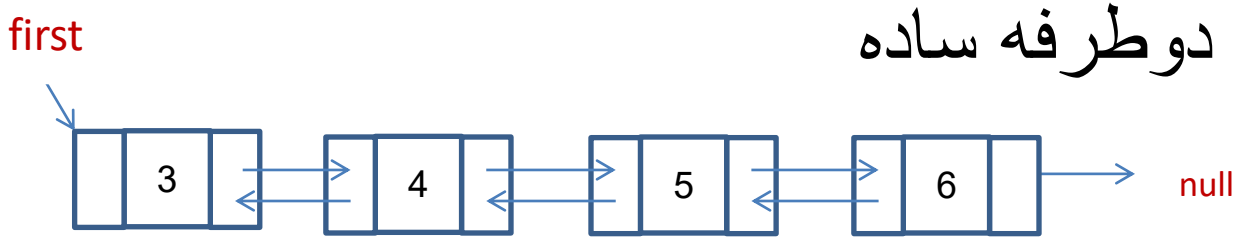
- در لیست پیوندی با گره رأس، یگ گره برای نگهداری تعداد گره های لیست در نظر گرفته می شود.
- مزیت: همواره تعداد گره های لیست به راحتی و بدون پیمایش آن، در دسترس است.
- عیب:

هر عمل درج یا حذف از لیست، نیازمند بروزرسانی مقدار گره رأس می باشد.

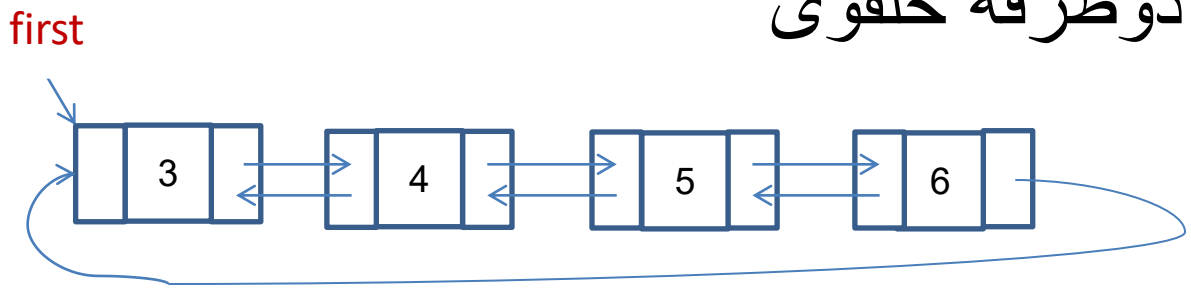


لیست پیوندی دو طرفه(لیست دو پیوندی)

- لیست پیوندی دوطرفه ساده



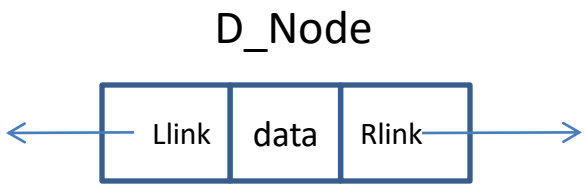
- لیست پیوندی دوطرفه حلقوی



گره لیست دو پیوندی

یک گره لیست دو طرفه دارای دو اشاره گر به گره های بعدی و قبلی می باشد.

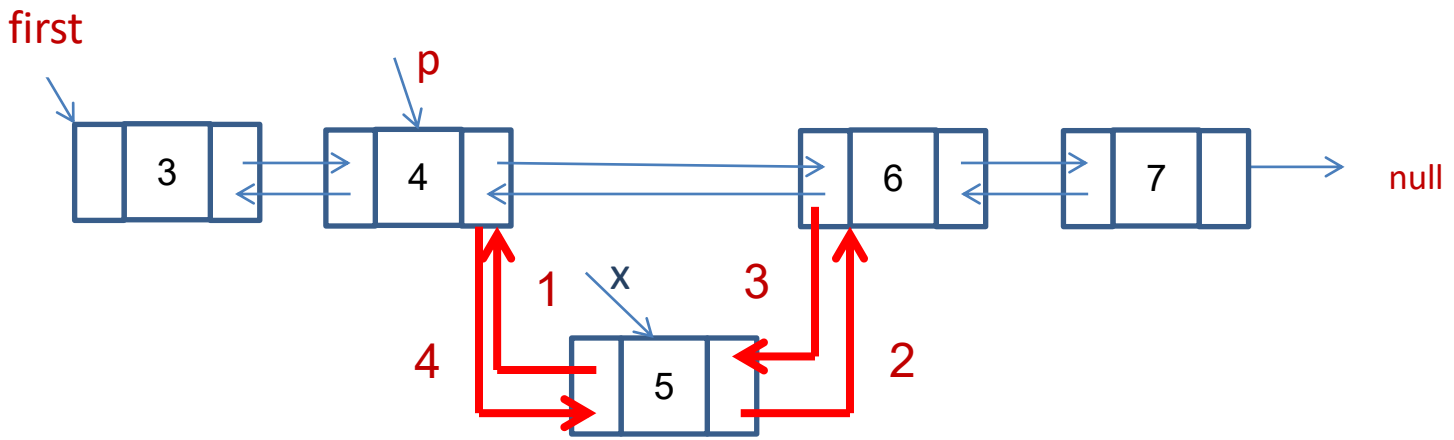
```
struct  D_Node{
int   data;
D_Node   *Rlink;
D_Node   *Llink;};
```



کاربرد:

استفاده در ساختارهایی که نیاز است به عنصر بعدی و قبلی دسترسی داشته باشیم.

درج عنصر در لیست دو طرفه



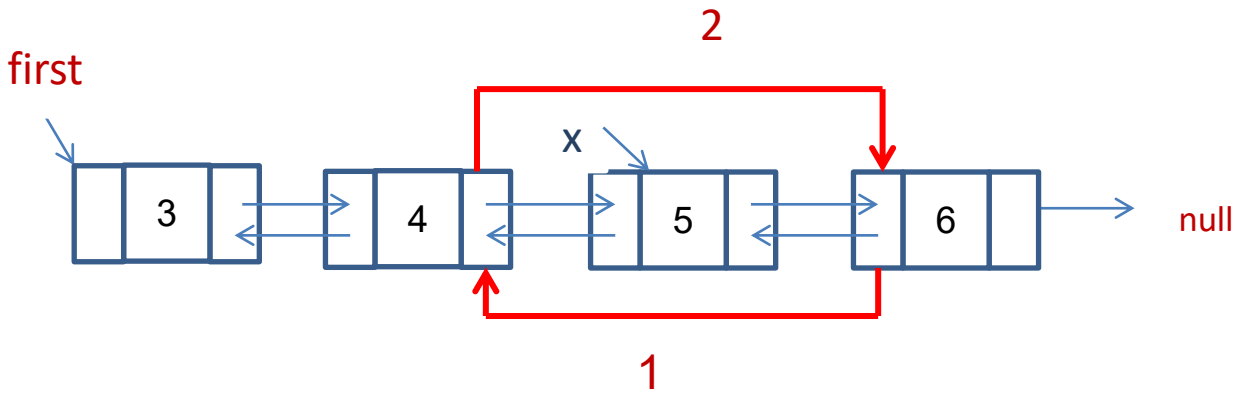
1)x->Llink = p;

2)x->Rlink = p->Rlink;

3)p->Rlink ->Llink = x;

4)p->Rlink = x;

حذف عنصر از لیست دو طرفه



1) x->Rlink->Llink = x->Llink;

2) x->Llink->Rlink = x->Rlink;

تمرین

- الگوریتم درج یک یک عنصر در محل پنجم یک لیست پیوندی یکطرفه ساده را بنویسید.
- ساختار پشته و صف را با استفاده از لیست پیوندی پیاده سازی نموده و اعمال درج و حذف عنصر از آنها را بنویسید. مزایای پیاده سازی صف و پشته با لیست پیوندی چیست؟