ساختمان داده ها و الگوریتم ها

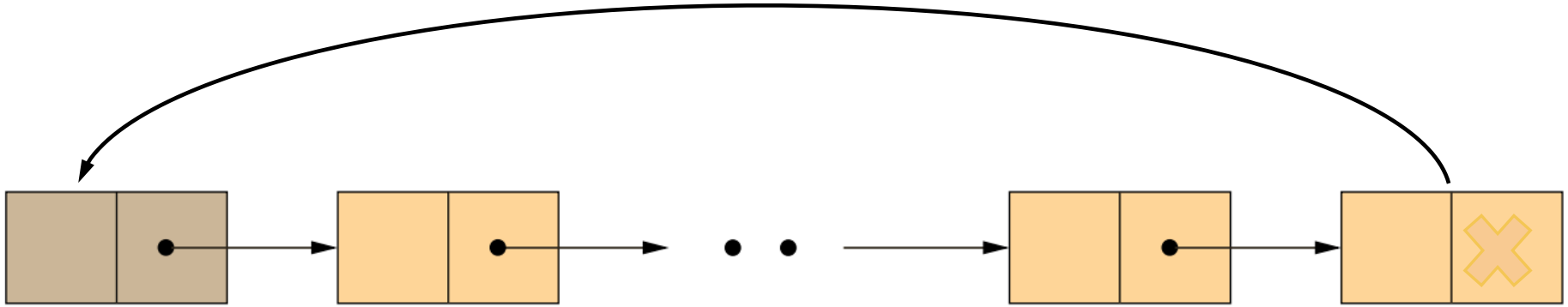
انواع لیست های پیوندی
پیاده سازی صف و پشته

محمدحسین اولیائی

رئوس مطالب

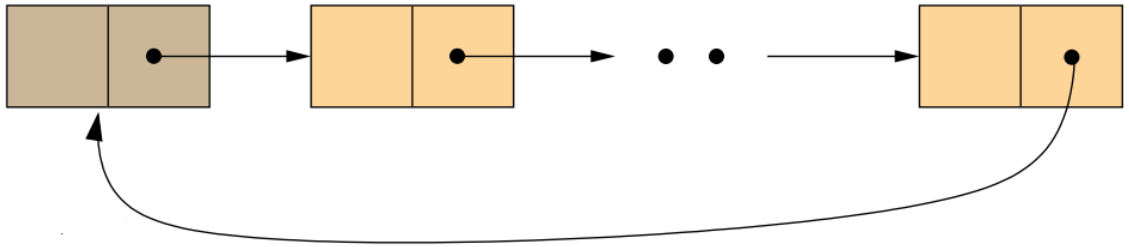
- لیست پیوندی حلقوی
- لیست پیوندی دوطرفه
- پیاده سازی پشته
- پیاده سازی صف

لیست پیوندی حلقوی



لیست پیوندی حلقوی

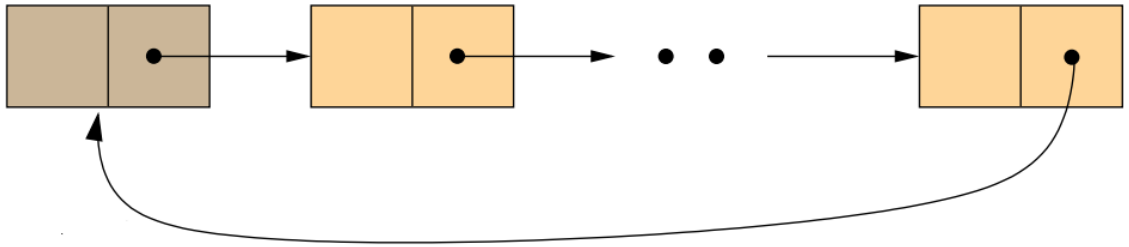
- پیمایش لیست حلقوی



لیست پیوندی حلقوی

- پیمایش لیست حلقوی

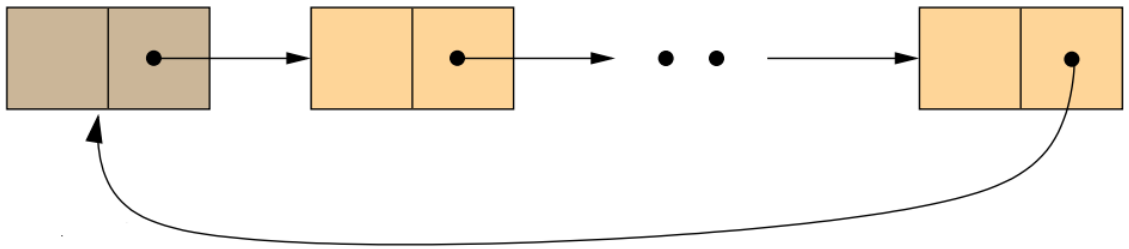
```
void show()
{
    if(!start) return;
    node *temp=start;
   do{
        cout<<temp->data<<“\t”;
        temp=temp->link;
    }while(?);
}
```



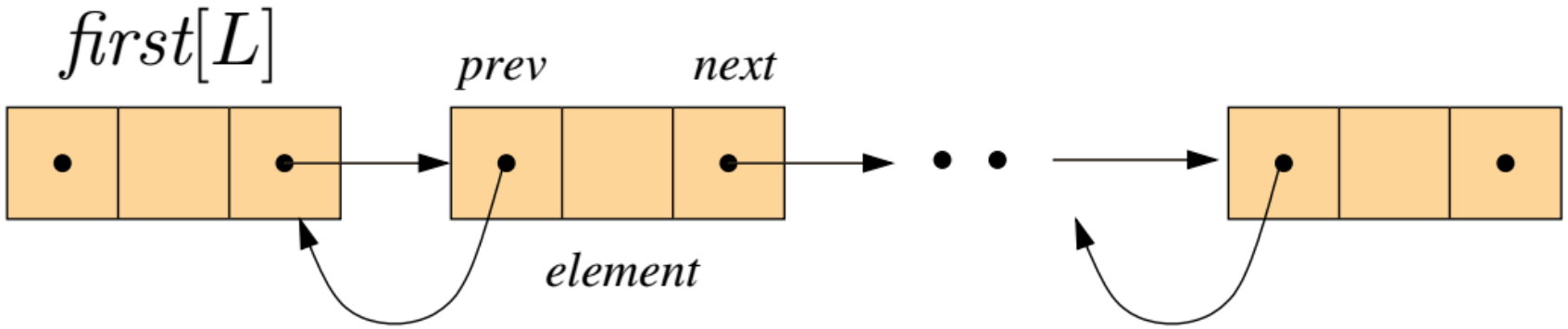
لیست پیوندی حلقوی

- پیمایش لیست حلقوی

```
void show()
{
    if(!start) return;
    node *temp=start;
   do{
        cout<<temp->data<<“\t”;
        temp=temp->link;
    }while(temp->link!=start);
}
```



لیست پیوندی دوطرفه



```
class BiLinkList
{       private:
                struct node* start;
        public:
                BiLinkList();
                int IsEmpty();
                void insert_First(int);
                void insert_Last(int);
                void insert_after(struct node *,int d);
                struct node* search(int);
                void remove_First();
                void remove_Last();
                void remove_after(struct node *);
                void remove(int);
                int ListSize();
                ~BiLinkList();
                void show();
};
```

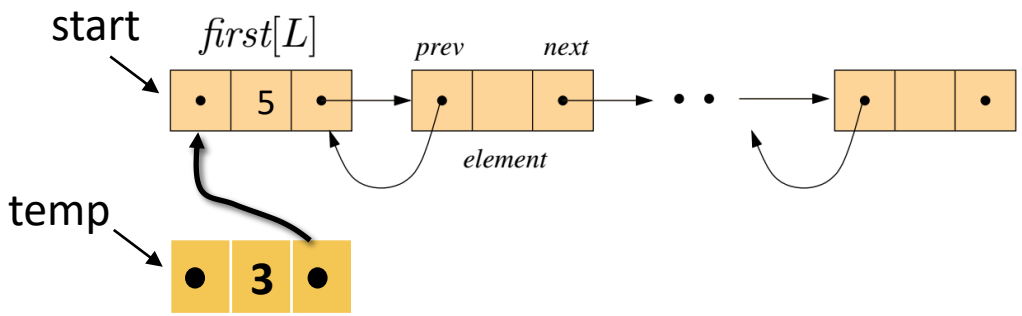
```
struct node
{
        int data;
        struct node* next;
        struct node* prev;
};
```

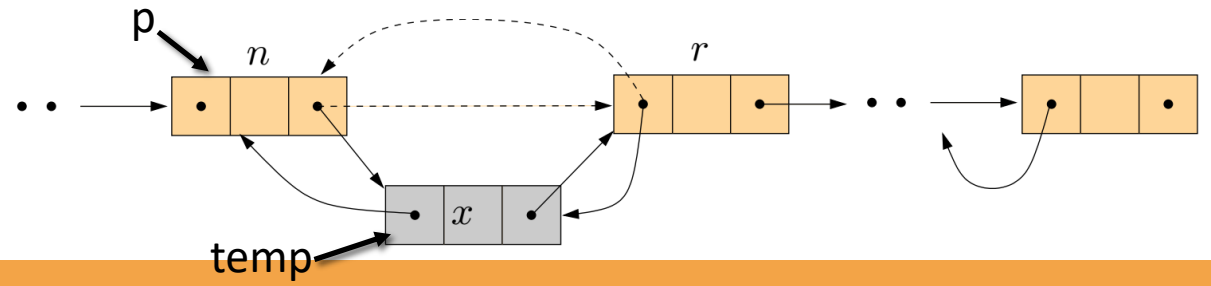
لیست پیوندی دوطرفه

- درج در لیست پیوندی دو طرفه مرتب

```
void insert(int x)
{  node * p;
   node *temp=new node;
   if(!temp) return;
   temp->data=x;
   temp->next=NULL;
   temp->prev=NULL;
   if(!start || start->data>x)
    { temp->next=start;
      strat->prev=temp;
      start=temp;
      return;
    }
   for(p=start;p->next&&p->next->data<x; p=p->next);
   temp->prev=p;
   temp->next=p->next;
   if(p->next)
     p->next->prev=temp;
   p->next=temp;
}
```

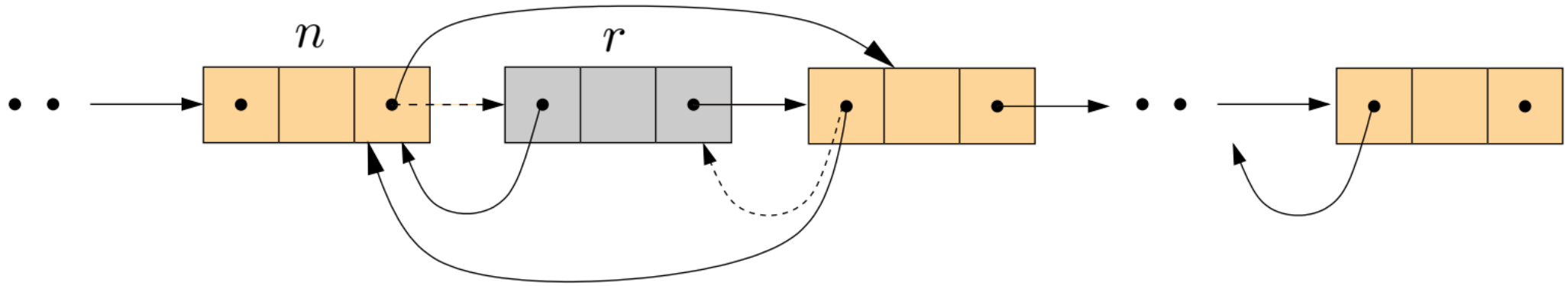
?





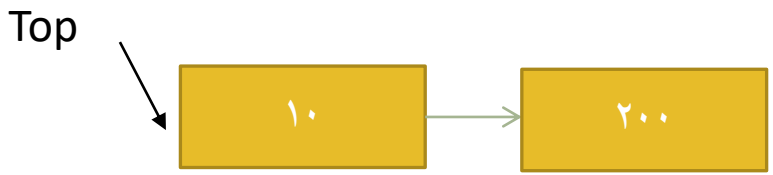
لیست پیوندی دوطرفه

- حذف در لیست پیوندی دو طرفه مرتب



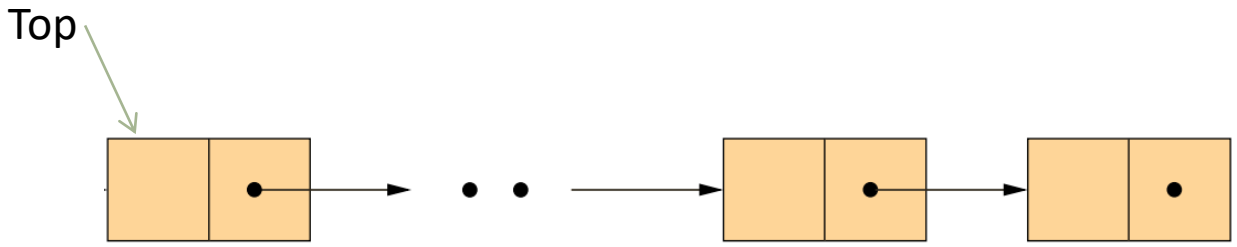
پیاده سازی پشته با لیست پیوندی

- شرط خالی بودن
- شرط پر بودن
- Push
- Pop



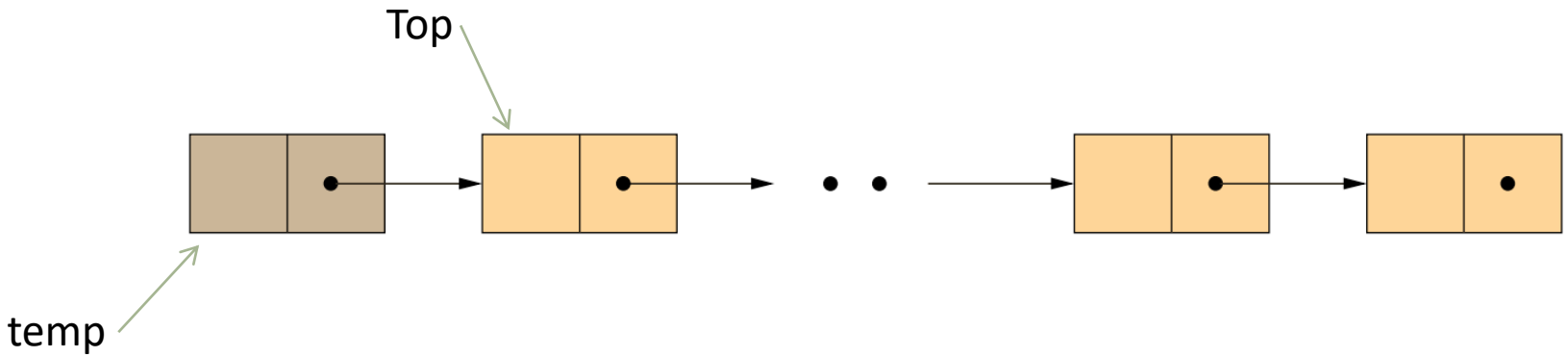
پیاده سازی پشته با لیست پیوندی

- شرط خالی بودن
- شرط پر بودن
- **Push**
- Pop



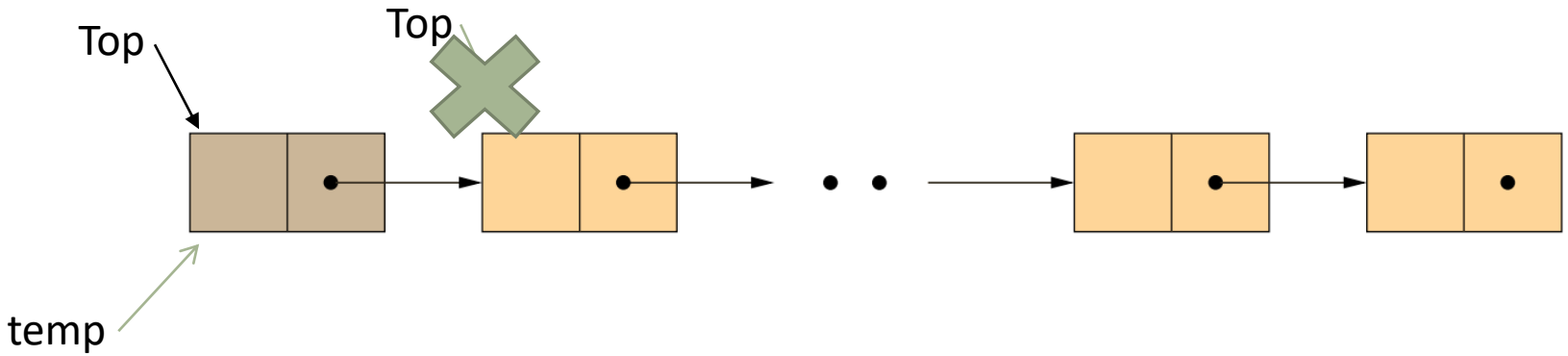
پیاده سازی پشته با لیست پیوندی

- شرط خالی بودن
- شرط پر بودن
- **Push**
- Pop



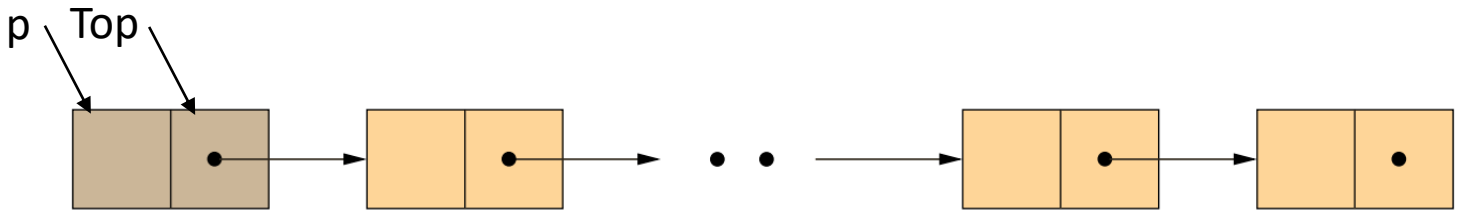
پیاده سازی پشته با لیست پیوندی

- شرط خالی بودن
- شرط پر بودن
- **Push**
- Pop



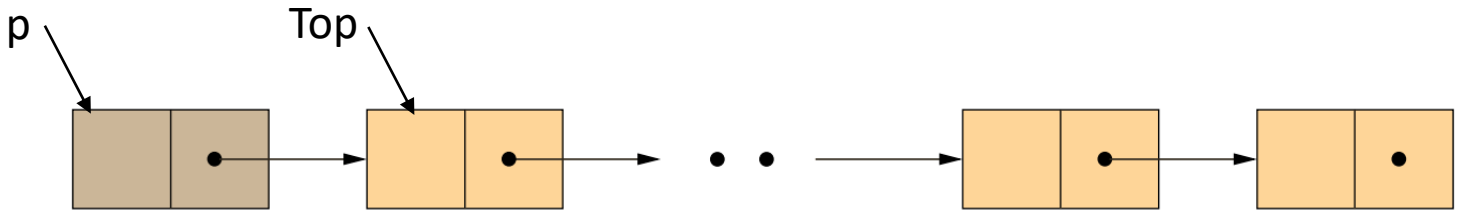
پیاده سازی پشته با لیست پیوندی

- شرط خالی بودن
- شرط پر بودن
- Push
- **Pop**



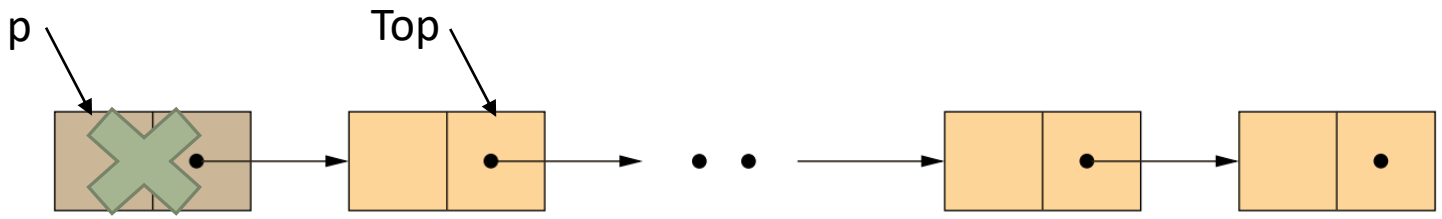
پیاده سازی پشته با لیست پیوندی

- شرط خالی بودن
- شرط پر بودن
- Push
- **Pop**



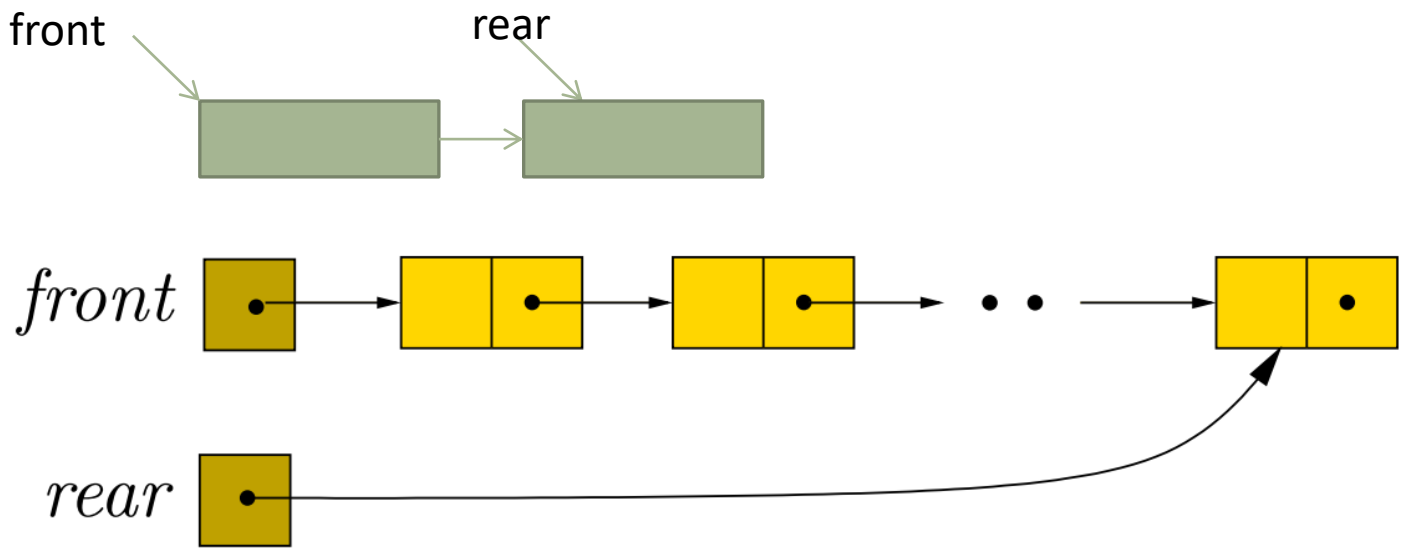
پیاده سازی پشته با لیست پیوندی

- شرط خالی بودن
- شرط پر بودن
- Push
- **Pop**



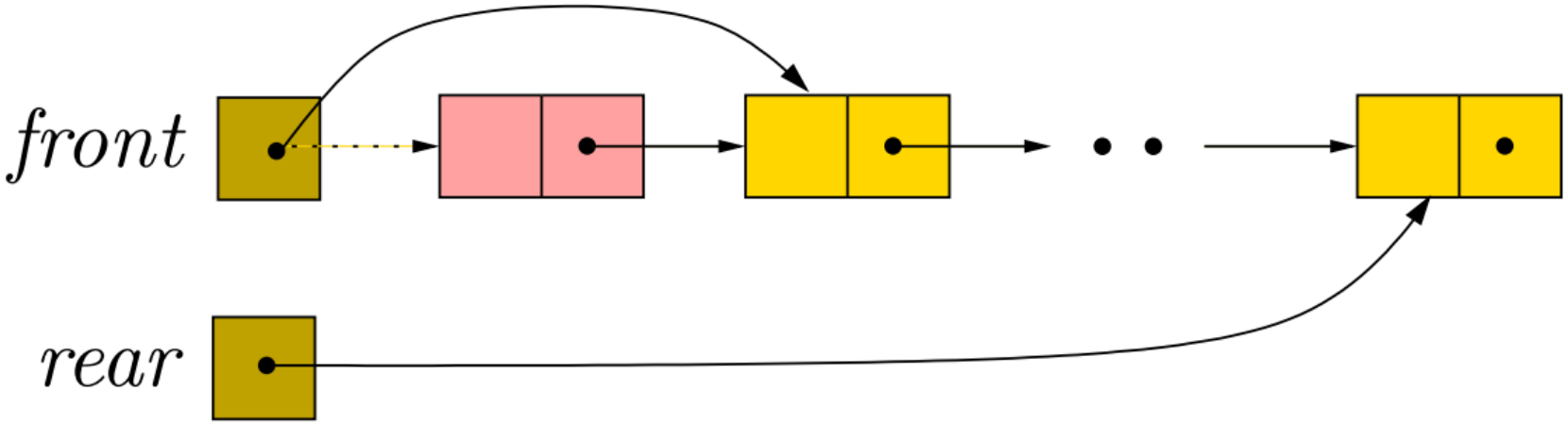
پیاده سازی صف با لیست پیوندی

- شرط خالی بودن
- شرط پر بودن



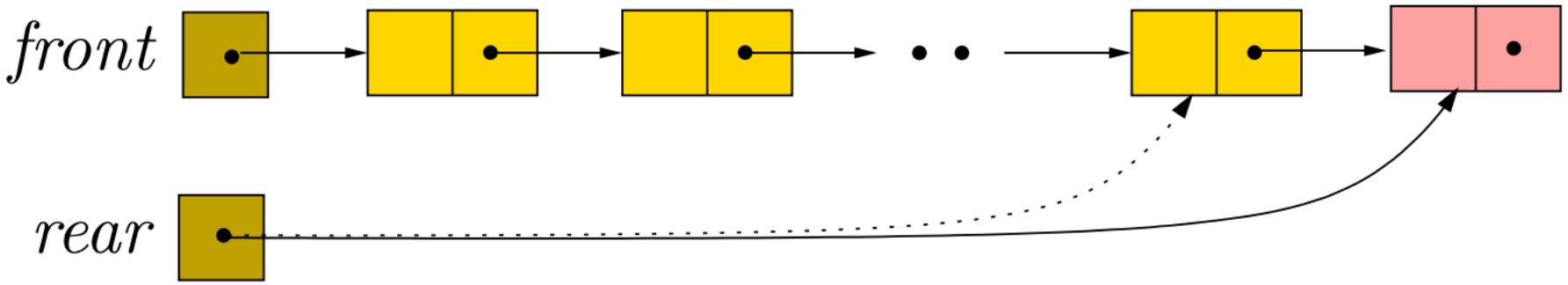
پیاده سازی صف با لیست پیوندی

حذف از صف



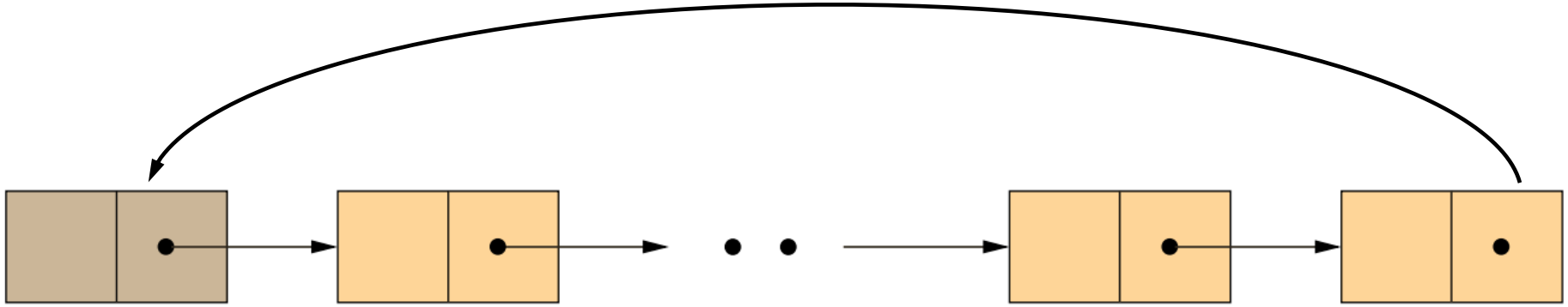
پیاده سازی صف با لیست پیوندی

درج در صف

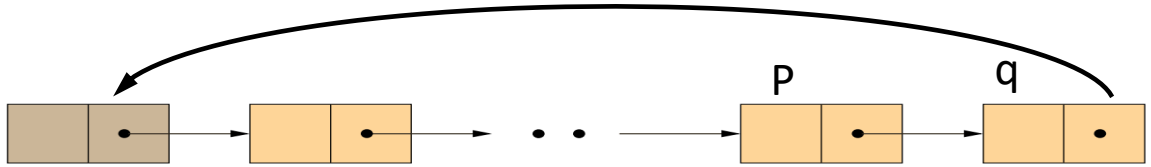


مثال

تابعی بنویسید که آدرس شروع یک لیست پیوندی حلقوی را دریافت کرده و آخرین نود آن را حذف نماید



```
Void removeLast(node *start)
{
if (!start) return;
Node *p,*q;
if (start->link==start) delete(start);
P=start; q=p->link;
While (q->link!=start)
{ p=p->link;
q=q->link;
}
P->link=start;
Delete(q);}
```



تمرینات

۱) کلاسی به نام stackLink پیاده سازی کنید که از ساختار لیست پیوندی ساده جهت نگهداری مقادیر استفاده می نماید.

۲) تابعی بنویسید که از یک لیست پیوندی دوطرفه، گره با مقدار r را حذف نماید.

پروژه۱

برنامه ای بنویسید که فایلی متنی را دریافت کرده و کلمات آنرا به ترتیب در یک لیست پیوندی دوطرفه درج نماید.

هر گره شامل کلمه و تعداد دفعات تکرار آن می باشد. این برنامه می بایست شامل قابلیت های زیر باشد:

- نمایش اطلاعات به ترتیب حروف
- نمایش اطلاعات به ترتیب تکرار کلمات
- جستجوی کلمه
- حذف یک کلمه خاص