

Индуктивни СД. Линейни едносвързани списъци

Калин Георгиев

16 април 2020 г.

Индуктивни СД

Необходимост от “влагане” на еднотипни обекти

```
struct Employee
{
    char name[100];
    double salary;
    ///???
    Employee boss;
};
```

Указател към обект от същия тип

```
struct Employee
{
    char name[100];
    double salary;
    Employee *boss;
};
```

Указател към обект от същия тип

```

struct Employee
{
    Employee (char *n, double s)
    {
        strcpy (name,n);
        salary = s;
        boss = nullptr;
    }
    char name[100];
    double salary;
    Employee *boss;
};

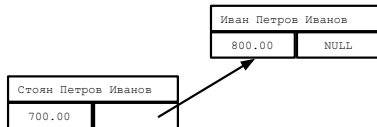
int main ()
{
    Employee
        stoyan ("Stoyan_Petrov_Ivanov", 700),
        ivan ("Ivan_Petrov_Ivanov", 800);
    return 0;
}

```

Стоян Петров Иванов	
700.00	NULL

Иван Петров Иванов	
800.00	NULL

Указател към обект от същия тип

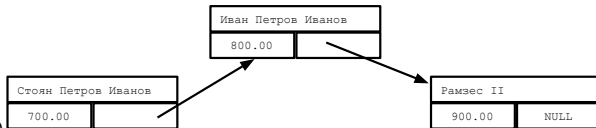


```
int main ()
{
    Employee
        stoyan ("Stoyan_Petrov_Ivanov", 700),
        ivan ("Ivan_Petrov_Ivanov", 800);

    stoyan.boss = &ivan;

    return 0;
}
```

Указател към обект от същия тип



```

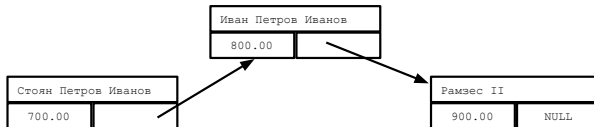
int main ()
{
    Employee
        stoyan ("Stoyan_Petrov_Ivanov", 700),
        ivan ("Ivan_Petrov_Ivanov", 800),
        bigboss ("Big_Boss", 900);

    stoyan.boss = &ivan;
    ivan.boss = &bigboss;
    //stoyan.boss->boss = &bigboss;

    cout << stoyan.boss->name;
    cout << stoyan.boss->boss->name;

    return 0;
}
  
```

“Обхождане”



```

Employee *findSuperBoss (Employee *e)
{
    while (e->boss != nullptr)
        e = e->boss;
    return e;
}

Employee *findSuperBossRec (Employee *e)
{
    if (e->boss == nullptr)
        return e;
    return findSuperBossRec (e->boss);
}

//...
cout << findSuperBoss (&stoyan)->name;
//...

```


Линейни едносвързани списъци

Т. нар. “двойна кутия”

```
struct box
{
    int data;
    box *next;
    box (int d, box *n):
        data(d), next (n) {}
};
```

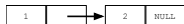
• Един елемент

```
box *first = new box (1,nullptr);
```



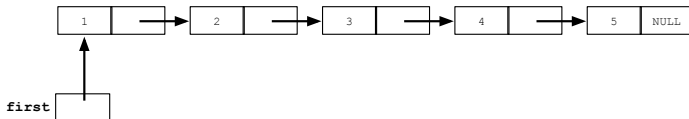
• Два свързани елемента

```
box *first = new box (1,new box (2, nullptr));
```



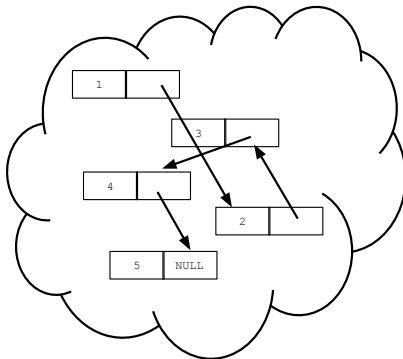
“Плосък” изглед

```
box *first = new box (1,  
    new box (2,  
        new box (3,  
            new box (4,  
                new box (5,nullptr))));
```



“Реален” изглед

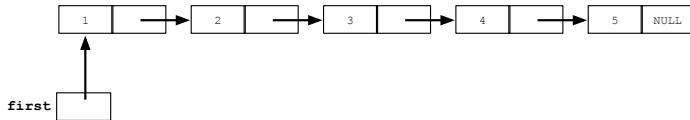
```
box *first = new box (1,  
    new box (2,  
        new box (3,  
            new box (4,  
                new box (5,nullptr))));
```



“Вмъкване” на елемент в началото (push)

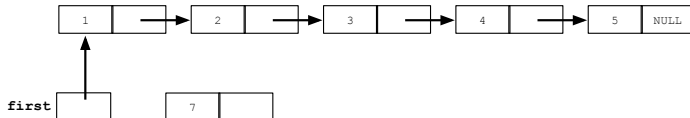
“Вмъкване” на елемент в началото

```
first = ...
```



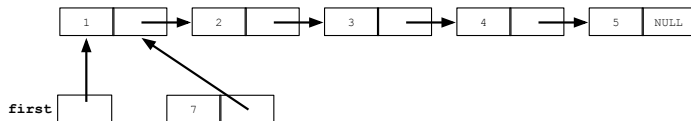
“Вмъкване” на елемент в началото

```
box *newbox = new box (7,nullptr);
```



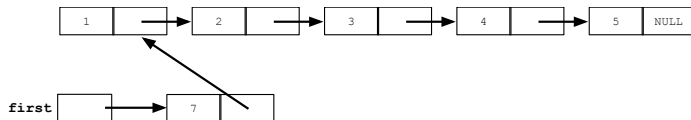
“Вмъкване” на елемент в началото

```
box *newbox = new box (7,nullptr);  
newbox->next = first;
```



“Вмъкване” на елемент в началото

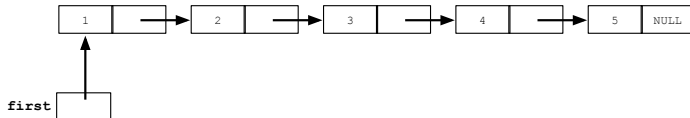
```
box *newbox = new box (7,nullptr);  
newbox->next = first;  
first = newbox;
```



Обхождане

Обхождане на всички елементи

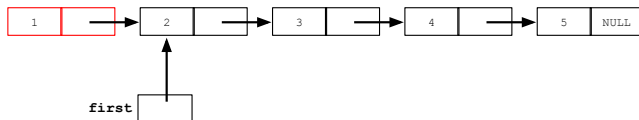
```
cout << first->data;
```



Обхождане на всички елементи

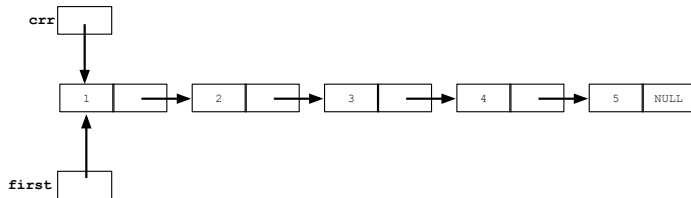
- Трябва ни помощен указател!

```
first = first->next;  
cout << first->data;
```



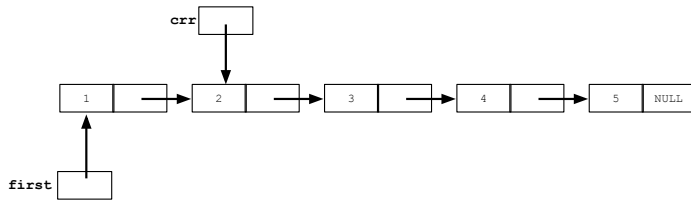
Обхождане на всички елементи

```
box *crr = first;  
cout << crr->data;
```



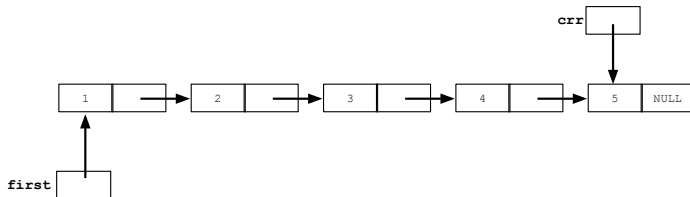
Обхождане на всички елементи

```
box *crr = first;  
crr = crr->next;  
cout << crr->data;
```



Обхождане на всички елементи

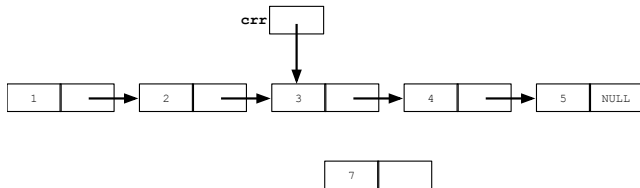
```
box *crr = first;  
while (crr != nullptr)  
{  
    cout << crr->data;  
    crr = crr->next;  
}
```



Вмъкване във вътрешността

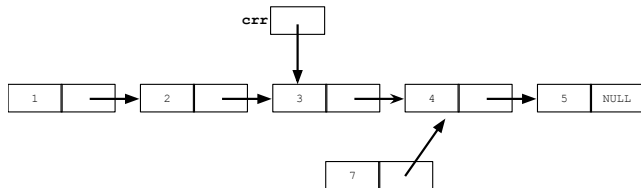
Вмъкване

```
box *newbox = new box (7,nullptr);
```



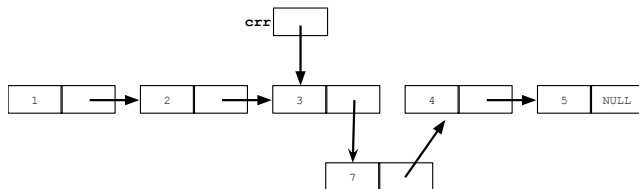
Вмъкване

```
box *newbox = new box (7,nullptr);  
newbox->next = crr->next;
```



Вмъкване

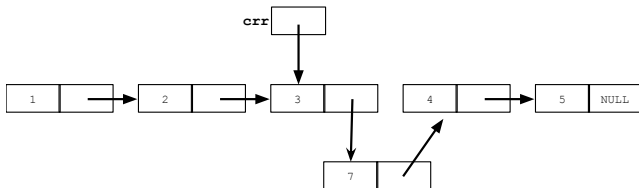
```
box *newbox = new box (7,nullptr);  
newbox->next = crr->next;  
crr->next = newbox;
```



Вмъкване

```
box *crr = first;  
while (3 != crr->data)  
    crr = crr->next;
```

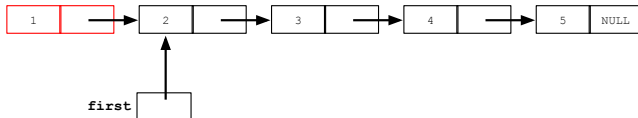
```
box *newbox = new box (7,nullptr);  
newbox->next = crr->next;  
crr->next = newbox;
```



Изтриване на елемент от началото (pop)

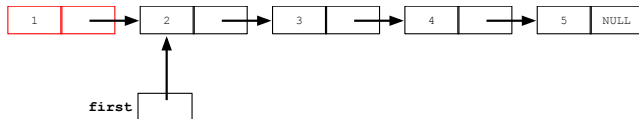
Pop

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



Pop

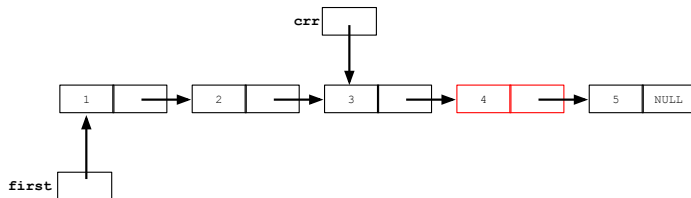
```
box *save = first;  
first=first->next;  
delete save;
```



Изтриване на елемент от позиция

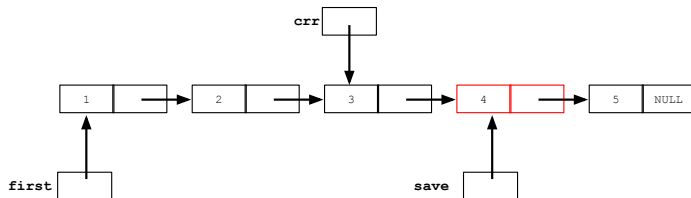
Изтриване

```
crr=...
```



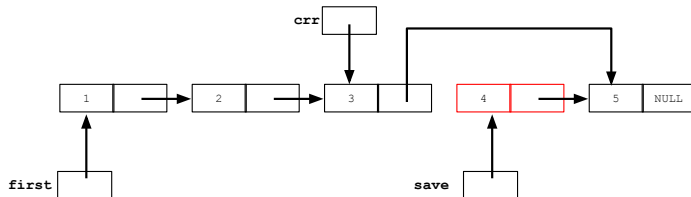
Изтриване

```
box *save = crr->next;
```



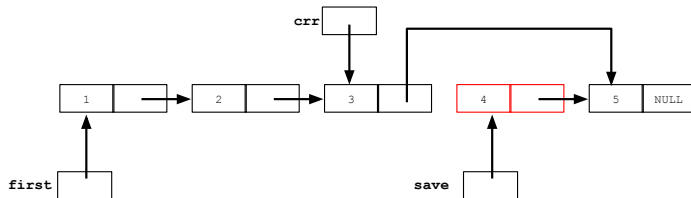
Изтриване

```
box *save = crr->next;  
crr->next = crr->next->next;
```



Изтриване

```
box *save = crr->next;  
crr->next = crr->next->next;  
delete save;
```



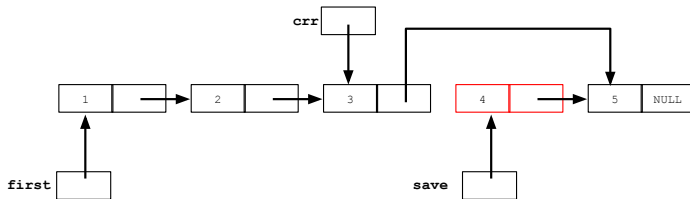
Изтриване на ел. 4

```

box *crr = first;
while (crr->next->data != 4)
    crr = crr->next;

box *save = crr->next;
crr->next = crr->next->next;
delete save;

```



Благодаря ви за вниманието!