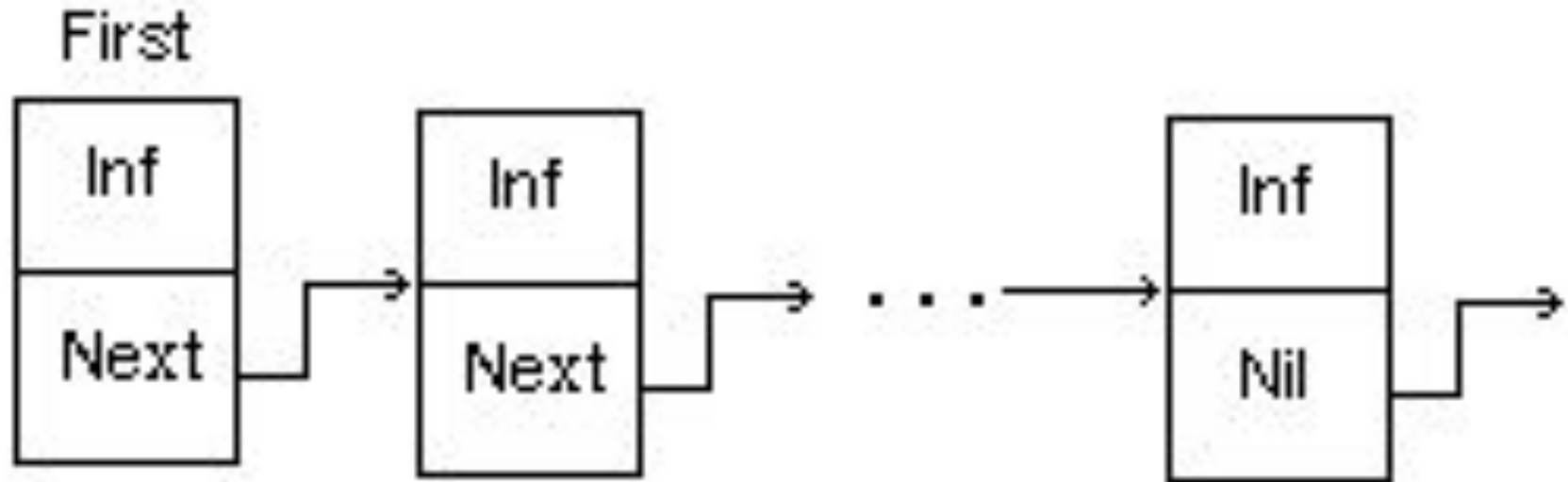


СПИСКИ

Список располагается в динамически распределяемой памяти, в статической памяти хранится лишь указатель на заглавное звено. Структура, в отличие от массива, является динамической: звенья создаются и удаляются по мере необходимости, в процессе выполнения программы.



Где *Inf* — информационная часть звена списка (*величина любого простого или структурированного типа, кроме файлового*), *Next* — указатель на следующее звено списка; *First* — указатель на заглавное звено списка.

Для объявления списка сделано исключение: указатель на звено списка объявляется раньше, чем само звено.

В общем виде объявление списка:

Type U = ^Zveno;

Zveno = Record Inf : BT; Next: U End;

Здесь BT — некоторый базовый тип элементов списка.

Если указатель ссылается только на следующее звено списка, то такой список называют **однонаправленным**, если на следующее и предыдущее звенья — **двунаправленным списком**. Если указатель в последнем звене установлен не в Nil, а ссылается на заглавное звено списка, то такой список называется **кольцевым**. Кольцевыми могут быть и однонаправленные, и двунаправленные списки.

Типовые операции над СПИСКАМИ

- добавление звена в начало списка;
- удаление звена из начала списка;
- добавление звена в произвольное место списка, отличное от начала (например, после звена, указатель на которое задан);
- удаление звена из произвольного места списка, отличного от начала (например, после звена, указатель на которое задан);
- проверка, пуст ли список;
- очистка списка;
- печать списка.

1. Добавление звена в начало списка

```
Procedure V_Nachalo(Var First : U;  
X:BT);
```

```
  Var Vsp : U;
```

```
  Begin
```

```
    New(Vsp);
```

```
    Vsp^.Inf := X;
```

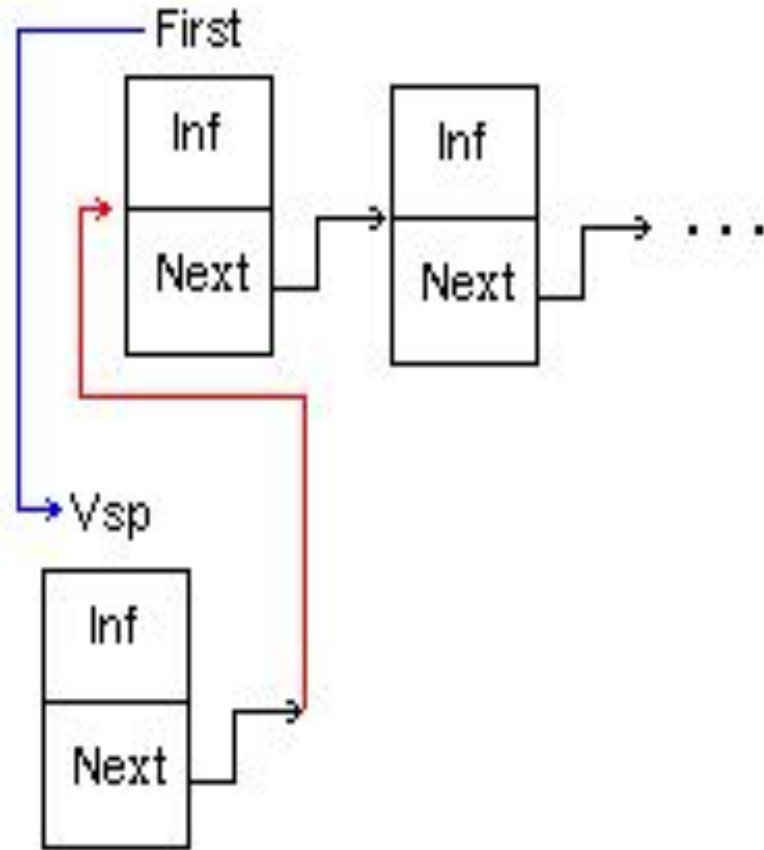
```
    Vsp^.Next := First;
```

{То звено, что было заглавным,
становится вторым по счёту}

```
    First := Vsp;
```

Новое звено становится
заглавным}

```
  End;
```



2. Удаление звена из начала списка

```
Procedure Iz_Nachala(Var First : U; Var X :  
BT);
```

```
  Var Vsp : U;
```

```
  Begin
```

```
    Vsp := First;
```

```
    {Ссылка забирается на текущее  
    заглавное звено}
```

```
    First := First^.Next;
```

```
    {Второе звено по счёту, становится  
    заглавным}
```

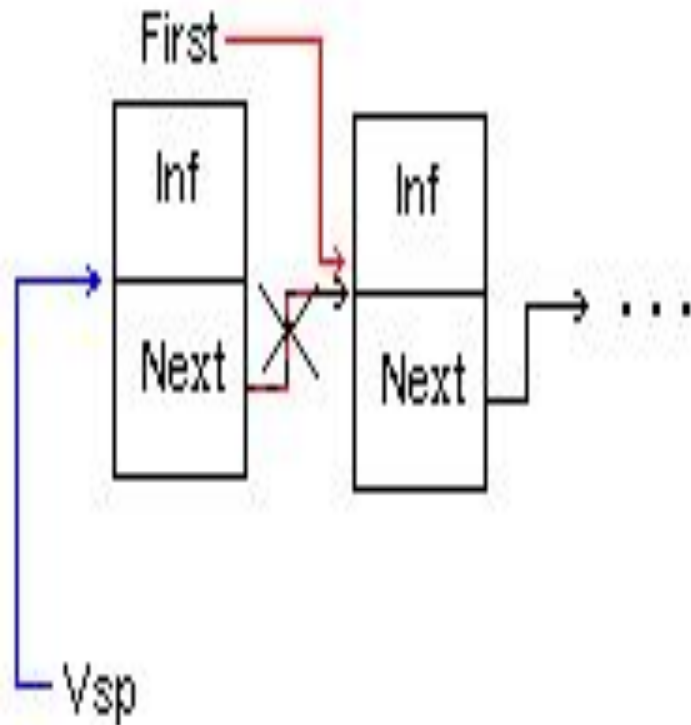
```
    X := Vsp^.Inf;
```

```
    {Информация из удаляемого звена}
```

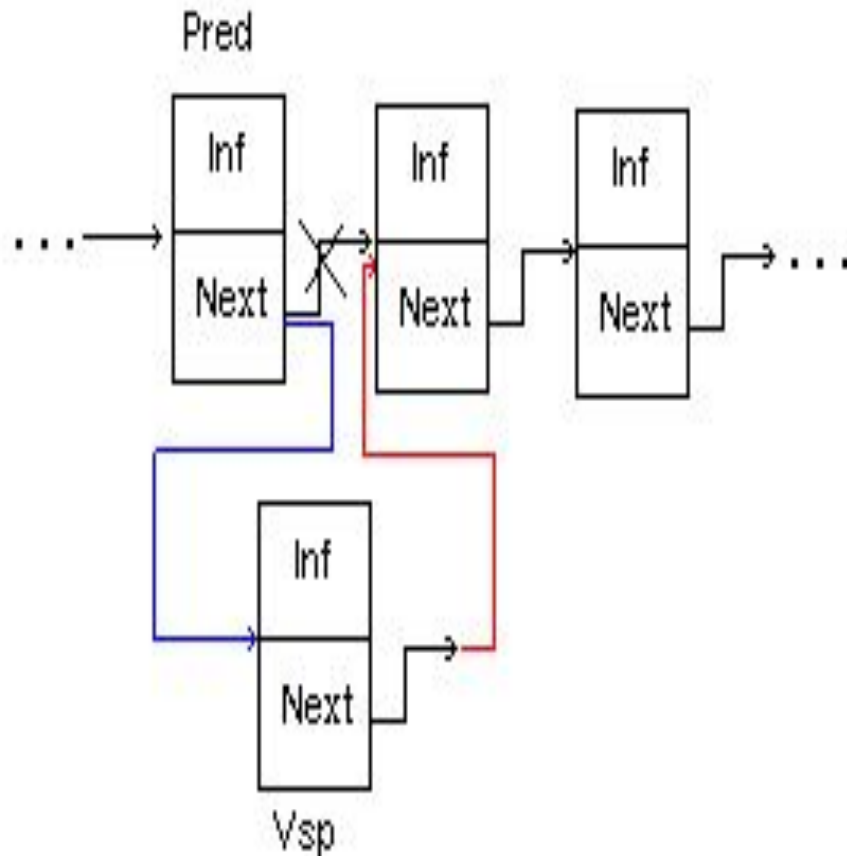
```
    Dispose(Vsp);
```

```
    {Уничтожается звено}
```

```
  End;
```



3. Добавление звена в произвольное место списка, отличное от начала (после звена, указатель на которое задан)



{Процедура добавления звена в список после звена, на которое ссылается указатель Pred; в x содержится информация для добавления}

```
Procedure V_Spisek(Pred : U; X : BT);
```

```
Var Vsp : U;
```

```
Begin
```

```
  New(Vsp); {Создается пустое звено}
```

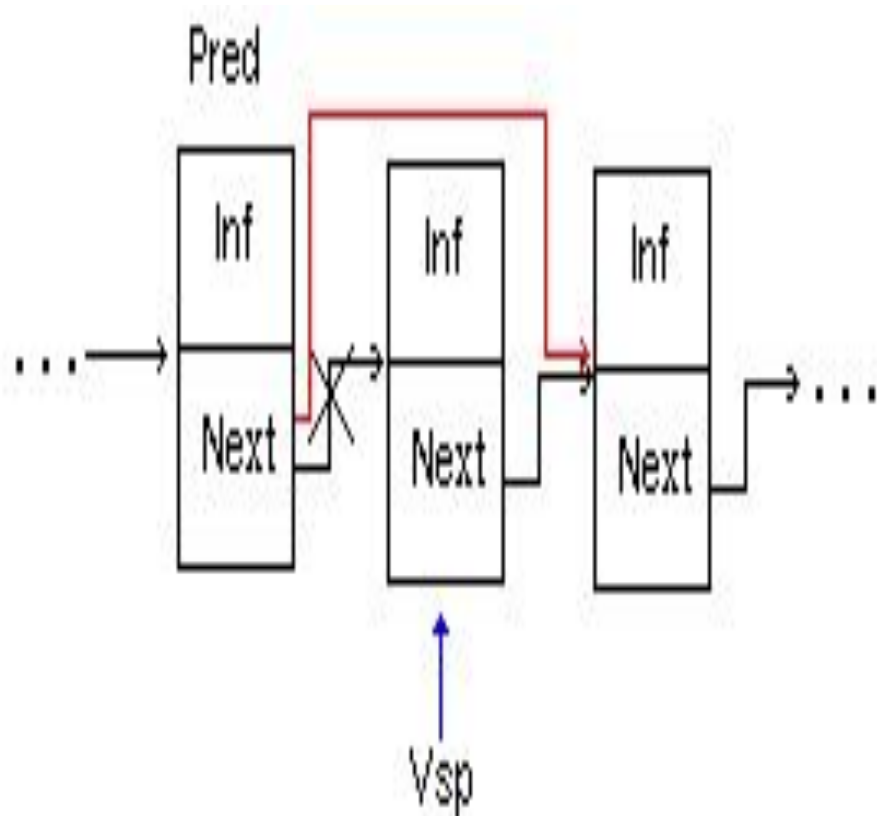
```
  Vsp^.Inf := X; {Заносится информация}
```

```
  Vsp^.Next := Pred^.Next; {Звено ссылается на то, что было следом за звеном Pred}
```

```
  Pred^.Next := Vsp; {Новое звено встало вслед за звеном Pred}
```

```
End;
```


4. Удаление звена из произвольного места списка, отличного от начала (после звена, указатель на которое задан)



{Процедура удаления звена из списка после звена, на которое ссылается указатель Pred; в x содержится информация из удалённого звена}

```
Procedure Iz_Spiska(Pred : U; Var X : BT);  
  Var Vsp : U;  
  Begin  
    Vsp := Pred^.Next; {Забирается ссылка на удаляемое звено}  
    Pred^.Next := Pred^.Next^.Next;  
    {Удаляется звено из списка, перенаправлением ссылки на следующее за ним звено}  
    X := Vsp^.Inf;  
    {Информация из удаляемого звена}  
    Dispose(Vsp);  
    {Уничтожение звена}  
  End;
```