

Лекция 2. Структуры

Понятие структуры

Структура - это набор из одной или более переменных, сгруппированных под одним именем для удобства обработки.

Объявление структуры

Синтаксис объявления структуры:

```
struct <тег шаблона структуры >
{
    <тип поля1> <имя поля1>;
    <тип поля2> <имя поля2>;
    ...
    <тип поляN> <имя поляN>;
};
```

- **struct** - служебное слово, используемое для определения структур;
- **Тег шаблона структуры** - имя, которым обозначают структуры данной конструкции, фактически играет роль типа структуры, хотя понятие тип не используется;
- **{}** - Скобки, ограничивающие перечень полей структуры;
- **Тип поля и имя поля** - стандартное объявление каждого из полей.

Объявление структуры

```
struct student
{
    char name[50];
    int order;
    float average_point;
};
```

```
student john = {"John", 2, 4.5};
std::cout << "Name: " << john.name << "\n";
std::cout << "Order: " << john.order << "\n";
std::cout << "Average point: " << john.average_point;
```

```
student john;
strcpy(john.name, "John");
john.order = 2;
john.average_point = 4.5;
std::cout << "Name: " << john.name << "\n";
std::cout << "Order: " << john.order << "\n";
std::cout << "Average point: " << john.average_point;
```

Результат:

```
Name: John
Order: 2
Average point: 4.5
```

Вложенность структур

```
struct info
{
    char address[100];
};
struct student
{
    char name[50];
    int order;
    float average_point;
    info data;
};
int main()
{
    student john = {"John", 2, 4.5, "My_street"};
    std::cout << "Name: " << john.name << "\n";
    std::cout << "Order: " << john.order << "\n";
    std::cout << "Average point: " << john.average_point << "\n";
    std::cout << "Address: " << john.data.address;
}
```

Результат:

Name: John

Order: 2

Average point: 4.5

Address: My_street

Массивы структур

Из структур одинакового тега может быть создан массив, точно так же как из данных других типов.

```
student ar[AR_SIZE] =
{
    {"John", 1, 2.5},
    {"Bill", 2, 3.5},
    {"Chris", 3, 5.0},
    {"Bonny", 4, 4.0},
    {"Megan", 5, 3.3}
};

for (int index = 0; index < AR_SIZE; index++)
    std::cout << "Name: " << ar[index].name << " Order: " << ar[index].order << " Average: " <<
ar[index].average_point << "\n";
```

Результат:

```
Name: John Order: 1 Average: 2.5
Name: Bill Order: 2 Average: 3.5
Name: Chris Order: 3 Average: 5
Name: Bonny Order: 4 Average: 4
Name: Megan Order: 5 Average: 3.3
```

Перегрузка операторов

```
struct Complex  
{  
    double re;  
    double im;  
};
```

```
Complex C_add( const Complex& x, const Complex& y )  
{  
    Complex t;  
    t.re = x.re + y.re;  
    t.im = x.im + y.im;  
    return t;  
}
```

```
Complex operator +( const Complex& x, const Complex& y )  
{  
    Complex t;  
    t.re = x.re + y.re;  
    t.im = x.im + y.im;  
    return t;  
}
```

Перегрузка операторов

```
int main()
{
    Complex u, v;
    u.re = 2.1; u.im = 3.6;
    v.re = 6.5; v.im = 7.8;

    Complex res1 = C_add(u, v);

    Complex res2 = u + v;

    std::cout << "Result1 = (" << res1.re << " + " << res1.im << "i)" << "\n";
    std::cout << "Result2 = (" << res2.re << " + " << res2.im << "i)";
    return 0;
}
```

Результат:

Result1 = (8.6 + 11.4i)

Result2 = (8.6 + 11.4i)

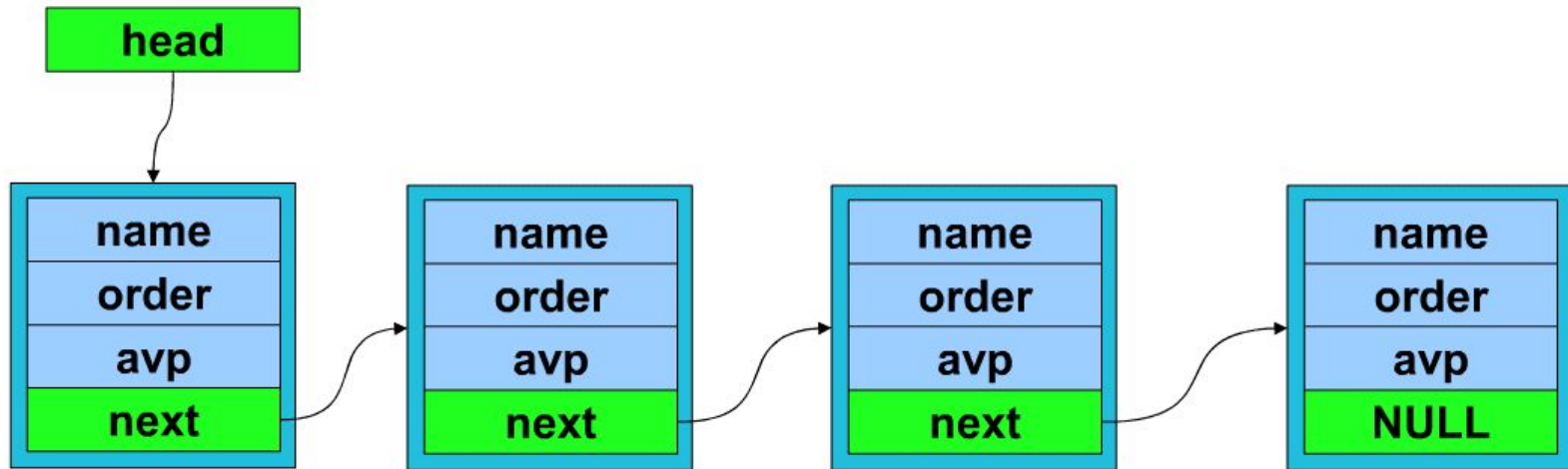
Списковые структуры

Списком называется упорядоченное, возможно пустое, множество (набор) элементов (**узлов**), которые состоят из **данных** и **полей связи** между узлами.

К спискам применимы ряд операций, например, **включения, исключения, копирования, поиска, сортировки.**

Списковые структуры

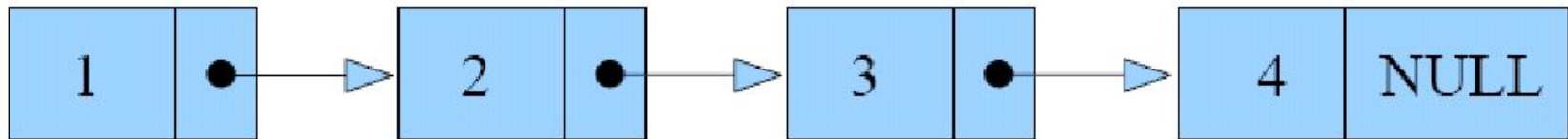
```
struct student
{
    char name[50];
    int order;
    float average_point;
    student *next;
};
```



Линейные односвязные списки

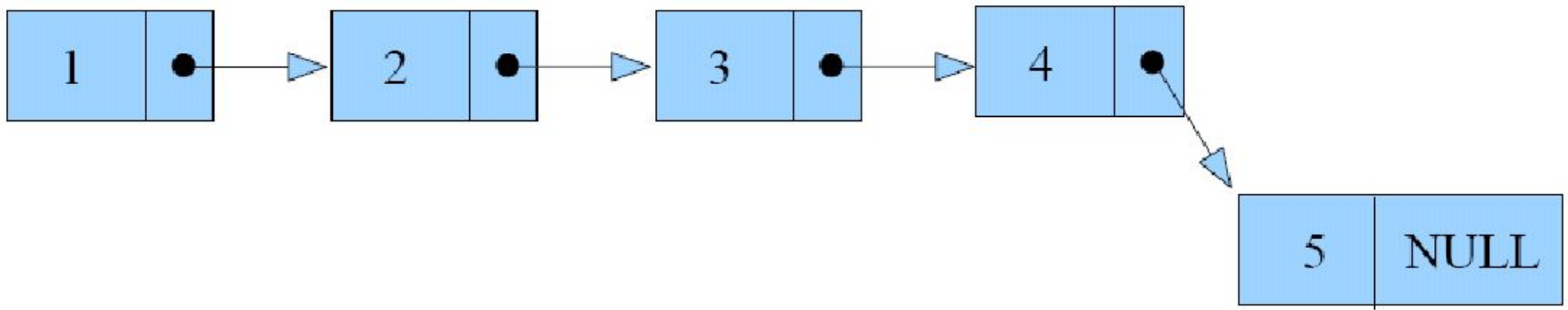
- 1) Односвязный список состоит из узлов (элементов), которые состоят из поля данных и указателя на следующий узел.
- 2) Начальный узел называется головой списка.
- 3) Значение указателя связи последнего узла равно **NULL**.
- 4) Полей данных может быть несколько.

Однонаправленный список



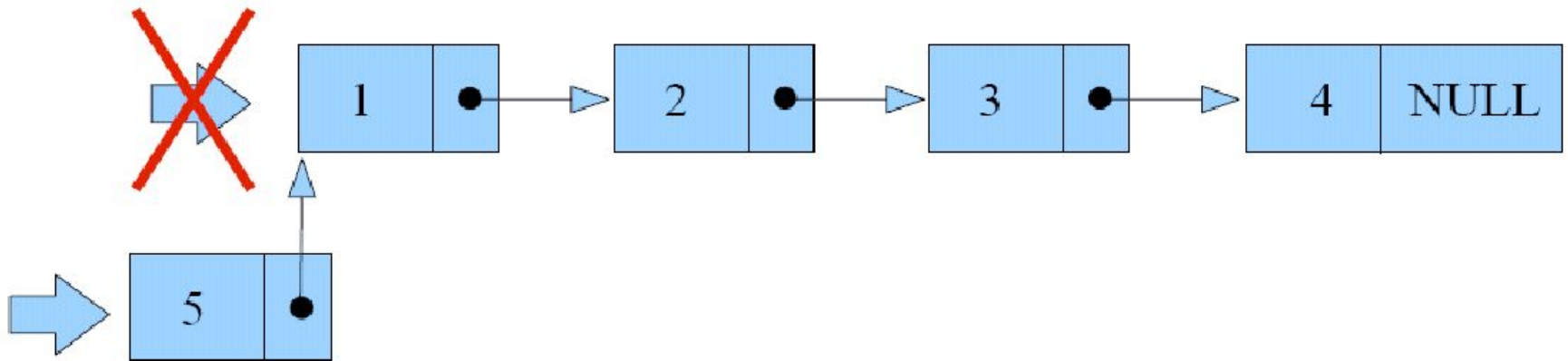
Линейные односвязные списки

Добавление в конец списка



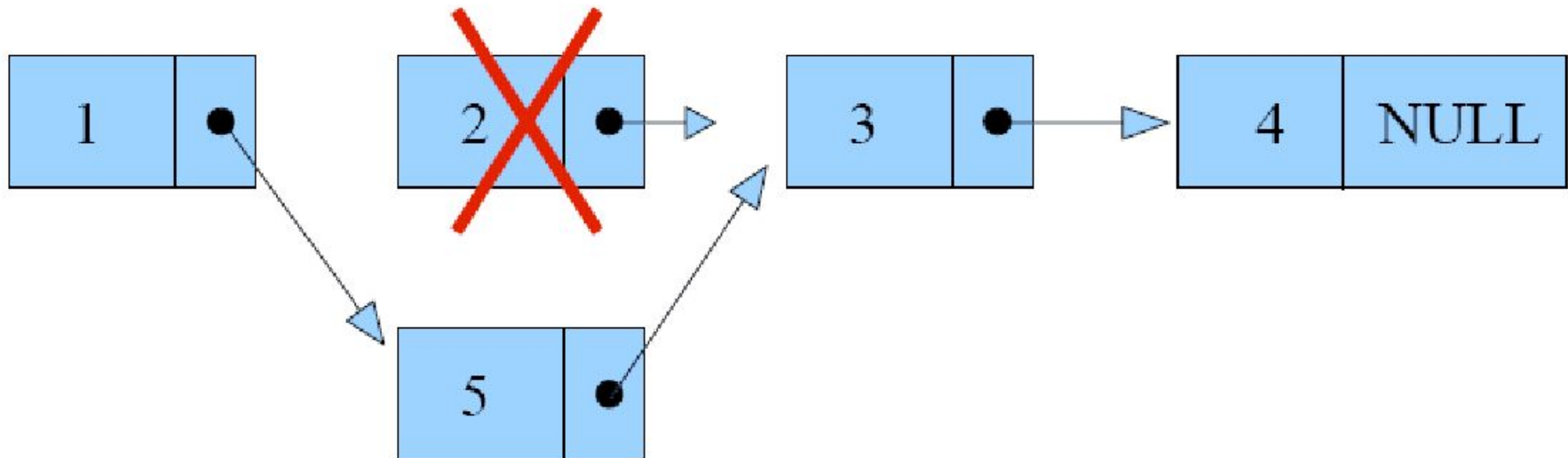
Линейные односвязные списки

Добавление в начало списка



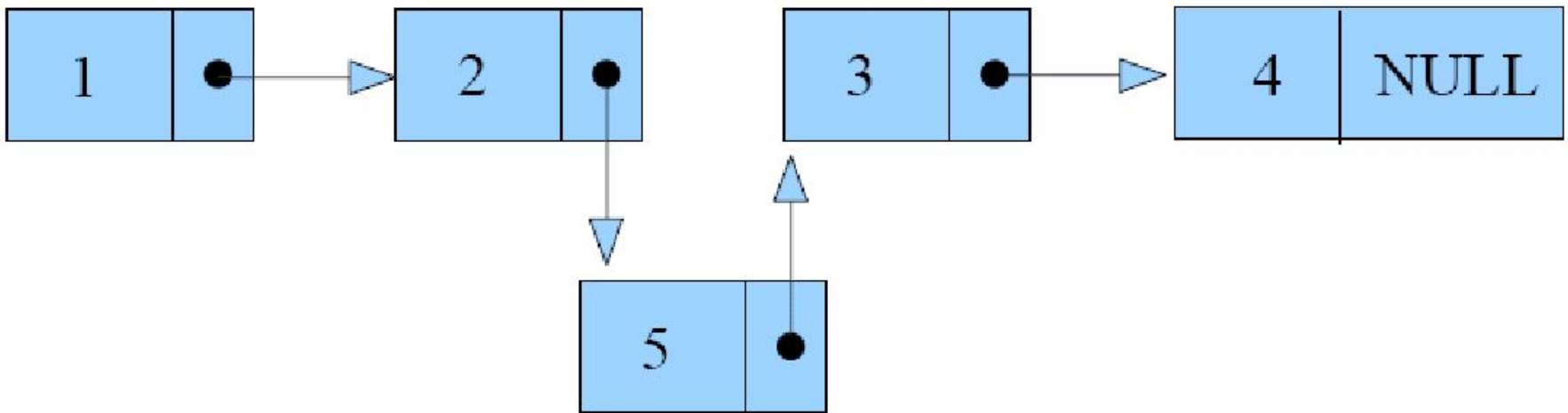
Линейные односвязные списки

Удаление из списка



Линейные односвязные списки

Вставка в список



Линейные односвязные списки

Сортировка списка

