**ЛЕКЦИЯ 7**

**ПЕРЕГРУЗКА ОПЕРАТОРОВ И НАСЛЕДОВАНИЕ**

**ДРУЖЕСТВЕННЫЕ ФУНКЦИИ (ДФ)**

1. Друж. функции не является компонентом класса, имеет доступ к защищенным и собственным компонентам класса, определена в классе со спецификатором friend;
2. Дружественная функция при вызове не использует this;
3. Объекты класса должны передаваться дф только через параметры;
4. Итого: дружественна функция не может быть компонентом класса, к которому является другом. Может быть глобальной или компонентом другого класса, который определен раньше.

**КЛАССЫ ДРУЗЬЯ (КД)**

1. Дружественный класс должен быть (если он друг) другом другому классу;
2. Все методы дк являются дружественными другому классу;
3. Дк должен быть определен вне тела класса, который предоставляет дружбу;

**ПЕРЕГРУЗКА ОПЕРАТОРОВ**

1. Есть множество операций, которые применены к данным определенного типа. Нам надо сделать так, чтобы операторы применялись корректно к классам. Решение вопроса – перегрузка операторов;
2. Перегрузка операторов позволяет определять действия, которые выполняет оператор;
3. Перегрузка – создание функции, создание которой содержит слово **operator** и символ оператора (**+**, **-**, **==** и т.д);
4. Может быть определена как член класса, внешняя функция, как дружественная функция;

**ИСПОЛЬЗОВАНИЕ ПАРАМЕТРОВ**

1. Операции бывают унарные (с одним) и бинарные (с двумя);
2. Могут быть определены способами:

А) Компонентная функция без параметров (с одним);

Б) Глобальная с одним (двумя) параметрами;

**УНАРНЫЕ ОПЕРАТОРЫ**

1. В постфиксных операторах должен быть один параметр (int). Он ***АБСОЛЮТНО БЕСПОЛЕЗЕН, НО БЫТЬ ОБЯЗАН***;

**БИНАРНЫЕ ОПЕРАТОРЫ**

1. Для перегрузки необходимо определить приоритет и ассоциативность;
2. Приоритет – порядок операции в выражении если операций несколько;
3. Ассоциативность – порядок выполнения нескольких операторов с одинаковым приоритетом (слева или справа);

**ПЕРЕГРУЗКА ОПЕРАТОРА ПРИСВАИВАНИЯ**

1. Оператор не наследуется;
2. Определен по умолчанию для каждого класса определен как побитовое копирование значений другого класса (включая его адрес);
3. Может быть перегружен только как функция – компонент класса;
4. Стоит перегружать, когда класс содержит динамически инициализированные элементы, чтобы избежать утечку памяти.