**Умные указатели.**

При работе с кучей необходимо не забывать делать delete в пару к каждому new. Иначе в ОП будут объекты, ссылок на которые не осталось, но место они занимают. Напишем несколько таких же оболочек, например scoped\_ptr – оболочка для указателей. Хочется получить что-то такое:

{

scoped\_ptr p = new Object (); .... работа с p ..... // внутри scoped\_ptr будет содержаться указатель на Object

} // здесь должен вызваться delete и освободиться память

Scoped\_Ptr p1 = new Object (); // когда p1 дойдёт до конца области видимости вызовется деструктор, всё отлично, минусы: любая безопасность несёт за собой замедление работы.

Scoped\_Ptr p2=p1; // плохо, т.к. будет двойной delete в конце области видимости.

Scoped\_Ptr p2 = new Object(); p2=p1; // сначала утечка памяти, а потом опять двойной delete.

Как решать проблемы? Запретить перегрузку operator=. Конструкторы копирования обычно нужны для передачи параметров в функцию по значению, создается копия в стеке и с ней функция работает. Наш указатель не должен выходить за область видимости, никуда в функции передаваться не должен, его задача работать в области видимости.

Auto\_Ptr (семантика передачи владением).

Shared\_Ptr (семантика разделения владением).