Лабораторная работа II A
Сетка.

В некоторой вычислительной модели для работы с данными двумерной сетки используется простой массив, память под который выделяется выражением **new**[] и возвращается выражением **delete**[]. С ростом количества кода и его усложнением такой подход начал приводить к многочисленным ошибкам при работе с памятью. Разрешить эту проблему может грамотная реализация класса **Grid**, инкапсулирующего управление памятью. Сконструируйте такой класс.

Класс **Grid** должен содержать следующие открытые методы, облегчающие работу с двумерными данными:

```
class Grid {
private:
    float *memory;
    size_t x_size, y_size;

public:
    Grid(size_t x_size, size_t y_size);

    float operator()(size_t x_idx, size_t y_idx) const;
    float& operator()(size_t x_idx, size_t y_idx);

    size_t get_xsize() const;
    size_t get_ysize() const;

    Grid& operator=(float);

    friend std::ostream& operator<<(std::ostream&, Grid const&);
    friend std::istream& operator>>(std::istream&, Grid &);
};
```

Класс должен содержать также и другие методы, необходимые для упрощения управления памятью. Предполагается, что в дальнейшем данный класс будет использоваться не только для чисел типа **float**, но и для других данных, допускающих конструирование по умолчанию и копирование.

Поведение оператора круглых скобок при вызове со значениями, выходящими за допустимые для сетки пределы, не определено. Поведение конструктора не определено для случая нулевого размера сетки.
Оператор присваивания, принимающий одиночное значение, заменяет все элементы сетки на значение входящего параметра.

Лабораторная работа II B
Подсетка.

Усложнение модели привело к тому, что возникла необходимость хранить в элементе сетки другую сетку для увеличения разрешения данных. Теперь каждый элемент сетки может быть либо числом, либо вложенной сеткой. Для работы с вложенными сетками следует добавить дополнительныt методы:

```
class Grid {
public:
    Grid& make_subgrid(
        size_t x_idx, size_t y_idx,
        size_t x_sub_size, size_t y_sub_size);

    Grid& collapse_subgrid(
        size_t x_idx, size_t y_idx,
        size_t x_sub_size, size_t y_sub_size);

    Grid& get_subgrid(size_t x_idx, size_t y_idx);
    Grid const& get_subgrid(size_t x_idx, size_t y_idx);

    bool is_subrid(size_t x_idx, size_t y_idx) const;
};
```

Метод **make_subgrid** создаёт на месте элемента с индексом (x_idx, y_idx) вложенную сетку **Grid** размером (x_sub_size, y_sub_size), каждый элемент которой равен значению элемента в родительской ячейке в момент вызова. Метод возвращает ссылку на объект, на котором произведён вызов. Если в указанном месте уже существовала подсетка, старая подсетка удаляется.
Метод **collapse_subgrid** уничтожает вложенную сетку и помещает на её место значение, равное среднему арифметическому по вложенной сетке. Метод возвращает ссылку на объект, на котором произведён вызов метода. Если в указанном месте не было подсетки, то никаких действий не производится.
Метды **get_subgrid** позволяют получить ссылки на подсетку. Если в указанной ячейки нет подсетки, поведение метода не определено.
Метод **is_subgrid** возвращает **true** в случае, если в данной ячейке находится вложенная сетка.
Для всех методов результат вызова с индексами, выходящими за пределы сетки, или недопустимыми размерами для конструктора **Grid** не определён.
Операторы круглых скобок должны вести себя ожидаемым образом на элементах не являющихся подсетками. В случае обращения к подсетке перегрузка для константного объекта возвращает копию среднего значения на сетке и перегрузка для неконстантного объекта заменяет все значения подсетки на правую часть оператора присваивания.