Bignum

mp::bignum

Напишите класс длинного целого неотрицательного числа **bignum**. Такое число представляет собой аналог unsigned int, однако может иметь неограниченное заранее количество разрядов.

Класс должен обладать как минимум следующим небольшим списком операций:

1. конструктор по умолчанию инициализирует нулём;
2. неявное конструирование от целого числа (uint32_t);
3. явное конструирование от десятичной строки (std::string). Допускается квадратичная трудоемкость;
4. копирование (конструктор и оператор присваивания);
5. явные преобразования к целому числу (uint32_t); если число не помещается в uint32_t, происходит сужение значения (как int в short);
6. возможность использования в условных выражениях (true, если не ноль);
7. класс не должен позволять делать вычитание;
8. функция получения десятичного строкового представления to_string. Допускается квадратичная трудоемкость;
9. ввод/вывод в стандартные потоки (std::ostream/std::istream), ввод/вывод аналогичен обычным целым числам в десятичном представлении, поддержка манипуляторов ввода вывода не требуется;
10. операции +, *, +=, *= (запятая здесь - всего лишь разделитель перечисления, перегружать ее не надо); Умножение имеет квадратичную трудоемкость.

Дополнительные требования:

1. Класс должен называться bignum и находится в пространстве mp (multi precision).
2. Решение необходимо оформить как header-only библиотеку, состоящую из одного файла bignum.hpp.
3. bignum может быть реализован через шаблонный класс, но сам bignum - уже нешаблонный.
4. Все операции над длинными числами, если не указано иного, должны выполняться за линейное время от количества разрядов входных длинных чисел.
5. Базой системы счисления в bignum должен выступать тип uint32_t (с основанием 2^{32}).

Рекомендации:

1. Если какая-то функция может быть реализована через public интерфейс класса, сделайте ее внешней.

mp::polynomial

Реализуйте класс **polynomial** в области видимости mp. Реализует абстракцию полином
$P(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots a_0$
Коэффициенты - целые неотрицательные короткие uint32_t.

Операции:

1. Явный конструктор от строки формата: 5*x^3+2*x^1+6*x^0 (без пробельных символов, степень указана всегда, коэффициент может отсутствовать).
2. Функции at (константная и неконстантная) для доступа к коэффициенту полинома по индексу. Если конструктором полином задан меньшей степени, автоматически расширяет его до нужной степени (для неконстантной).
3. Оператор () для вычисления полинома в точке. Шаблонный по типу аргумента (может быть int, может double, а может и bignum). Или хотя бы перегрузка по int, uint32_t, double, bignum. Результат имеет тот же тип, что и аргумент. **Внимание: количество умножений на значение аргумента должно быть не более степени полинома, т.е. необходимо реализовать схему Горнера.**

Дополнительное задание

Реализуйте small object optimization для bignum. Это означает, что если ваше число меньше некоторого значения (для определенности, в данном задании, помещается в размер указателя, т.е. uint64_t), то не происходит выделение памяти. Как только стало больше - выделения уже не избежать.
Успешное выполнение этого задания дает вам право на еще одну проверку любого дз.