Matemáticas.

Material de referencia.

Índice

1. Big Numbers.

1.1. Implementación

```
#include <iostream>
#include <algorithm>
#include <utility>
using namespace std;

typedef string BigInteger;

//Regresa el i-esimo digito de derecha a izquierda de un numero.
int digit(const BigInteger &num, int i) {
    if (i < num.size())
        return num[num.size() - 1 - i] - '0';
    return 0;
}

//Compara dos numeros y regresa: 1 si el primero es mayor; 0 si son iguales; -1 si el segundo es mayor.
int compareTo(const BigInteger &a, const BigInteger &b) {
    for (int i = max(a.size(), b.size()) - 1; i >= 0; --i) {
        if (digit(a, i) > digit(b, i))
            return 1;
        if (digit(b, i) > digit(a, i))
            return -1;
    }
    return 0;
}

//Regresa la suma de dos numeros.
BigInteger sum(const BigInteger &a, const BigInteger &b) {
    BigInteger ans;
    int carry = 0;
    for (int i = 0; i < max(a.size(), b.size()); ++i) {
        carry += digit(a, i) + digit(b, i);
        ans.push_back((carry %10) + '0');
        carry /= 10;
    }
    if (carry)
        ans.push_back(carry + '0');
    reverse(ans.begin(), ans.end());
    return ans;
}

//Regresa la diferencia de dos numeros. El primer numero debe ser mayor o igual que el segundo.
BigInteger substract(const BigInteger &a, const BigInteger &b) {
    BigInteger ans;
    int carry = 0;
    for (int i = 0; i < a.size(); ++i) {
        carry += digit(a, i) - digit(b, i);
        if (carry >= 0) {
            ans.push_back(carry + '0');
            carry = 0;
        }
```

```
51          else {
52              ans.push_back(carry + 10 + '0');
53              carry = -1;
54          }
55      }
56      while (ans.size() > 1 && ans.back() == '0')
57          ans.pop_back();
58      reverse(ans.begin(), ans.end());
59      return ans;
60  }
61
62  //Regresa el producto de dos numeros (BigInteger x int).
63  BigInteger multiply(const BigInteger &a, int b) {
64      if (b == 0)
65          return "0";
66      BigInteger ans;
67      int carry = 0;
68      for (int i = 0; i < a.size(); ++i) {
69          carry += digit(a, i) * b;
70          ans.push_back((carry %10) + '0');
71          carry /= 10;
72      }
73      while (carry) {
74          ans.push_back((carry %10) + '0');
75          carry /= 10;
76      }
77      reverse(ans.begin(), ans.end());
78      return ans;
79  }
80
81  //Regresa el producto de dos numeros (BigInteger x BigInteger).
82  BigInteger multiply(const BigInteger &a, const BigInteger &b) {
83      BigInteger ans;
84      for (int i = 0; i < b.size(); ++i)
85          ans = sum(ans, multiply(a, digit(b, i)).append(i, '0'));
86      return ans;
87  }
88
89  //Regresa el cociente y el residuo de la division (BigInteger / int).
90  pair<BigInteger, int> divide(const BigInteger &a, int b) {
91      pair<BigInteger, int> ans;
92      for (int i = a.size() - 1; i >= 0; --i) {
93          ans.second = 10*ans.second + digit(a, i);
94          if (!ans.first.empty() || ans.second >= b || i == 0)
95              ans.first.push_back((ans.second / b) + '0');
96          ans.second %= b;
97      }
98      return ans;
99  }
100
101 //Regresa el cociente y el residuo de la division (BigInteger / BigInteger).
102 pair<BigInteger, BigInteger> divide(const BigInteger &a, const BigInteger &b)
        {
103     pair<BigInteger, BigInteger> ans;
104     BigInteger table[10];
105     for (int i = 0; i < 10; ++i)
```

```
106             table[i] = multiply(b, i);
107         for (int i = a.size() - 1; i >= 0; --i) {
108             int q = 0;
109             ans.second.push_back(digit(a, i) + '0');
110             while (q < 9 && compareTo(ans.second, table[q + 1]) >= 0)
111                 ++q;
112             if (!ans.first.empty() || q > 0 || i == 0)
113                 ans.first.push_back(q + '0');
114             ans.second = substract(ans.second, table[q]);
115         }
116         return ans;
117     }
118
119     int main() {
120         BigInteger a, b;
121         cin >> a >> b;
122
123         cout << a << " + " << b << " = " << sum(a, b) << '\n';
124         if (compareTo(a, b) >= 0)
125             cout << a << " - " << b << " = " << substract(a, b) << '\n';
126         else
127             cout << b << " - " << a << " = " << substract(b, a) << '\n';
128         cout << a << " * " << b << " = " << multiply(a, b) << '\n';
129         cout << a << " = " << b << " * " << divide(a, b).first << " + " << divide(
            a, b).second << '\n';
130
131         return 0;
132     }
```

Entrada	**Salida**
1894821 589613	1894821 + 589613 = 2484434 1894821 - 589613 = 1305208 1894821 * 589613 = 1117211094273 1894821 = 589613 * 3 + 125982