for practice session.txt

```
1. Проверить ONLINE_JUDGE  и LOCAL
2. Какой вердикт даёт throw
```

template.txt

```
//VISUAL ONLY:
//-----------------------------------
#define _CRT_SECURE_NO_WARNINGS
#pragma comment(linker, "/STACK:16777216")

#include <iostream>
#include<vector>
#include<string>
#include<map>
#include<algorithm>
#include<deque>
#include<set>
#include<queue>
#include<stack>

//-------------------------------------

//GCC:
#include <bits/stdc++.h>

using namespace std;

typedef long long ll;
typedef unsigned long long ull;

int main()
{
    ios_base::sync_with_stdio(0);
    cin.tie(0);
    cout.tie(0);
#ifndef ONLINE_JUDGE //МБ LOCAL
    freopen("in.txt", "rt", stdin);
    freopen("out.txt", "wt", stdout);
#endif

return 0;
}


НАДО ДОБАВИТЬ ВРЕМЯ И РАНДОМ. Проверь throw
```

1 Number theory

extended euclid.txt

```
int gcd (int a, int b, int & x, int & y) {
if (a == 0) {
x = 0; y = 1;
return b;
}
int x1, y1;
int d = gcd (b%a, a, x1, y1);
x = y1 - (b / a) * x1;
y = x1;
return d;
}
```

modulo inverse.txt

```
//Решаем a*b=1 mod m относительно b
//Сводим к a*x+m*y=1 -> ax=1 mod m
//gcdex - extended euclid
int x, y;
int g = gcdex (a, m, x, y);
if (g != 1)
cout << "no solution";
else {
x = (x % m + m) % m;
cout << x;
}
```

all modulo inverses.txt

```
//Для всех чисел [1,m-1] находим обратное по модулю m

r[1] = 1;
for (int i=2; i<m; ++i)
r[i] = (m - (m/i) * r[m%i] % m) % m;
```

bigInt.txt

```
typedef vector<int> lnum;
const int base = 1000*1000*1000;

void print(lnum& a)
{
printf ("%d", a.empty() ? 0 : a.back());
for (int i=(int)a.size()-2; i>=0; --i)
printf ("%09d", a[i]);
}

void read(lnum& a)
{
sting s;
```

```
cin>>s
for (int i=(int)s.length(); i>0; i-=9)
if (i < 9)
a.push_back (atoi (s.substr (0, i).c_str()));
else
a.push_back (atoi (s.substr (i-9, 9).c_str()));
}
//a+=b
void add(lnum& a,lnum& b)
{
int carry = 0;
for (size_t i=0; i<max(a.size(),b.size()) || carry; ++i) {
if (i == a.size())
a.push_back (0);
a[i] += carry + (i < b.size() ? b[i] : 0);
carry = a[i] >= base;
if (carry)  a[i] -= base;
}


//a-=b
void sub(lnum& a,lnum& b)
{
int carry = 0;
for (size_t i=0; i<b.size() || carry; ++i) {
a[i] -= carry + (i < b.size() ? b[i] : 0);
carry = a[i] < 0;
if (carry)  a[i] += base;
}
while (a.size() > 1 && a.back() == 0)
a.pop_back();
}


// b<base
//a*=b
void mul(lnum& a,int b)
{
int carry = 0;
for (size_t i=0; i<a.size() || carry; ++i) {
if (i == a.size())
a.push_back (0);
long long cur = carry + a[i] * 1ll * b;
a[i] = int (cur % base);
carry = int (cur / base);
}
while (a.size() > 1 && a.back() == 0)
a.pop_back();
}
```

```
//c=a/b
lnum div(lnum& a,lnum& b)
{
lnum c (a.size()+b.size());
for (size_t i=0; i<a.size(); ++i)
for (int j=0, carry=0; j<(int)b.size() || carry; ++j) {
long long cur = c[i+j] + a[i] * 1ll * (j < (int)b.size() ? b[j] : 0) + carry;
c[i+j] = int (cur % base);
carry = int (cur / base);
}
while (c.size() > 1 && c.back() == 0)
c.pop_back();
return c;
}

//a/=b,  b<base
void div(lnum& a, int b)
{
int carry = 0;
for (int i=(int)a.size()-1; i>=0; --i) {
long long cur = a[i] + carry * 1ll * base;
a[i] = int (cur / b);
carry = int (cur % b);
}
while (a.size() > 1 && a.back() == 0)
a.pop_back();
}
```

2 Data structures

default segment tree.txt