大数乘法和阶乘

PB17081531 沈鹏飞

结果

大数乘法：

```
1111111111
11111111111111111111
123456790111111111110987654321

─ program is finished running (dropped off bottom) ─
```

mathematica参考结果：

```
In[1]:= 1111111111 * 11111111111111111111

Out[1]= 123456790111111111110987654321

In[2]:= % - 123456790111111111110987654321

Out[2]= 0
```

阶乘

尚未调出正确结果，等得到正确结果再去检查。

编译代码对比

```cpp
#pragma once
#include <memory>

class BigInt
{
    BigInt();
    BigInt(const BigInt& big);
    BigInt(BigInt&& big) noexcept;

    BigInt& operator=(const BigInt& big);
    BigInt& operator=(BigInt&& big)noexcept;
    ~BigInt();

    BigInt operator+(const BigInt& big) const;
    BigInt operator-(const BigInt& big) const;
    BigInt operator*(const BigInt& big) const;
    BigInt operator/(const BigInt& big) const;
    BigInt operator%(const BigInt& big) const;

private:
    unsigned size;
    std::unique_ptr<int> data;
};
```

一个标准的类型声明。实现方面想要得到工业级别的安全代码需要一系列复杂的设置，没有必要在此处完成。后面啥时候用得到再实现一个吧。

收获

感觉有这功夫不如去玩玩用Cmake实现大数乘法和递归阶乘，对于工程组织工具的使用能有一定训练。双学位的组原课程我只关心一件事情，那就是怎么让我的代码运行得更快，而目前从课程和作业都没有得到这方面的知识。这辈子也不太会用到汇编语言，现在能比编译器聪明的人也是凤毛麟角了。一个工业生产流水线上应该由不同位置的人来做不同位置的事，该把自己层次的事情做好了封装给另一层就完事了，这种操作寄存器的操作令人反胃。总体来说收获不大，浪费了人生的几个小时。