关于高精度实现的注意事项

一、命名

- 一个数被储存在名为 `Data` 的结构体中
- 结构体包含 `number` 和 `symbol` 两个变量
    1. `number` 为 `vector` 类型结构体
    2. `symbol` 为 `int` 类型整数取值为 -1 或 $+1$

二、使用说明

- `vector` 类型可以理解为动态长度数组
- 在编译自己写的代码时需要加上头文件

```
#include "vector.h"
```

  并将 `vector.h` 与你需要编译的源代码放在一个文件夹下

- 使用方法

```
vector x;//定义一个vector
x.init();//vector被定义后就应执行，防止出错
x.clear();//清空vector中所有元素
x.push_back(int y)//向vector的末尾加入一个元素
x.pop_back()//删除vector的末尾元素
x.size()//返回vector的大小
x[y]//可以理解为数组的访问方式
vector 的 下标为 [0,...,size()-1]
```

三、高精度实现

- `[0,...,n]` 位保存一个数的整数位，`[n+1,...,n+50]` 保存小数位
- 保存的位数由高到低
- 大家需要完成对 `vector` 类型运算符的重载，即完成以下函数

```
vector operator * (const vector &a, const vector &b)
{
    vector c;
    //运算 c=a*b
    return c;
}
```

格式为

```
返回值类型 operator 需要重载的运算符 (变量1, 变量2)
```

文件确认无误后，将后缀名改为 `.h` 并去掉 `main()` 函数，将文件名该为你实现的功能

例如：实现的高精度乘法，那么你的文件名为 `multiplication.h`