HW0

复现结果如下图：

```
cdm@cdm-virtual-machine:~/桌面/DS$ time ./test
933262154439441526816992388562667004907159682643816214685929638952175999932299156089414639761565182862536979208272237582511852109168640000000000000000000000000

real    0m0.002s
user    0m0.002s
sys     0m0.000s
cdm@cdm-virtual-machine:~/桌面/DS$ time python3 test.py
933262154439441526816992388562667004907159682643816214685929638952175999932299156089414639761565182862536979208272237582511852109168640000000000000000000000000

real    0m0.015s
user    0m0.005s
sys     0m0.010s
```

Q1:

1. C++是一种静态类型语言，需要在编译时指定变量的数据类型。这使得编译器可以更好地对代码进行优化和生成高效的机器码。而 Python 是一种动态类型语言，它在运行时才能确定变量的类型，这就导致 C++可以先编译为机器文件然后直接执行，而 Python 则需要在运行的同时解释各个语句；

2. Python 封装了很多函数以及类，这使得 Python 语言的代码编写非常的简洁，比等效的 C++代码要节省很多时间和空间。但是相应的，这也导致了 Python 在执行编译时需要进行大量的查找调用，自然的也就多耗用了时间；

3 .除此之外，还有一些系统上的原因，比如 C++允许直接访问计算机的底层资源和硬件，这使得 C++可以更好地利用计算机的特殊功能和高效的底层库。Python 作为高级语言，更多关注开发效率和易用性，对底层资源的访问相对较为受限。

总的来说，Python 语言用一定的运行性能换取了开发时间的缩短，这使得目前许多程序员会选择 Python 开发项目而不是 C++，因为在大部分情况下，开发流程的缩短远比运行性能要重要。

Q2:

结果如下图：

```
cdm@cdm-virtual-machine:~/桌面/DS$ time ./test
0

real    0m0.001s
user    0m0.001s
sys     0m0.000s
```

原因：程序结果远大于 int_max，原程序使用了 GMP 库中的 mpz_class 类型来处理大整数，避免了溢出问题。而改成 int 之后会导致溢出，也就是把原来的结果进行截断，只剩下了最后几位的 0，因此结果就是 0。

Q3:

除了编程语言之外，程序运行速度的快慢还可能受到以下因素的影响：

1. 算法复杂度：优秀的算法可以显著提高程序的运行效率;
2. 数据规模：处理的数据规模越大，程序运行的时间通常会更长;
3. 硬件性能：计算机的硬件性能（如 CPU、内存、硬盘）直接影响程序的运行速度;
4. 并发与并行：合理利用并发编程和并行计算可以加快程序的运行速度，充分利用多核处理器和多线程技术可以提高程序的效率;
5. I/O 操作：频繁的 I/O 操作（如文件读写、网络通信）会消耗大量时间;