In this workshop, I had to the chance to learn a new system library called chrono. By using the methods and objects from that library. I am able to track the performance of my code by displaying the duration of certain specific tasks, which is pretty nice. Apart from that, I learned some interesting knowledges like nameless classes, move operator, move contructor, and most importantly Rvalues and Lvalues, and more modern C++ coding tricks like C++ foreach style loop.

One of the things, I would like to point out is move or move constructor is actually significantly faster than the destructor or the one argument constructor. I assume the reason is because inside my code, I used the shallow copy instead of the deep copy, which means the computer does not need to allocate a new chunk of memory for the current object and copy the values from the original copy. It basically just moved everything from a existing object onto a new object.

Here is the code snippet from my Protein.cpp module:

This is the copy assignment operator. As I said, the current object will be allocated with memory and then deep copy all the values from the source.

ProteinDatabase& ProteinDatabase::operator=(const ProteinDatabase& src){

if (this!=&src){

delete[] this->recordedArr;

if (src.recordedArr){

this->recordedArr = new std::string[this->recordNum];

for (unsigned i = 0u; i < this->recordNum; i++){

this->recordedArr[i] = src.recordedArr[i];

}

}

}

return \*this;

}

This is the move assignment operator, I pretty much just used the current object’s pointer pointing to the resource of the src. Then let the original pointer pointing to nullptr.

ProteinDatabase& ProteinDatabase::operator=(ProteinDatabase&& src)noexcept{

if (this!=&src){

delete[] this->recordedArr;

this->recordedArr = nullptr;

if (src.recordedArr){

this->recordedArr = src.recordedArr;

src.recordedArr = nullptr;

}

}

return \*this;

}

Therefore, the move constructor and assignment are significantly faster than the copy constructor and copy assignment speaking of the run time speed.

Here is part of my output, which show the significant time difference between those two types of operation.

0-arg Constructor 300

1-arg Constructor 2527616900

Copy Constructor 29934500

Copy Assignment 34185500

Move Constructor 700

Move Assignment 300

Destructor 43853800

As for the difficulty, I would say the hardest part would be using a new library that I have never been exposed from. I did spend a lot of time on understanding the syntax and code examples by reading the materials from C++ reference and watching a lot of videos. Therefore, I was able to overcome it before the professor actually discuss this workshop on Tuesday’s lecture.

Other than that, everything was quite smooth, and I am looking forward to doing more workshops, which I can practice more for whatever I learned from the lectures.