// OOP345 Workshop 8: **Smart Pointers**

// File: reflect.txt

// Name: Jingmin (Jessica) Zhou

// Email: jzhou175@myseneca.ca

// ID: 119766194

// Date: 07/17/2020

In Workshop 8 (Smart Pointers), I have learned how to merge two lists into another one through raw pointer and smart pointer, and the difference of raw pointer and smart pointer, how the smart pointer to ensure the memory deallocation in possible presence of an exception.

On one hand, smart pointers implement the RAII, which means dynamic memory for an object is allocated at the same time as the pointer to that object is created. Smart pointers will ensure the memory deallocation automatically and simplify the memory management for an object that throws an exception. Take the following piece of code for example: unique\_ptr was used to create the object p and it’ no need to worry about the delete process and memory leak even if the object throws a validate() exception.

if (desc[i].code == price[j].code) {

std::unique\_ptr<Product> p(new Product(desc[i].desc, price[j].price));

p->validate();

priceList += p;

}

On the other hand, smart pointer is class that wraps a pointer and implements the deallocation of memory that pointer points to, while raw pointer is a built-in type that is a variable and hold an address. The most difference between raw pointer and smart pointer is the management of memory they point to. Smart pointer knows its object and can manage the memory of the object to which it points and avoid memory leak even if the object throw an exception. Raw pointer does not know how to allocate the memory being pointed and when or how to deallocate it.

===========================================================================================================================================

Quiz 6 reflection: 10/10.