Name: Jingmin Zhou (Jessica)

Student ID: 119766194

Email: [jzhou175@myseneca.ca](mailto:jzhou175@myseneca.ca)

In the Workshop 8 (Virtual Functions), I have learned how to implement an abstract definition of behavior (e.g. setSalary(), bonus(), workHours() and display()) for a specific type (Doctor or Engineer class). I have learned how to define a pure virtual function, code an abstract base class and implement behavior declared in a pure virtual function.

Firstly, an abstract base Employee was coded with defining several pure virtual functions, such as setSalary(), bonus(), workHours() and display(). Theses pure virtual functions as interfaces were implemented differently in two different concrete classes, Doctor and Engineer.

Secondly, an abstract base class that contains or inherits a pure virtual function cannot be instantiated because abstract class provides no implementation(s) for its pure virtual function(s). A concrete class is a derived class that implements the pure virtual function(s)/interface(s) defined in its abstract base class. Each concrete class gives a specific meaning to the interface.

Finally, it is a good practice to code the destructor in a base class as virtual (e.g. virtual ~Employee()), even if no class is currently derived from that base class, which ensures that the most derived destructor will be called if and when a class is derived from the base class without requiring an upgrade to the definition of the base class.