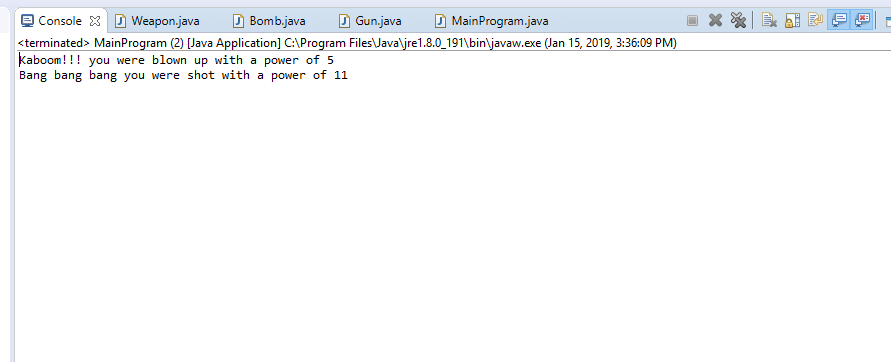
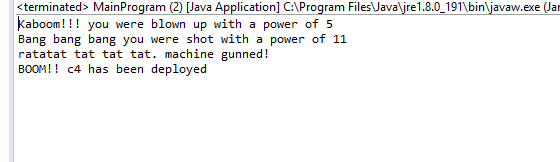
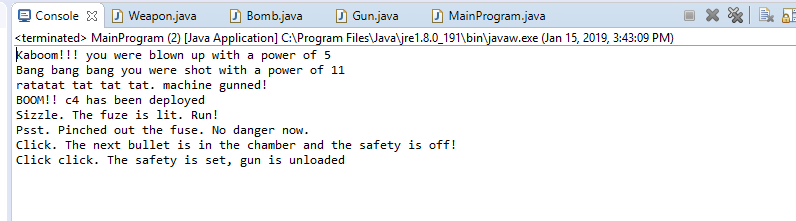
3d. Both sub-classes are being supervised by an abstract class called Weapon. The sub-classes are extending of the weapon class, which means they have the properties of the weapon class. The weapon class has a system output of power and both sub-classes follow it.

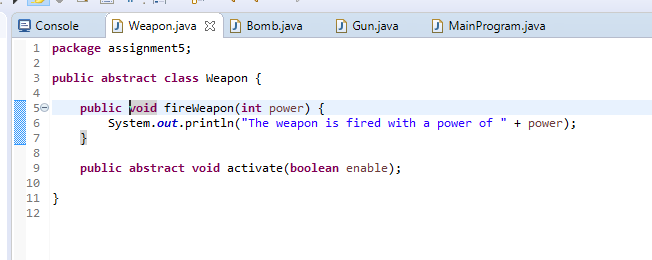
4d. Both sub-classes have been granted an override. An override happens when the same method and variables are called from the sub-classes instead of the abstract class. The gun and bomb class created new system outputs from the same Weapon.fireWeapon() placed in each of their classes.



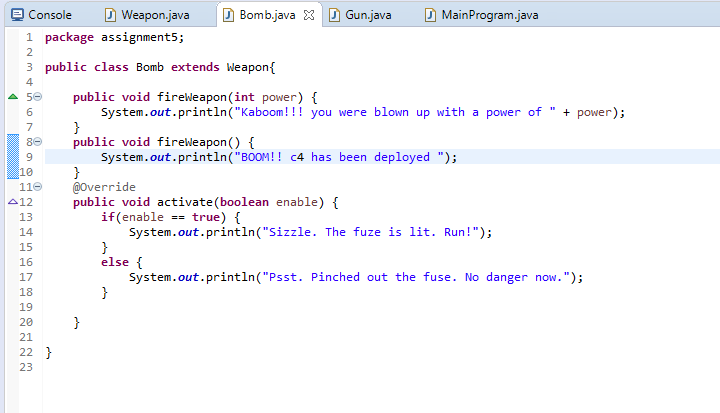
5d. An overload will occur when the same method is called from the classes but without the same signature requirement in the second method. With a variable in the method, the output will show the method with the variable. The same method without a signature required will overload the previous method and will output the new method.

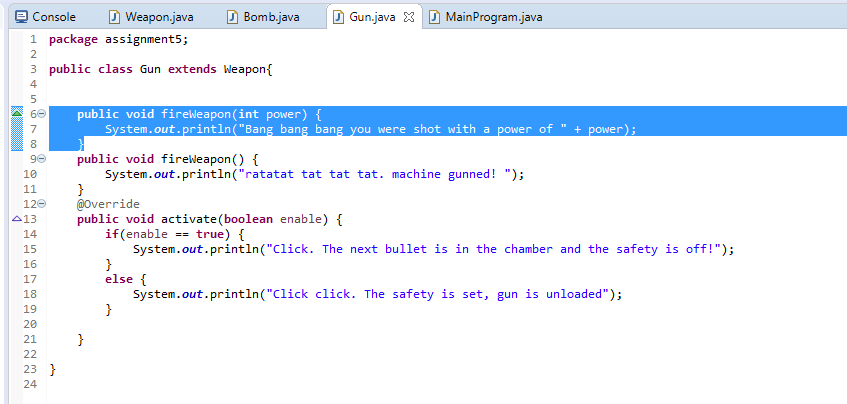


6b.

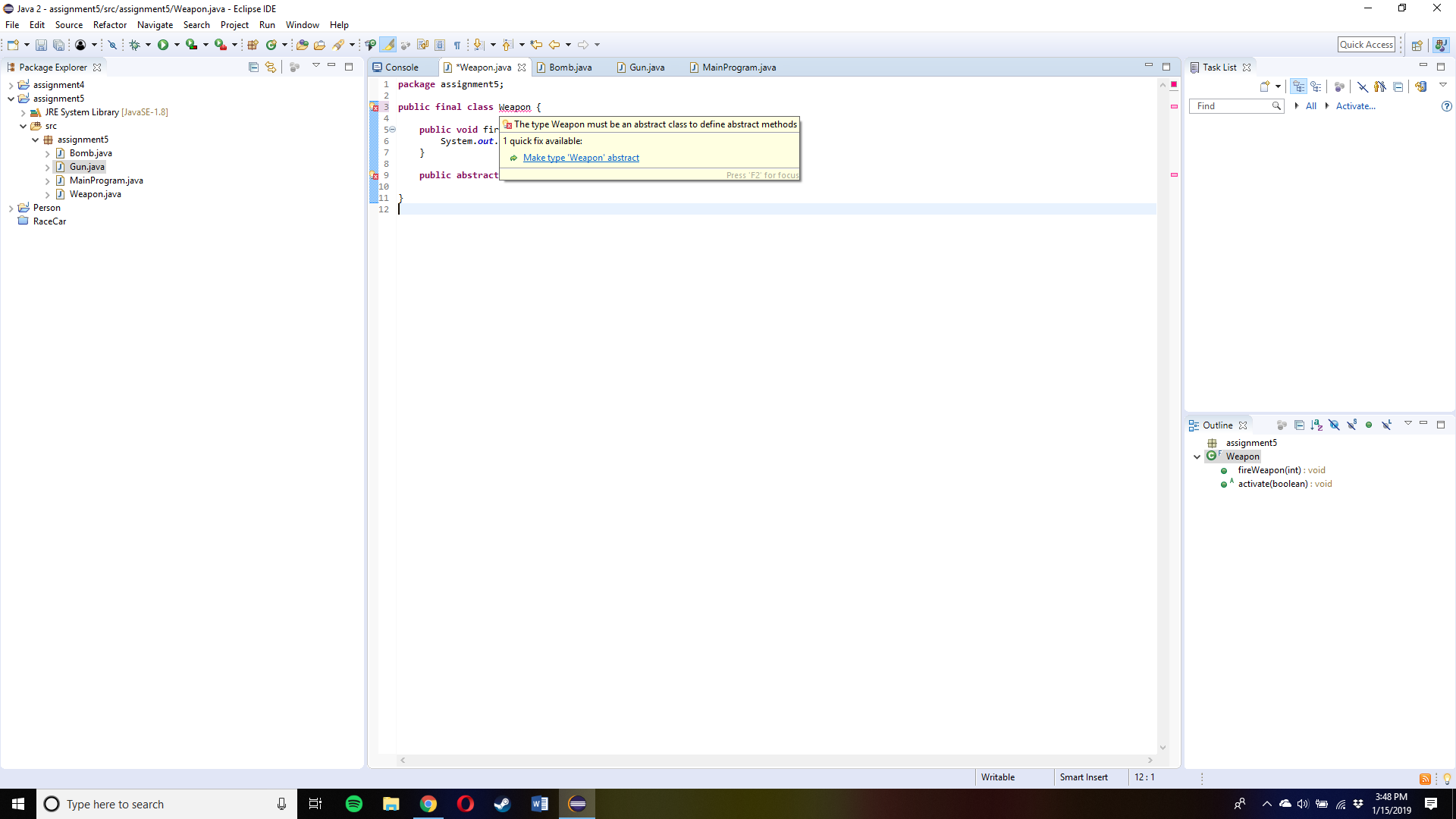


6c1. Implementation of activate() method.

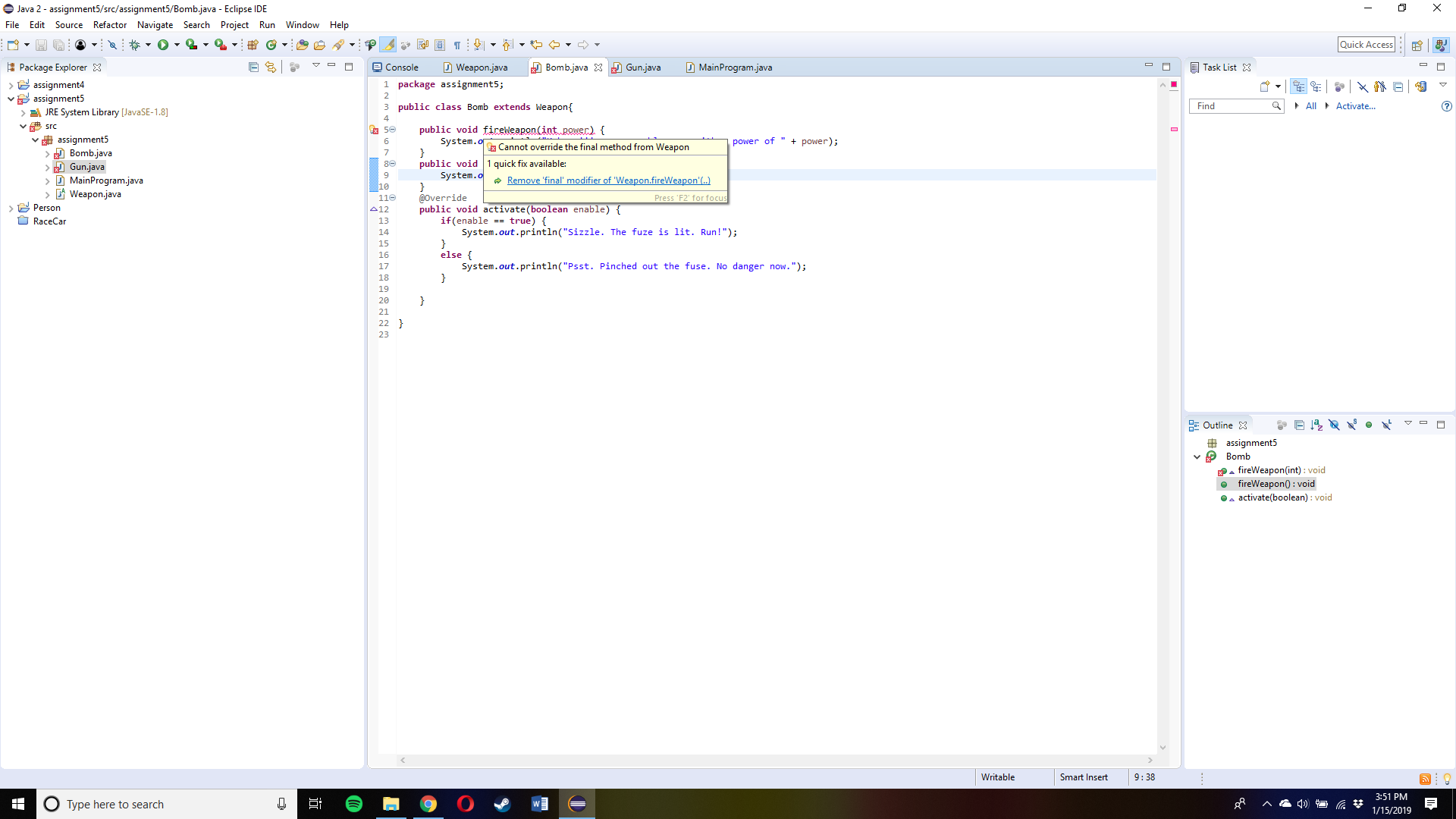
6c2. Implementation code for Bomb class



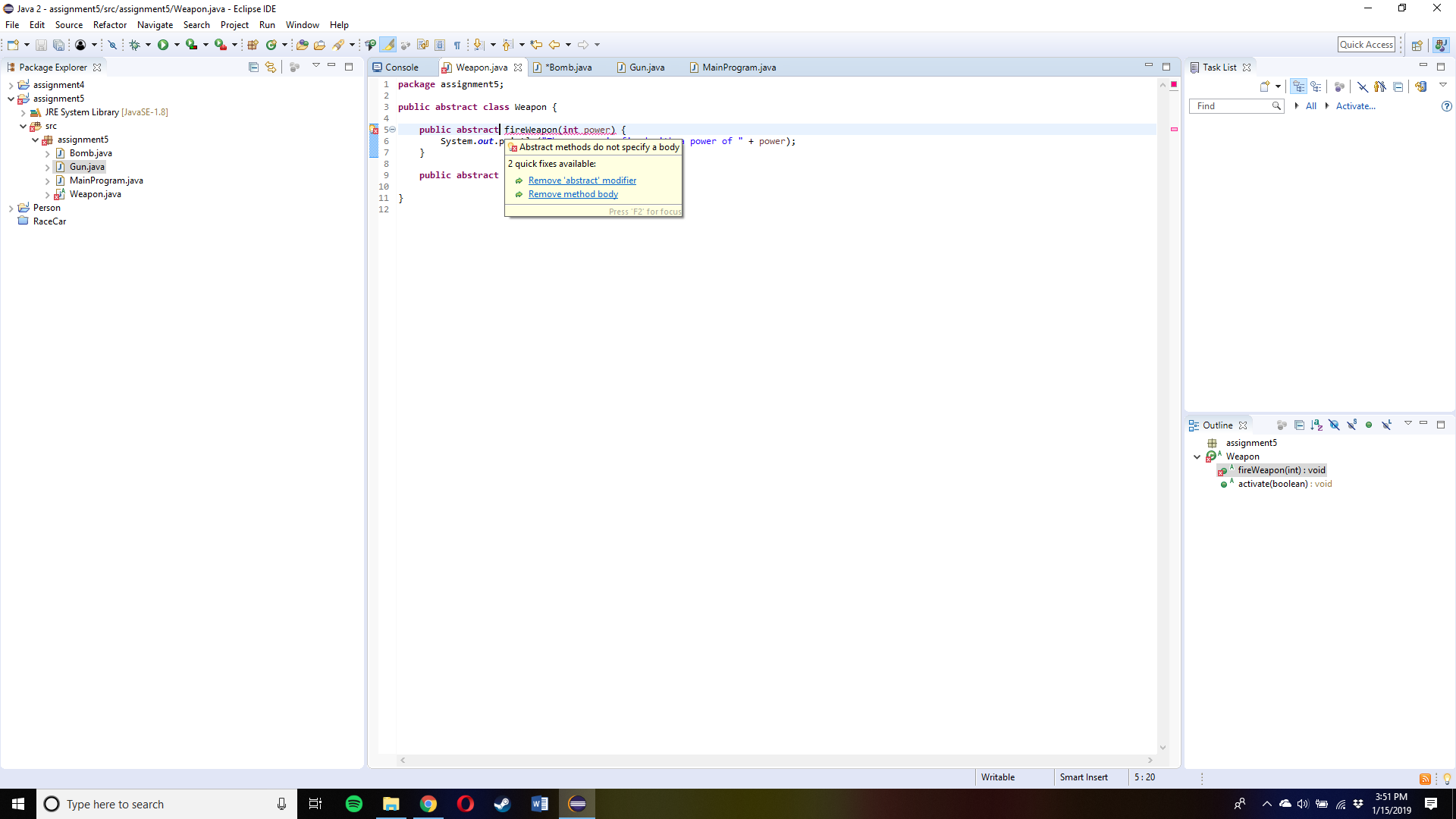
6c3. Implementation code for Gun class



7a. (Zoom in Word to see picture better) After changing the Weapon class from abstract to final, the class can no longer extend itself and it cannot take in abstract methods. The sub-classes could not extend the weapon class and will cause errors. The console would not run anything because every code extends from the Weapon class.



7b. The method fireWeapond(int power) cannot be changed to final because both sub-classes have the method in their class. The sub-classes are extending of the weapon class, the method cannot be final or else the error will occur. The sub-classes need to use the method in order to call the method in the MainProgram class.



7c. The error occurs when there is an abstract method with a body in an abstract class. The abstract class cannot have an abstract method with a body because the sub-classes will provide the implications instead of itself providing the implications.