River Crane

Professor Conrad

CIS 18A

28 January 2020

Software Reuse and Protected versus Private Members

In programming, a technique called inheritance provides useful benefits for all programmers. Inheritance is when a new class is created, but is able to use preexisting members from a class made before it. One of the benefits of inheritance is it promotes software reusability. The first classes created are called superclasses, while the other classes which use the prepared member variables or modify them are called subclasses. Subclasses may even become superclasses as more subclasses are created and reference previous data. Also, the ability to use the same code for multiple classes helps to save time. When creating a large application which may use hundreds or thousands of classes, it will be useful to reference and reuse previously created information. If programmers were to uniquely create every line of code in a program, time would be wasted for the classes which contain similar characteristics. The less a programmer needs to write often means they will encounter fewer errors. When a program becomes increasingly complex, it is easier to make simple mistakes. Making use of code from a class which has already functioned properly limits a large portion of troubleshooting. Creating multiple classes which contain various differences may cause problems when trying to piece them together. Inheritance is possible through the protected member part of a class. The positive aspect of protected members is they allow subclasses to access the protected members of a superclass, as if they were public. A possible downside to protected member variables is they are not entirely private. Just as previously explained, they can still be used by another class if they are referenced. This may be an issue if a programmer created a class and wanted the variables to be inaccessible, yet they wrote them as protected instead of private. This brings to light the benefit of private member variables, because no other class will be able to obtain or modify anything placed within the protected part of a class. All of these practices are part of inheritance, and demonstrates how versatile this strategy to programming can be. When used correctly, inheritance has the ability to promote software reusability, decrease programming time, and help reduce the amount of errors found when creating a program.