Final Project Articulation – Programming with Classes

CSE210 Programming with Classes covered 4 major topics and in doing the Foundation 4 final project I was able to use each of those principles of programming to better develop my programs.

The first principle of CS210 is Abstraction, this is the process of breaking down each program into individual classes that handle only one concept. For simple programs this often does not feel necessary, however, it provides options to continue using code in more flexible ways. An example of this is by Abstracting the Address class in the Foundation 2 assignment, I was able to then reuse that class (minus the IsUSA() method) in the Foundation 3 assignment. Abstraction was used through all four of the foundation programs.

The second principle is Encapsulation. This is the process of protecting or making private the member variables. One part of this method of encapsulation that is available is the process of making Getters and Setters for variables. One thing that I realized about this is that if we indiscriminately make Getters and Setters for every variable, then we are not truly encapsulating the class. Only variables that need to be accessed from other classes need to have Getters and Setters. I went through and initially programmed all my member variables with getters and setters, then after the code was complete, I eliminated any that were not directly needed for the function of the code. Encapsulation was used throughout the four foundation programs.

The third principle of CSE210 is Inheritance. This allows for code to be placed in a parent class and then used by child classes. The principle of inheritance also effects encapsulation as the member variables of encapsulation that are used by children classes need to be “protected” and not “private”. Inheritance is a very powerful part of programming as it allows simpler code and less redundant coding. You don’t have to define code in every class if the only thing that changes is the value of the variables. The principle of Inheritance was used in the Foundation 3 and Foundation 4 program. Foundation 3 is used between the General Event and Child Lecture, Reception, Outdoor, and Sport Events. The Foundation 4 program uses it between the parent Exercise and the Child Running, Stationary Bicycle, and Swimming classes.

The final principle is Polymorphism. This allows for one method to call different code based on the class that it is called from. This is done where the parent class is tagged with the key word “virtual” and the child classes are tagged with the key word “override”. This allows for common ideas or concepts that need to be handled differently based on the object they are contained in. In practice this was done with the idea of Area. Where the area calculation is based on the shape, the GetArea() method can be specific to the object of the shape. In the Exercise Foundation 4 activity, this was addressed in how the GetDistance(), GetSpeed(), GetPace(), methods completed the calculations. Every Exercise event that was included had a Distance, Speed, or Pace associated with them but the formula for calculating each was different. This also allowed for the Inheritance to be used to then have one GetSummary() method that will call the appropriate method based on the type of exercise. This will allow me to add new exercise specific classes in the future and be able to create new calculations but use the same base code.