## Abstraction

Abstraction is the process of making something complicated simple by breaking it up into simpler ideas and only showing the most important parts of the object. I used abstraction in the recent Exercise Tracking Program by splitting up the complex idea of activities into several classes like swimming, running and cycling. It helps make the code more flexible because a class can be added without having to update the rest of the program.

## Encapsulation

Encapsulation is the process of keeping some things in your program private and only showing safe parts of it through public methods. In the same Exercise Tracking Program, I used encapsulation by making all the member variables of the classes private, only showing data through the public methods. Which means the data couldn’t be changed from outside the class. It keeps it flexible for expansion by allowing us to make changes inside the class that won’t affect any other part of the program. It will help keep things more consistent as changes/expansions are made.

## Inheritance

Inheritance is the process of letting a class use methods defined in another class. In the Exercise Tracking Program, I use inheritance by letting the Running, Cycling, and Swimming classes share properties from the Activity class. They all use the date and duration methods from the Activity class. It keeps things flexible by allowing me to use code that I already have in future classes, as well as being able to update the parent class to make a change to the rest of the child classes at once.

## Polymorphism

Polymorphism is the allowing a method to act differently depending on the class that calls it. In the Exercise Tracking Program, I used Polymorphism to allow Running, Swimming, and Cycling to all override the GetDistance() method of the Activity class to work correctly for that type of activity. The same method was called, but the method worked differently in each class. It makes the code more flexible for future changes because it can scale with new activities being added without having to change the other previous classes.