# PPW1 – Task 1.2

To explain my UML diagram: I first need to cover inheritance and how my diagram covers this. Inheritance within object orientated programming happens when classes derive from another class, the child class (or subclass) will inherit the public and protected properties and methods from its masterclass/parent class. Whilst also allowing it to have its own methods and property’s. (in my above UML diagram “Cycling is a session”, cycling inherits the values from its parent class and allows it to have parent properties (for example – having a date and time)) without having to define the information twice: making it much more efficient and less data repetition needs to take place.

Polymorphism is the notion that objects of different types may use the same interface and present different results. Each class may utilize this interface with its own, independent implementation (for example: doing different things based on class properties). Polymorphism will be used within my code to make the code more dynamic and to ensure that it can easily be expanded to the client’s requirements: as this is a proof of concept code piece, this will be handy if the client chooses to expand. Within my UML diagram, I have used an overridden ToString() method, which allows the program to detect at runtime which operation to use when called. This will allow the program to use it’s properties and print them on screen depending on whether it’s a cycling or a running session: just by having an overridden method within the class: called by a foreach loop that is dynamic and allows the class to call it’s own to string method.

Lastly: I will evaluate why Object orientated coding is better than other methods of coding for this specific task and how the various properties of object orientated programming allow object oriented to be the best: by discussing it’s advantages. Firstly, the usage of Classes allows us to better organise our code, so that it is easy to interface with and ensure that the code works dynamically by treating classes as objects (such as a car), it allows us to build onto this and add subclasses and properties: it allows us to easily interact with the class by creating relationships and depending on the class for various purposes, such as utilizing facilities provided by a class. Secondly, Inheritance is a useful concept of object orientated programming, it allows us to abstract the information that repeats in multiple classes and use this to create a masterclass to avoid repeating of information and to simplify the coding process by building one method and having one set of properties accessible by other classes, instead of a confusing maze of 6 classes with the same data. It also allows us to get specific properties and use these to create subclasses, thus allowing us to do operations and store properties based on these more specialized classes: meaning it’s much simpler to keep track of the information and properties. Lastly, it allows us to write polymorphic programs, meaning that we can easily create systems that have overridden methods, which allow for convention within our classes: allowing us to always understand that running a single operation on any type of class that includes the overridden method (typically via a loop) will run the code specific to how that class wants to utilize that interface. This is great as it allows us to not have to remember or know how to interface with 20 different methods, allowing for simplified code.