- Abstraction:
    - Definition: a technique for separating the implementation from functionality and hiding the implementation details
    - Source: https://press.rebus.community/programmingfundamentals/chapter/encapsulation/
    - Example: writing a function called drawSquare() for a class is an abstraction for the steps to the drawing lines of equal length
- Encapsulation
    - Definition: encapsulation is the hiding of the state or values of a structured data object within a class and restricting direct access to object's components
    - Source: https://press.rebus.community/programmingfundamentals/chapter/encapsulation/
    - Example: declaring variables private in a class and creating public methods to access the variables

```
class Encapsulate {

    private String date;

    public String getDate() { return date; }
}
```

- Polymorphism
    - Definition: concept that refers to the ability of an object, function, or variable to take on multiple forms
    - Source: https://www.educative.io/answers/what-is-polymorphism
    - Example: Class objects that have the same parent class each have the function draw() but the function does different things in each class
- Coupling
    - Definition: the degree btw. two classes or function that violates the hiding of information
    - Source: https://www.educative.io/answers/what-are-the-different-types-of-coupling
    - Example: Class A modifies class B, in C++ friend classes can access each one's private variables using the friend keyword

```
class A
{
    int x;
    friend class B;
};
```

```
class B
{
   //friend class
};
```

- Cohesion
    - Definition: how focused the class is and refers to if the class is designed with a well defined purpose in mind
    - Source: https://www.geeksforgeeks.org/cohesion-in-java/
    - Example: high cohesion means class is focused on what it should be doing, each class is well defined and there is a separate class for each job

```
class Name {

}

class StudentID {

}

class YearInSchool {

}

class Major {

}

class Minor {

}

class Display {

}
```

- Identity

- Definition: property of the object that identifies and distinguishes the object from other objects
- Source: https://link.springer.com/referenceworkentry/10.1007/978-0-387-39940-9_1470
- Example: unique integer assigned to each new object (t1,t2,t3) when created to differentiate each of them

```
class triangle
{
...
} t1 ,t2, t3 ;
```