W07 Prove: Principles of Programming with Classes - Articulate

# Abstraction

## Briefly define the principle.

Abstraction is to make complicated processes into simpler ones by reducing them to their basic steps.

## How did you use that principle in one of your programs?

The first time I used Abstraction in this course was when writing the Entry class in the Journal program. An Entry was reduced to its basic elements: Date, Prompt, and Entry Text.

## How did using that principle help that program become more flexible for future changes?

The Entry process has been broken down to its core elements and it can be accessed by new Journal classes if needed in the future.

# Encapsulation

## Briefly define the principle.

Encapsulation is when attributes and methods get protected and not exposed except when necessary.

## How did you use that principle in one of your programs?

I used Encapsulation when writing the Scripture program. It made sense to me to not use getters and setters just for the sake of it.

## How did using that principle help that program become more flexible for future changes?

Even though the variables on the Scripture classes were set to private to protect the structure of the program, getters and/or setters can be written to access or set the values of the corresponding attributes.

# Inheritance

## Briefly define the principle.

Inheritance is when a child class gets the properties (attributes and methods) of the parent class.

## How did you use that principle in one of your programs?

I was able to use Inheritance when writing the Activity program by defining a parent class and some child classes that inherited attributes and methods from the parent class.

## How did using that principle help that program become more flexible for future changes?

By having a parent class from where the child classes can inherit attributes and methods it makes the program so much more flexible for future changes since we can make changes to the methods in the parent class and not have to change the children classes that inherit those methods.

# Polymorphism

## Briefly define the principle.

Polymorphism is when we can change the behavior of methods in child classes from parent defined methods.

## How did you use that principle in one of your programs?

I used the Polymorphism principle in the Goals, by overriding the GetStringRepresentation parent method, when writing the unique behavior in each one of the child classes.

## How did using that principle help that program become more flexible for future changes?

Polymorphism makes everything very flexible. If changes are needed in the future one can write changes in the methods overrides and not have to change other areas of the program, unless specifically needed.