CSE210 Team Adelie

Brother Christopher Pitt

Scott Kamerath

3/15/23

**Polymorphism Articulation**

Polymorphism has been an interesting concept of C# for me to learn. The best explanation that I can give is that it’s the capability in C# to have the objects, getters, setters, variables and methods able to interface and communicate with each variable and or method. I was intrigued by how the effects of late binding work because the capability in the code of polymorphism requires the use of reflection and it almost reminds me of the concept of inheritance. The unique thing here is that the behavior of the object or code in question. This is important that it all runs together at the run time and that each interface is smooth and consistent. This will all work and run at the same time which provides the aspect of polymorphism which in its basic form means that it will run in “many forms.”

// The Parent class needs to show a "virtual" keyword or similar.

public abstract class Employee

{

  private string \_employeeName;

  private string \_employeeID;

  // We need to realize the abstract method doesn't have to have body at all (not even an empty one),

  // and it is always followed by a semicolon. ;....

  public abstract float CalculatePay();

}

// This is the child class.

public class SalaryEmployee : Employee

{

  private float salary = 100f; // 200f etc.

  // The salary can have the dollar amount float to almost any dollar value and decimal point.

  public override float CalculatePay()

  {

    return salary;

  }

}

// This is the child class.

public class HourlyEmployee : Employee

{

  private float rate = 9f; // 8f etc.

  private float hours = 100f; // 200f etc.

  public override float CalculatePay()

  {

    return rate \* hours; // The pay is calculated differently here.

  }

}