Polymorphism

Polymorphism

- Polymorphism enables you to “program in the *general*” rather than “program in the *specific*.”
- In particular, polymorphism enables you to write programs that process objects which share the same superclass, either directly or indirectly, as if they were all objects of the superclass; this can simplify programming

Example

- Suppose we create a program that simulates the movement of several types of animals for a biological study.
- Classes Fish , Frog and Bird represent the types of animals.
- We create a generic class “Animal” that covers most activities as its methods which are common among these animals

  (fish, frog, Bird)
  - move()
  - rest()
- Later we make individual classes of Fish, Frog & Bird extending from Animals using keyword “extends”.
  - They will receive Animal class`s methods in inheritance
  - We can also change logic of those methods to meet specific requirements.
  - We would add additional methods to cover all unique capabilities.

- Animal Class
- Move() walk /jumps/hogs
- Rest()

- Fish Class extends Animal // Inheritance
- Move () Overridden swims
- Rest()

- Frog Class extends Animal
- Move () Overridden jumps
- Rest()

- Bird Class extends Animal
- Move () Overridden flys
- Rest()

Contd.,

- Each specific type of Animal responds to a “move()” method in its own way
  - A Fish would swim
  - A Frog would jump
  - A Bird might walk or fly to move.
- Each object modify its *x*-*y* coordinates appropriately for its *specific* type of movement.
- Relying on each object to know how to “do the right thing” (i.e., do what’s appropriate for that type of object) in response to the *same* method called, is the key concept of polymorphism.
- The *same* method (in this case, move) sent to a *variety* of objects has *many forms* of results—hence the term polymorphism.

Example 2

- ***Quadrilaterals***
- If class Rectangle is derived from class Quadrilateral, then a Rectangle object *is a* more *specific* version of a Quadrilateral. Any operation (e.g., calculating the perimeter or the area) that can be performed on a Quadrilateral can also be performed on a Rectangle.
- These operations can also be performed on other Quadrilaterals, such as Squares, Parallelograms and Trapezoids. The polymorphism occurs when a program invokes a method through a superclass Quadrilateral variable—at execution time, the correct subclass version of the method is called, based on the type of the reference stored in the superclass variable.

Practice

- In your Lab 2 create SavingsAccount class, with the following instance variables and methods:
- noOfTransaction (int) , transactionFees (double),interest (double)
- SavingsAccount extend Account.
- Overriding the methods:

deposit()

withdraw()

This practice is not be included in Lab2 while submitting

- Savings account are usually free accounts.

1. Online transfers are free
2. If you w/d money from savings a/c ---- transaction fees apply
3. Transaction fees is based on no of transactions
4. $2/ transaction