River Kelly
ESOF 322
Homework 1
09/07/2021

Exercise 1 (4 pts)

What is polymorphism?

Object-oriented programming enabled the construction of specific routines to handle different variables, methods, and objects at different times. This is polymorphism. To better remember polymorphism, it is helpful to know the definition roots.

“The word poly means "many" and the word morph means "form", so when we talk about a polymorphism we're talking about something that appears in many different forms.”
(“What is polymorphism in computer science?”)

According to *ComputerHope.com*, there are three primary types of polymorphism:
- **Ad hoc polymorphism**: when a function takes on different forms
- **Parametric polymorphism**: code is written without specification using abstraction
- **Inclusion polymorphism**: when a single name refers to multiple different class types that share the same superclass, also know as subtyping

(Hope)

Let’s take a look at the first type of polymorphism, Ad hoc polymorphism, where a function can take on multiple different implementations. In Java, this is known as method overloading. It is done by having two or more methods with the same name but each accepts a different number of variables.

```
public class Calculator {

  // returns sum of 2 integers
  public int add(int a, int b) {
    return a + b;
  }

  // returns sum of 3 integers
  public int add(int a, int b, int c) {
    return a + b + c;
  }

  // returns sum of 4 integers
  public int add(int a, int b, int c, int d) {
    return a + b + c + d;
  }
}
```

In the code above, we have a class named Calculator. This specific calculator has the ability to calculate the sum of either 2, 3, or 4 integers, it is very rudimentary. The Calculator class has three methods, all of which have the same name; ***sum***. They differentiate from each other by the number of parameters passed to each method. Therefore, the specific method invoked depends on the number of parameters being passed. Looking at the internal working of the last two methods, you will notice that instead of simply summing the input parameters, they take advantage of the previous `sum` methods, and simply add the value of the last parameter.

What is inheritance?

Describe the differences between public vs private inheritance

When describing a data model, there are often times when multiple different components are closely related to one another. When this happens, it is beneficial to group these objects in parent-child relationships. Where the parent object describes the greater common characteristics and the individual children more uniquely describe themselves. The properties of the parent objects are "inherited" by each of the children. This is inheritance.

Public inheritance is when inherited properties are visible to outside members, private inherited properties are when the properties are only visible to the sub class.

What is encapsulation?

Encapsulation is the practice of grouping together related data and operations which impact such data. This would imply that all of the necessary "knowledge" for a specific mechanism is located in the same area, or "encapsulated" together. Encapsulation can also refer to limiting the available "knowledge" of a specific mechanism.

I am currently enrolled in 466-Networks, and we recently learned about the layered architecture of protocols that make up the internet. The responsibilities of each layer are encapsulated, creating defined dependencies within each layer.

What is the difference between static vs dynamic binding?

Last semester, I was enrolled in CSCI-305 Concepts/Programming Languages, and I would like to refer back to my notes and textbook from this class to provide some formal definitions.

"A **binding** is an association between an attribute and an entity, such as between a variable and its type or value, or between an operation and a symbol. The time at which a binding takes place is called **binding time**." (Sebesta 227)

Now, let's define static and dynamic binding:

- A binding is *static* if it first occurs before run time and remains unchanged throughout program execution.
- A binding is *dynamic* if it first occurs during execution or can change during the execution of the program.

The primary difference between the two, static and dynamic binding, is at what time the binding takes place, either before or during run time.

Exercise 2 (8 pts)

Provide a definition for a prescribed lifecycle, and name two types of prescribed lifecycles

A prescribed lifecycle is a process the models advocate for an orderly and structured approach to each step in the software development process. Two different models are the waterfall and incremental models.

For each type provide an example (i.e. a use case)

The waterfall model is used for a fixed system, such that each phase is completed to 100% before moving on to the next.

The incremental model is used in order to stay in contact with the customer, keeping them informed of each step as the project move throughout its lifecycle.

Research Question (3 pts)

What do low coupling and high cohesion mean?

Low coupling means that individual modules have a low dependency on each other. High cohesion means that each module is concentrated in its own mechanisms.

Why is this important? Let’s say for example you have two classes. These two classes depend on the existence of one another. The question becomes, how much information should each of these two classes know about the internal workings of the other? The correct answer is dependent on the specifics of the problem, but ideally, less information is better. The reason for this is because the result will be much easier to maintain and less prone to problems when changes are made.

For example, what if class A depends on class B. The successful execution of a routine or method of class A must depend on some assistance from class B. If class A knows more information about class B than it should, and heavily depends on the inner workings of class B, then you would not be able to suggest that class A is “encapsulated” to itself. Now let’s say that some changes need to be made to class B. These changes may drastically affect the dependence class A has on class B.

Bibliography

Hope, Computer. “Polymorphism.” *Computer Hope*, 06 31 2019, https://www.computerhope.com/jargon/p/polymorphism.htm. Accessed 05 09 0221.

Sebesta, Robert W. *Concepts of Programming Languages*. vol. 11, Pearson, 2019.

“What is polymorphism in computer science?” *Sumo Logic*, https://www.sumologic.com/glossary/polymorphism/. Accessed 5 September 2021.