Alexander J King

Comparative Programming Languages  
Object Oriented Programming

4.22.20

Using Chapter 12 slides 25-51

Explain how C++, Objective-C, C#, and Java each support the 3 features of OOP

3 Features of oop in general:

ADTs: are usually called classes   
inheritance: Inheritance allows new classes defined in terms of existing ones, i.e., by allowing them to inherit common parts . This is especially advantageous when I get to reutilize code. For example for my final project I’m using hava and sql databases. The functions where I am able to add additional students, classes, professors, or assignments are quite similar and students and profs are almost identical.   
dynamic binding: A polymorphic variable can be defined in a class that is able to reference (or point to) objects of the class and objects of any of its descendants • When a class hierarchy includes classes that override methods and such methods are called through a polymorphic variable, the binding to the correct method will be dynamic • Allows software systems to be more easily extended during both development and maintenance

C++:  
ADTs: A class need not be the subclass of any class  
Inheritance: Access control for members are private, public, and protected. Multiple inheritance is supported. If there are two inherited members with the same name they can both be refrenced using the scope resolution operator.

Dynamic Binding: A method can be defined to be virtual, which means that they can be called through polymorphic variables and dynamically bound to the messages. A pure virtual function has no definition at all and a class that has at least one pure virtual function is an abstract class.

Objective C:

Largest syntactic difference between c++ and objective c: method calls

ADTs: Single inheritance only.   
Inheritance: Every class must have a parent.– A category is a secondary interface of a class that contains declarations of methods (no instance variables category is a mixin – its methods are added to the parent class The other way to extend a class: protocols – A protocol is a list of method declarations   
Dynamic Binding: Different from other OOP languages – a polymorphic variable is of type id. An id type variable can reference any object. The run-time system keeps track of the type of the object that an id type variable references If a call to a method is made through an id type variable, the binding to the method is dynamic

Java: Because of its close relationship to C++, focus is on the differences from that language

ADTs: All data are objects except the primitive types – All primitive types have wrapper classes that store one data value – All objects are heap-dynamic, are referenced through reference variables, and most are allocated with new – A finalize method is implicitly called when the garbage collector is about to reclaim the storage occupied by the object   
Inheritance: Single inheritance supported only, but there is an abstract class category that provides some of the benefits of multiple inheritance (interface) – An interface can include only method declarations and named constants, e.g., Methods can be final (cannot be overriden)   
Dynamic Binding:– In Java, all messages are dynamically bound to methods, unless the method is final (i.e., it cannot be overriden, therefore dynamic binding serves no purpose) – Static binding is also used if the methods is static or private both of which disallow overriding

C#:Support for OOP similar to Java   
ADTs– Includes both classes and structs Classes are similar to Java’s classes   
Inheritance: structs are less powerful stack-dynamic constructs (e.g., no inheritance) Uses the syntax of C++ for defining classes – A method inherited from parent class can be replaced in the derived class by marking its definition with new – The parent class version can still be called explicitly with the prefix

Dynamic Binding: To allow dynamic binding of method calls to methods: • The base class method is marked virtual • The corresponding methods in derived classes are marked override – Abstract methods are marked abstract and must be implemented in all subclasses – All C# classes are ultimately derived from a single root class, Object