Homework 4: Polymorphism

Jack Shi - A92122910
Wyatt Guidry - A12994977
Austin Coleman - A12888539

Feb 23, 2018

1 [12pts] Smalltalk Implementation Decisions

1. **Answer:** It is an space efficiency optimization. /tbc
2. **Answer:** Advantage would be fast lookup time because this elimates dereferencing and finding definiton from super classes. Disadvantage would be slow compile time since all the subclasses that have many super classes will need to copy all the super class method dictionaries.
3. **Answer:** Assuming instance variable is accessed very often at run-time, this approach can speed up methods that reference instance variables. The disadvantage would be memory requirement since in case a class has many super classes and all the variable names from super classes must be added to the class template of the subclasses.

2 [30pts] Contravariant Method Specialization

1. **Answer:** Without contravariant method specialization, the `update` method is from `class A`, because the class pointer points to `A`. With contravariant method specialization, a method that accept a more general argument could be used in place of one that uses a more specific argument. This means that `update(Circle& c)` can be replaced with `update(Shape& s)`.
2. **Answer:** The compiler should not accept `class B` as a valid subclass of `class A` because with contravariant method, the more general `update` method of `Class B` will be used in place of the `update` method of `class A`. This is bad because now instances of `Class B` will loose access to `update` method that is specific for `Circle`. The user may want to update `Circle` in a specific way but it will be treated like `Shape` instead.
3. **Answer:** Under current version of C++, the code would invoke `update` from `Class A`. This means it will print the following:
   `Circle radius: 1`
   With this method specialization, the code would invoke `update` from `Class A`:
   `Circle radius: 7`
4. **Answer:** The users expect function overloading. When they pass in a parameter of more specific type, they would like it to be handled by the more specific method. (If they wanted the opposite, they would just down cast bruh ... lol UPenn scrub). By implementing the contravariant method specialization, we are simply loosing interfaces and implementations for more specific types.

3 [10pts]Function Subtyping

Answer:

$$C \mapsto D <: A \mapsto B \Rightarrow A <: C \wedge D <: B$$