CPSC 210 Type Hiearachy/Polymorphism/Dispatch

Lecture/Lab

Use the SPACEINVADERSREFACTORED code to answer these questions unless otherwise directed by a question.

a) Sketch the type hierarchies for *all* the classes in the `ca.ubc.cpsc210.spaceinvaders.model` package

b) Does the type hierarchy you sketched in part (a) include any abstract classes? What functionality is defined in the abstract classes which is common to every subclass of that abstract class?

c) Can you find an example of two methods that override an (inherited) implementation? For each method, does the method extend or replace the inherited implementation? Why?

d) Imagine that you have been asked to add the ability for the user to fire either a skinny missile or a wide missile. Instead of using the space key to fire a missile, the user will hit the "s" key to fire a skinny missile and the "w" key to fire a wide missile. Would you add any class(es) to the class hierarchy? If so, what would they be? Draw where the class(es) would be placed in the class hierarchy? [Don't worry about changes to handling the different key presses.]

e) If you decided to add class(es) for question (d), would you need to override any methods in those classes? List any methods you would override and be able to explain why.

f) Draw the call graph for `GamePanel.drawGame(Graphics g)` in SPACEINVADERSREFACTORED. Compare it to the call graph for `GamePanel.drawGame(Graphics g)` from SPACEINVADERSBASE. Which version do you think is better and why?

g) Look at `SIGame.moveSprites()`. Imagine you have a breakpoint set in this method and execution pauses at the breakpoint after firing two missiles and with one invader appearing on the screen. How many sprite objects would be in the list referred to by the `sprites` field of `SIGame`? What is the apparent type of the `next` variable in the `SIGame.moveSprites()` method? What is the actual type of the object referred to by the `next` variable each time through the loop (with two missiles and one invader appearing on the screen)? [Note, you could set a breakpoint – try on the 200th execution of the method or so – and step through the method to determine this information.] Each time through the loop, which `move` method is being called?