1. @Overide annotation

File Name	Class & Method	Line
Monster.java	Monster.ping(Model)	15
Monster.java	Monster.draw(Graphics, Point, Dimension)	31
Keys.java	Keys.keyTyped(KeyEvent)	19
Keys.java	Keys.keyPressed(KeyEvent)	21
Keys.java	Keys.keyReleased(KeyEvent)	26
Sword.java	Sword.location()	14
Sword.java	Sword.ping(Model)	26
Sword.java	Sword.draw(Graphics, Point, Dimension)	36

2. Consider Direction, Controller, Camera and Sword
    a. What is the name of the design pattern encoded by those classes?

       State pattern

    b. Explain in the detail the meaning of
       setAction(KeyEvent.VK_W, c.set(Direction::up), c.set(Direction::unUp))

       In the event that the W key is pressed the camera object calls the enum Direction up, moving the camera up, and then sets to unUp once the key is let go

3. Explain the behavior of Cells.forAll.
    a. How is Cells.forAll currently used?

       It is checking through all cells on the screen to a given range and adding them to a consumer called action, which calls another function get, which by the looks of things is in charge of placing the cobblestone texture.

    b. Can you describe some features that we could add to our game that would take advantage of this method?

       Perhaps if we wanted to create new biomes you could give it a new texture to place ad tell it place from x to y

4. In Compact, the concept of changing phase encodes a programming pattern.
   a. What is this programming pattern?

      Observer pattern

   b. How is this implemented? What classes serve what roles in that programming pattern?

      Subject -
      Observer -

5. Explain why using *SwingUtilities.invokeLater(Compact::new)* is needed and why *assert SwingUtilities.isEventDispatchThread();* is a good check to have in this setting.

   .invokeLater() can be used to perform tasks asynchronously in the Abstract Window Toolkit (AWT) Event Dispatcher Thread (EDT), which is a background thread dedicated to processing events from the GUI event queue. Because this is an asynchronous action you need to check if the code you are running is on the correct thread otherwise it will start throwing all sorts of concurrency eros, hence the .isEventDispatchThread() method is called.

6. Explain why the provided code was forced to use *addWindowListener(new WindowAdapter(){..})* instead of a Lambda.

   Well a window listener is used to implement custom window behavior and because it's got things running on multiple threads the game doesn't actually close when the window is closed. So when the game is exited it executes the command close() and shuts down the running threads.

7. Currently there is a bug: if you press the 'up' key the hero keeps going up even after the key has been released.
   a. Explain what is happening with this bug.

      When ever the up key is pressed the player will continually move upwards until you press another key

   b. Fix the bug changing a single character. Report changed code and the line number here.

      The bug was a typo. I changed line 8 in Direction.java from unUn() to unUp() as it wasn't canceling the up movement unlike the rest of the move directions

c. What features could help to avoid this and similar bugs?

Perhaps create an interface just with all the direction types so that whenever I type one incorrectly it gives me an error rather than an unused method error.