Given below are the world functions.

- boolean logic:
  - not a
  - both a and b
  - either a or b, or both

- math:
  - a = b
  - a ≠ b
  - a > b
  - a ≥ b
  - a < b
  - a ≤ b

- random:
  - choose true with probability 0.5 of the time
  - random number

- string:
  - a joined with b
  - what a is as a string

- ask user:
  - ask user for a string
  - ask user for a number
  - ask user for yes or no

- advanced math:
  - minimum of a and b
  - maximum of a and b
  - absolute value of a

- mouse:
  - mouse distance from left edge
  - mouse distance from top edge

- time:
  - time elapsed
  - year
  - month of year
  - day of week
  - day of year
  - day of month
  - day of week in month
  - is AM
  - is PM
  - hour of AM or PM
  - hour of day
  - minute of hour
  - second of minute

- arcsin a
- arcos a
- arctan a
- arctan2 a b
- a raised to the b power
- natural log of a
- a raised to the a power
- IEEE remainder of a / b
- round a
- a converted from radians to degrees
- a converted from degrees to radians
- the b th root of a
- floor a
- ceiling a
- sin a
- cos a
- tan a
- other
  - right, up, forward

Tiles at the bottom of the Alice window.

Events:

- create new event
- When the world starts
- When a key is typed
- When the mouse is clicked on something
- While something is true
- When a variable changes
  - Let the mouse move <objects>
  - Let the arrow keys move <subject>
  - Let the mouse move the camera
  - Let the mouse orient the camera

Not shown: When something is true
Format of instructions:

Chicken properties and methods:
Given below are the chicken functions.

Array Visualization special methods and functions
1. (10 pts) Consider the following snapshot of an Alice world in which there are three objects which are (from left to right) shark, seaweed, and goldfish. The shark and seaweed are 3 meters apart. The seaweed and goldfish are 3 meters apart. Imagine the three of them are standing on an invisible line.

A) List the events in this world.

B) List the event handlers in this world.

C) Explain what happens in the world when the Play button is pressed.
2. (4 pts) Consider the following array of animals shown below as a picture and as a property, and the following code. Also shown in the picture are an ArrayVisualization and an ObjectVisualization.

Here is the code:

```
let ObjectVisualization = the value at ArrayVisualization [ index ]
let ArrayVisualization [ index ] = the value at ArrayVisualization [ ( 5 - index ) ]
let ArrayVisualization [ ( 5 - index ) ] = the value of ObjectVisualization
```

Remember that when an object is printed, its name is printed.
What output is printed when this code is executed?
3. (7 pts) Consider the following function called `world.mystery`. Assume a list called `animals` has been created and filled with objects.

```
world.mystery someObject
```

A. What type of value does the function `world.mystery` return?

B. What is `someObject`: a local variable, a class variable or a parameter?

C. Explain what the function `world.mystery` does.

D. (8 pts) Consider the following Alice code.

```
Loop 5 times times show complicated version
cheerleader turn left 1 revolution more...
Loop 3 times times show complicated version
cheerleader say Go Duke! more...
cheerleader turn left 1 revolution more...
Loop 4 times times show complicated version
cheerleader say Let's Win! more...
cheerleader turn left 1 revolution more...
```
A. How many times does the cheerleader say “Go Duke?” when this code executes?
B. How many times does the cheerleader say “Let’s Win” when this code executes?
C. How many times does the cheerleader turn around when this code executes?
D. Explain how Prof. Rodger could have gotten the word Duke on the cheerleader’s shirt.

5. (7 pts) Consider the following recursive function world.doSomething whose lines are numbered.

A. Which line of code has recursion?
B. Explain where the “way out” of the recursion is.
C. What is the output of the following code?
6. (10 pts) Consider the following Alice world, which includes some code and a picture of the program running. There is no code in world.myfirstmethod.

A. In this world, the user intended to have a game in which you click on the biplane or the boeing707 (other code not shown moves them around randomly) and get the specified points each time you click on them. However, this world does not work as intended, the score is always displayed as 0.0. Explain why this happens and how to fix it.

B. Consider rewriting score.increment in the following way.
Rewrite `score.increment` so that it randomly generates an integer from 1 to the amount given in the parameter and adds this amount to the score. You should also display this random integer to the left of the score. For example, in the figure above, the score was 6.0 before the `boeing707` was clicked on. When the `boeing707` was clicked on the number 3 was generated (a random integer from 1 to 6). This random number 3.0 is shown as the number on the left, and the sum of the 3.0 and old score 6.0 is shown as the new score 9.0 on the right above.

Give the new code for `score.increment` and explain any other changes that need to be done in the world to make these changes.

7. (8 pts) Consider the following Alice world with (from left to right) a dragon, a princess (far in the background) and a Knight sitting on a horse (he is glued to the horse). There is a race to see if the dragon or the knight can get to the princess first. The dragon moves forward by calling `dragon.fly`, which moves the dragon forward a short distance while flapping his wings (he stays on the ground for this). The horse (with the knight riding on the horse) moves forward by calling `horse.gallop`, which moves the horse forward a small amount.
Give the Alice code that does the following. The knight (on the horse) and the dragon face the princess and then repeatedly move forward by calling the methods given above (horse.gallop and dragon.fly). When one of them is less than a meter from the princess they both stop moving. At that time, if the knight is closest, the princess says “Thanks for saving me!” If the dragon is closest, the princess says “The Dragon has got me!”

8. (18 pts) Assume all the penguins below are in a list called penguins. Note that some of the penguins are green in color (look shaded), the rest are white. In addition there is a lighthouse object, which is actually a space ship from another planet, desiring to kidnap green penguins.

A. Write the function world.GetColoredPenguin with two parameters: selectedColor and distance. This function returns a penguin from the list penguins if the penguin has the color selectedColor and is close to the lighthouse, within the given distance. If there is no such penguin, then the lighthouse is returned.

B. Write the method world.takePenguins that has two parameters range and desiredColor. This method has all the penguins from the list penguins that have the desired color and are within range of the outhouse do the following. They should move to the center of the lighthouse (if they do so, they will fit inside of it and will not be shown). Then the lighthouse should take off straight up with the penguins it has collected. For full credit you must call the function world.getColoredPenguin you wrote in Part A.

The picture above shows the result after the call to takePenguins with desiredColor set to green and distance set to 6. All but one of the green penguins was taken. Complete the code below.
9. (18 pts) Consider the array visualization of people shown below.

A) Write the function `World.positionShorterThanIndex` that returns the position of the first item in the array that is shorter than the index position it stands on. If none of the objects are shorter than the position number, then return -1. For example, in the array above, 2 is returned because the Eskimo in slot 0 is taller than 0, the coach in slot 1 is taller than 1 and the teacher in slot 2 is less than 2 in height. Your code should work for any array, and any objects in the array.

B) Write the method `world.resizeTilNoMore` that repeatedly calls the function `world.positionShorterThanIndex` that you wrote in part A. Each time it finds an object in the array that is shorter than the index position it stands on, it doubles the object’s height. This method repeatedly calls the function until no more objects can be resized. It then prints the number of resizes that were needed. For the example on the previous page, the method prints 6 after resizing objects. The teacher and toySoldier were both resized once and the ballerina and magician were resized twice for a total of 6 resizes. The picture above shows the array after the method finishes.