Given below are the world functions.

Tiles at the bottom of the Alice window.

Events:

- 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:

- If `true`:
  - Do Nothing
- Else:
  - Do Nothing

- Loop `5` times:
  - Do Nothing

- Loop `[2]` index from `0` to (but not including) `5` times:
  - Incrementing by `1`

- While `true`:
  - Do Nothing

- For all `world.horses`:
  - Item from `world.horses` at a time
  - Do Nothing

- For all `world.horses`:
  - Item from `world.horses` together
  - Do Nothing

Chicken properties and methods:

- **Chicken's details**
  - properties
  - methods
  - functions

- create new variable
- capture pose
- color
- opacity
- vehicle
- skin texture
- filling style
- point of view
- is showing

- **Chicken's details**
  - properties
  - methods
  - functions

- create new method
- move
- turn
- roll
- resize
- say
- think
- play sound
- move to
- move toward
- move away from
- orient to
- turn to face
- point at
- set point of view to
- set pose

- **Chicken's details**
  - properties
  - methods
  - functions

- stand up
- move at speed
- turn at speed
- roll at speed
- constrain to face
- constrain to point at
Given below are the chicken functions.

```
<table>
<thead>
<tr>
<th>properties</th>
<th>methods</th>
<th>functions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>create new function</td>
<td></td>
</tr>
</tbody>
</table>
```

- **proximity**
  - Chicken is within threshold of object
  - Chicken is at least threshold away from object
  - Chicken distance to
  - Chicken distance to the left of
  - Chicken distance to the right of
  - Chicken distance above
  - Chicken distance below
  - Chicken distance in front of
  - Chicken distance behind

- **size**
  - Chicken's width
  - Chicken's height
  - Chicken's depth
  - Chicken is smaller than
  - Chicken is larger than
  - Chicken is narrower than
  - Chicken is wider than

```
<table>
<thead>
<tr>
<th>properties</th>
<th>methods</th>
<th>functions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>create new function</td>
<td></td>
</tr>
</tbody>
</table>
```

- **spatial relation**
  - Chicken is shorter than
  - Chicken is taller than

- **point of view**
  - Chicken's point of view
  - Chicken's position
  - Chicken's quaternion

- **other**
  - Chicken's current pose
  - Chicken's part named key
  - Chicken's variable named variableName of type valueClass

Array Visualization special methods and functions

```
<table>
<thead>
<tr>
<th>properties</th>
<th>methods</th>
<th>functions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>create new method</td>
<td></td>
</tr>
</tbody>
</table>
```

ArrayVisualization's details

```
let ArrayVisualization [ index ] = item
```

ArrayVisualization's details

```
array

the value at ArrayVisualization [ index ]

ArrayVisualization's size
```
1. (10 pts) Consider the following snapshot of an Alice world in which there are three objects which are (from left to right) eskimo, pj, and pinkBallerina. The Eskimo and pj are 3 meters apart. The pinkBallerina and pj 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.
D) Explain what happens if “S” is pressed 20 seconds after the Play button is pressed.

2. (4 pts) Consider the following code and array of animals shown below as a picture and as a property. Also shown are an ArrayVisualization and an 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 an `ArrayVisualization` has been created and filled with objects.

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

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

C. Explain what the function `world.mystery` does.
4. (8 pts) Consider the following Alice code.

A. How many times does the Kangaroo say “What’s up?” when this code executes?

B. How many times does the horse say “Gotta Go” when this code executes?

C. How many meters does the kangaroo move when this code executes?

D. How many meters does the horse move when this code executes?
5. (7 pts) Consider the following recursive function `world.mystery` 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.

A. In this world, the user intended to have a game in which you click on the cow or the chicken (other code not shown moves them around randomly) and get 5 points each time you click on one of them. When 100 points are reached the cow says “you win.” However, this world does not work as intended. Explain what happens when the world runs and you click on objects.
B. Here is the function `score.increment` again.

Rewrite below the body of this method so that if you click on the chicken and the chicken is blue in color, then you get double the number of points (10 points). This function should still add 5 points for clicking on the cow and 5 points for clicking on the chicken when it is not blue in color. Do not modify the parameters or add any more parameters.
7. (8 pts) Consider the following Alice world with Plato (with his hand out) and a basketball. Assume the basketball is oriented in the same direction as Plato and the basketball is directly above Plato’s hand.

Give the Alice code that does the following. Repeatedly move plato up 0.1 meters and the basketball down 0.1 meters until the ball is resting in Plato’s hand. At that time Plato should move back down to the ground and the basketball should move with him.
8. (18 pts) Assume there is a list of goldfish (the smaller fish in the picture below). The list is called school. The big fish below is not part of the list. The list may have some goldfish in it that are invisible. (Assume an invisible object means the isShowing property of that object has been set to false).

A. Write the function howManyVisible that returns the number of fish in the list school that are visible. For example, considering the picture above, the call to howManyVisible returns 5.

```
world.howManyVisible
world.howManyVisible No parameters
```
B. Write the method `eatVisibleFish` This method has the `bigFish` (who is not part of the list) turn toward each visible fish in the list `school` and eat them by moving close to them and then having them disappear (you do not have to move the `bigFish`’s mouth. For example, if the list `school` is initialized as shown on the previous page, then the result of the call “`eatVisibleFish`” is the picture on the previous page with all the little fish now invisible. Remember that the list may start out larger then 5 as it may contain some invisible fish before this method is called.

Complete the code below.

```
world.eatVisibleFish

world.eatVisibleFish  No parameters
```
9. (18 pts) Consider the array Visualization of sport balls called elements shown below. There is also a baseball behind the array (not part of the array). It is used in part B.

A) Write the method `World.jumping` that does the following. For each ball in the array (one at a time), the ball should move up a random amount either 2, 4, or 6 meters, and then move down the same random amount back to its spot in the array. For example, a call to `jumping` might randomly move the basketball up 4 meters, then back down 4 meters, then the football may move randomly up 2 meters, then back down 2 meters, etc.
B) Write the function `World.firstTaller` that has one object parameter called `ball` and does the following. This function returns the first ball in the array `elements` that is taller in height than the parameter `ball`. If no object is taller than the parameter `ball`, then return `ball`. For example, if the array looked like the figure above, then the call `world.firstTaller ball = baseball` would return `beachBall` since the `beachBall` is the first ball in the array starting at position 0 that is taller than the `baseball` (this baseball has been resized and is larger than the `basketball`, `football`, `volleyball`, `soccerBall`, `beachBall`, and `toyBall`).