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

- **random**
  - choose true, probabilityOfTrue, of the time
  - random number

- **string**
  - a joined with b
  - what as a string

- **ask user**
  - 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 year
  - day of month
  - day of week
  - day of week in month
  - is AM
  - is PM
  - hour of AM or PM
  - hour of day
  - minute of hour
  - second of minute

- **square root of a**
  - floor a
  - ceiling a
  - sin a
  - cos a
  - tan a
  - arccos a
  - arcsin a
  - arctan a
  - arctan2 a, b
  - a raised to the b power
  - natural log of a
  - e raised to the a power
  - EEEERemainder of a / b
  - round a
  - a converted from radians to degrees
  - a converted from degrees to radians
  - the b'th root of a
  - right, up, forward
Given below are the chicken properties and methods.

<table>
<thead>
<tr>
<th>Chicken's details</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>properties</strong></td>
<td><strong>methods</strong></td>
</tr>
<tr>
<td>create new variable</td>
<td></td>
</tr>
<tr>
<td>capture pose</td>
<td></td>
</tr>
</tbody>
</table>

- **color** = [ColorPicker]
- **opacity** = 1 (100%)
- **vehicle** = world
- **skin texture** = Chicken.TextureMap
- **fillingStyle** = solid
- **pointOfView** = position: 0, 0, -0.1; orientation: (0, 0, 0) 1
- **isShowing** = true

<table>
<thead>
<tr>
<th>create new method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chicken move</td>
</tr>
<tr>
<td>Chicken turn</td>
</tr>
<tr>
<td>Chicken roll</td>
</tr>
<tr>
<td>Chicken resize</td>
</tr>
<tr>
<td>Chicken say</td>
</tr>
<tr>
<td>Chicken think</td>
</tr>
<tr>
<td>Chicken play sound</td>
</tr>
<tr>
<td>Chicken move to</td>
</tr>
<tr>
<td>Chicken move toward</td>
</tr>
<tr>
<td>Chicken move away from</td>
</tr>
<tr>
<td>Chicken orient to</td>
</tr>
<tr>
<td>Chicken turn to face</td>
</tr>
<tr>
<td>Chicken point at</td>
</tr>
<tr>
<td>Chicken set point of view to</td>
</tr>
<tr>
<td>Chicken set pose</td>
</tr>
<tr>
<td>Chicken stand up</td>
</tr>
<tr>
<td>Chicken move at speed</td>
</tr>
<tr>
<td>Chicken turn at speed</td>
</tr>
<tr>
<td>Chicken roll at speed</td>
</tr>
<tr>
<td>Chicken constrain to face</td>
</tr>
<tr>
<td>Chicken constrain to point at</td>
</tr>
</tbody>
</table>
Given below are the chicken functions.

Tiles at the bottom of the Alice window.
1. (3 pts) Consider the following web page.

Unbreakable Luggage

The wheels on our luggage never fall off.

Which html code corresponds to this web page?

A) `<html>
   <head>
     <title>Unbreakable Luggage</title>
   </head>
   <body>
     <p> The wheels on our luggage <br>
         never fall off.
   </p>
   </body>
   </html>`

B) `<html>
   <head>
     <title>Unbreakable Luggage</title>
   </head>
   <body>
     <p> The wheels on our luggage never fall off.
   </p>
   </body>
   </html>`

C) `<html>
   <head>
   </head>
   <body>
     <h2>Unbreakable Luggage</h2>
     <p> The wheels on our luggage <br>
         never fall off.
     </p>
   </body>
   </html>`

D) `<html>
   <head>
   </head>
   <body>
     <h2>Unbreakable Luggage</h2>
     <p> The wheels on our luggage never fall off.
     </p>
   </body>
   </html>`
2. (6 pts) Consider the following html code:

```
<a href="frogs.html">frogs
and
<a href="snakes.html">snakes</a>
```

When displayed in a browser the html code above looks like the picture below on the right. The intention is for it to look like the picture below on the left.

```
frogs and snakes    frogs and snakes
```

a) Explain what the error is and give the correct html code to display it as in the picture on the left above.

b) Explain what the following html code does, assumptions it makes and describe how it is displayed in a browser.

```
<img src="apples.png">
<img src="apples.png">
<img src="apples.png">
```

c) Explain how the following html code will be displayed. Be precise.

```
Duke <i>will</i> win <b>again</b>
```
3. (3 pts) List three things that the Lynx browser could not do that Netscape can do:

4. (3 pts) Consider the following list on a web page.

   <ol>
   <li> Coach C</li>
   <li> Coach K</li>
   <ul>
   <li> Coach D</li>
   </ul>
   <li> Coach B</li>
   </ol>

Which one of the following is the html code to produce this web page component?

A)  • Coach C
    1. Coach K
    2. Coach D
    • Coach B

B)  • Coach C
    • Coach K
    1. Coach D
    • Coach B

C)  1. Coach C
    o Coach K
    o Coach D
    2. Coach B

D)  1. Coach C
    2. Coach K
    o Coach D
    3. Coach B
5. (3 pts) Consider the following HTML code.

```html
<table border=1>
<tr><td>Chordia</td></tr>
<tr><td>Rivera</td></tr>
<tr><td>Sully</td></tr>
<tr><td>Milanzi</td></tr>
<tr><td>Kibbe</td></tr>
<tr><td>Alvarez</td></tr>
<tr><td>Curtis</td></tr>
</table>
```

Which picture corresponds to this code?

A)  
```
<table>
<thead>
<tr>
<th>Chordia</th>
<th>Rivera</th>
<th>Sully</th>
</tr>
</thead>
<tbody>
<tr>
<td>Milanzi</td>
<td>Kibbe</td>
<td>Alvarez Curtis</td>
</tr>
</tbody>
</table>
```

B)  
```
<table>
<thead>
<tr>
<th>Chordia</th>
<th>Rivera</th>
<th>Sully</th>
<th>Milanzi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kibbe</td>
<td>Alvarez</td>
<td>Curtis</td>
<td></td>
</tr>
</tbody>
</table>
```

C)  
```
<table>
<thead>
<tr>
<th>Chordia</th>
<th>Rivera</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sully</td>
<td></td>
</tr>
<tr>
<td>Milanzi</td>
<td>Kibbe</td>
</tr>
<tr>
<td>Alvarez</td>
<td>Curtis</td>
</tr>
</tbody>
</table>
```

D)  
```
<table>
<thead>
<tr>
<th>Chordia</th>
<th>Rivera</th>
<th>Sully</th>
</tr>
</thead>
<tbody>
<tr>
<td>Milanzi</td>
<td>Kibbe</td>
<td>Alvarez</td>
</tr>
</tbody>
</table>
```

6. (3 pts) In Alice, which one of the following is not an Object type?

A) A Bunny  
B) A parameter  
C) A method  
D) The return value of a function

7. (3 pts) Explain why the command “move to” can cause the object to go into the ground when the command “move forward” for the same object does not move the object into the ground.
8. (14 pts) Consider the following Alice code in which the lines are numbered.

A) In line 2, list the name of the function and what type of value it returns.

B) Explain what must be true for line 4 to be executed when this program runs.

C) In line 5, list the words that are arguments.

D) In line 5, list the words that are parameters.

E) Name one function above that is a user-built class function.

F) In line 5, what type of value does the function `world.Something` return?

G) Which tab was line 6 dragged in from (properties, methods or functions tab)?
9. (4 pts) Consider the following world that has the three objects: tortoise, chicken and penguin (shown below from left to right) and given code. The world has been setup as shown below. The chicken is exactly 1.0 meter from the tortoise, and the chicken is exactly 1.0 meter from the penguin.

![Diagram of the world with tortoise, chicken, and penguin.]  

The diagram below is looking from above over the scene. The tortoise is represented by the T, the chicken is represented by the C, and the penguin is represented by the P. The animals are facing the bottom of the page. Using the diagram below, draw the path of tortoise and chicken as a solid line and the path of penguin as a dashed line.
10. (6 pts) Consider the following `world.Mystery` function.

A) What does `world.Mystery` return when the following call is made?

```
print world.mystery num1 = 2 ⊲ num2 = 1 ⊲ num3 = 3 ⊲
```

B) What does `world.Mystery` return when the following call is made?

```
print world.mystery num1 = 5 ⊲ num2 = 6 ⊲ num3 = 5 ⊲
```

C) Of the 5 return values, list all of them that can never be returned and explain why for each one.
11. (10 pts) Consider the following Alice world that has two objects: bunny and hawk.

The world starts as shown in the figure above with the hawk on the ground far behind the bunny. Write code to do the following in this order. When you move the hawk you do not need to flap his wings, just move him.

a) The hawk should face the bunny, then move up so he is at the height of the bunny.

b) The hawk should fly at the same height (but not moving the wings) stopping two meters before reaching the bunny.

c) The hawk will say a loud noise like SQUAWK.

d) At the same time: the bunny will turn its head towards the hawk and the hawk will move 2 meters and be touching the top of the bunny’s ears.

e) At the same time: the hawk will move forward 5 meters and will move up 5 meters, taking the bunny with him.
12. (8 pts) Complete the following class method called `crush` whose header is shown below. This method has two parameters, an object named “balloon,” and a string named “prize.” This method first has the motorboat face the balloon, then the motorboat runs over the balloon, stopping on top of it, saying the prize statement and having the balloon you just crushed becoming invisible.

```
motorboat.crush balloon = yellowBalloon prize = We win a book.
```

```
motorboat.crush balloon = greenBalloon prize = We win an ipad!
```

For example, suppose the scene starts as shown in the picture on the left above. In the first call above, the motorboat faces the yellowBalloon, moves over top of it, says “We win a book”, and makes the yellowBalloon invisible (the yellowBalloon is the one on the left). In the second call the motorboat faces the greenBalloon, moves over top of it, says “We win an ipad!”, and makes the greenBalloon invisible. The picture on the right above, shows the result after the two calls. Note the two balloons are no longer visible.
13. (10 pts) Complete the following function called `possibleHooper` that has three parameters, one object named `animal1`, one object named `animal2`, and one object named `animal3`, and returns the object that can fit through the hoop, another object in the scene. The animal must fit based on its height and width. Don’t worry about the thickness of the hoop, just use its diameter for the determination of whether an object fits through it. Assume at least one animal fits, and if more than one does, return any one that fits.

A) Complete the function below.

```javascript
world.possibleHooper
world.possibleHooper(Obj animal1, Obj animal2, Obj animal3)
```
B) Assume there are three objects hawk, chicken and flamingo, in addition to hoop. Assume at least one of them can fit through the hoop. Give Alice code to have one of the objects that will fit through the hoop say “I’ll fit through the hoop.” Then have that object turn to face the hoop and then move through the hoop stopping 5 meters past it. You must call the function you wrote in Part A) to receive full credit.
14. (14 pts) Consider an Alice world with three cones named yellowCone, blackCone and redCone, and a kangarooRobot.

A) (4 pts) Complete the following function called `furthestDistance` that has two object parameters named `cone1` and `cone2` and for the parameter that is furthest away from the kangarooRobot, this function returns a number that is the distance between that furthest cone and the kangarooRobot. For example, if the distance between the kangarooRobot and cone1 was 8 meters, and the distance between the kangarooRobot and cone2 was 6 meters, then 8 would be returned.
B) (5 pts) Complete the following function called coneFurthestAwayOf3 that returns the cone (of three cones) that is furthest away from the kangarooRobot. To receive full credit you must call the function from part A. Note that function returns a number and this function returns an object.

```javascript
kangarooRobot.coneFurthestAwayOf3
```
C) (5 pts) Complete the following Alice code. Have the kangarooRobot turn to face and then move over stopping on top of the cone that is the furthest away. The three cones are called yellowCone, blackCone and redCone. To receive full credit you must call the function in Part B.
(extra page, must turn in)