Review of HTML, CSS, and JavaScript

Quiz, 10 questions

1 point

1. What is the purpose of adding CSS to a web page?

- To put images on the web page
- To make the web page interactive
- **To style the web page**
- To layout the web page

1 point

2. Which of the following are examples of nesting? Select all that are correct.

- An image
- A list inside a table
- **A list of lists**
- **A for loop inside a for loop**
- A table

1 point

3. Consider the following HTML and CSS to make a web page.

HTML:

```
<head>
</head>
<title>Cities</title>
<body>
<p><img src="http://s12.postimg.org/yj9oyjs3x/DSCN6056_copy.jpg"/>
</p>
<p>
<ol>
  <li class = oddNums>New York</li>
    <ol>
      <li>Empire State Building</li>
      <li>Statue of Liberty</li>
      <li>Times Square</li>
    </ol>
  <li>Los Angeles</li>
  <li class = oddNums>Chicago</li>
</ol>
</p>
</body>
```

CSS:

```
body {
  background-color : #567898;
}
oddNums {
  color : purple;
}
```

Which of the following are errors in this code? Select all that are correct.

- **The unordered list inside the ordered list should go inside the list element New York, not after it.**
- There should not be semicolons at the ends of the statements in the CSS.
- **In the CSS there should be a dot before oddNums to indicate that it is a class.**
- **The <title> tag should be inside the <head> tag.**
- **The HTML is missing <html> tags.**
- The <img> tag is missing the width attribute.
- The property background-color is not the correct property to change the background of the page.

1 point

4. Consider the following image in which the upper left quadrant is cyan, the upper right quadrant is green, the lower left quadrant is blue, and the lower right quadrant is black:

Now consider the code that attempts to create that image but has a mistake, and instead produces this image, in which the upper left quadrant is blue instead of cyan:

Here is the code:

```
var img = new SimpleImage(200,200);
for (var px of img.values()){
    var x = px.getX();
    var y = px.getY();
    if (x < img.getWidth()/2){
        px.setBlue(255);
    }
    else {
        if (y < img.getHeight()/2){
            px.setGreen(255);
        }
    }
}
print (img);
```

Which of the following is the best explanation of why this code doesn't produce the first image?

- First the upper half of the image is made green, then when the left half is made blue it overwrites the green pixels and makes them blue.
- The if statement inside the else statement checks if pixels are in the upper half and the right half of the image, so only the upper right quadrant of the image is made green.
- **The code inside the else statement is only applied to pixels that did not satisfy the first if statement. So only pixels in the upper half of the image that are not also in the left half of the image are made green by the if statement inside the else statement.**

1 point

5. Consider the following short program that defines a function to make an image darker by a certain amount and applies it to the image chapel.png.

```
function makeDarker(image,amount){
    for (var px of image.values()){
        px.setRed(px.getRed() -amount);
        px.setGreen(px.getGreen() -amount);
        px.setBlue(px.getBlue() -amount);
    }
}
img = new SimpleImage("chapel.png");
img = makeDarker(50);
print(img);
```

Which of the following are errors in the program? Select all that are correct.

- **The line that initializes the variable *img* is missing the keyword *var*.**
- **The call to *makeDarker* does not pass an image as an argument.**
- **The function *makeDarker* is missing a return statement so there will be an error when the program assigns the return value of *makeDarker* to the variable *img*.**
- **The function *makeDarker* doesn't make an image darker, it makes an image gray, because it sets the red, green, and blue values to the same value.**
- **The line img = makeDarker(50); is missing the *function* keyword.**

1 point

6. Imagine you want to write a program to turn an image into a mirror image of itself. Which of the following would be the best approach to take?

- Write code to solve the problem, test and debug your program, improve your program by adding more features.
- Work small examples by hand, write down what you did, look for patterns, translate your algorithm to code, test and debug your program.
- **Gather domain knowledge, work small examples by hand, write down what you did, look for patterns, translate your algorithm to code.**

1 point

7. Consider the following JavaScript code.

```
var grayImage = null;
var image;
function loadImage(){
  var ff = document.getElementById("fbutton");
  gcanvas = document.getElementById("can");
  doClear(c);
  image = new SimpleImage(ff);
  image.drawTo(can);
}
function makeGray(theImage) {
  for (var pix of theImage.pixels()){
    var total = pix.getGreen() + pix.getRed() + pix.getBlue();
    var avg = total/3;
    pix.setGreen(avg);
    pix.setBlue(avg);
    pix.setRed(avg);
  }
  return theImage;
}
```

Which of the variables are global variables? Select all that are correct.

- **image**
- ff
- avg
- pix
- theImage
- **grayImage**

1 point

8. Which is the appropriate event handler to do something once a file has loaded?

- onmouseover
- oninput
- **onchange**
- onclick

1 point

9. Consider the following code that calls the function filterGreen (code for this function not shown) to apply a green filter to the image greenImage.

```
function doGreen() {
  if (imageIsLoaded(greenImage)) {
    filterGreen();
  }
}
```

What line needs to be added to this code to display the final image on the canvas? You can assume that there is a variable named **canvas** that can be used to reference the canvas.

greenImage.drawTo(canvas);

1 point

10. Consider the examples you have seen of web pages that enable users to upload images and add filters to them. Which of the following describes what happens when the user clicks a button to add a filter to an image?

- The mouseover event handler calls the function that draws the image to the canvas, then the onclick event handler calls the function that applies the filter to the image.
- The onclick event handler calls the function that draws the image to the canvas, then the filter is added.
- The onclick event handler calls a function that applies the filter to the image, then the filtered image is drawn on the canvas.
- **The onclick event handler allows the user to choose an image to apply the filter to, then it calls a function that applies the filter to the image, and the filtered image is drawn on the canvas.**

I, **Ning Zheng**, understand that submitting work that isn't my own may result in permanent failure of this course or deactivation of my Coursera account.

Learn more about Coursera's Honor Code

Submit Quiz