COMP4021
Internet Computing

# Final Review

Gibson Lam

## Question

You have been asked to put some image content on a webpage. In the following table, you need to choose the best image type (SVG, PNG or JPEG) to use on the page for each of the image content by putting a tick at the appropriate space.

| Image Content | Use SVG | Use PNG | Use JPEG |
|---|---|---|---|
| HKUST Logo | | | |
| Gibson's Face | | | |
| A plot of $y = e^x$ | | | |

# Answer

| Image Content | Use SVG | Use PNG | Use JPEG |
|---|---|---|---|
| HKUST Logo | ✓ *OR* | ✓ | |
| Gibson's Face | | | ✓ |
| A plot of $y = e^x$ | ✓ *OR* | ✓ | |

## Question

Here is the HTML of an example shown in the class.

```
<!DOCTYPE html>
<html>
<head>
  <title>The Example Document</title>
</head>
<body>
  <ol>
    <li>Breakfast <b>$15.00</b></li>
    <li>Lunch <b>$25.00</b></li>
    <li>Dinner <b>$50.00</b>
      <ul>
        <li>Main course <b>$30.00</b></li>
        <li>Desert <b>$20.00</b></li>
      </ul>
    </li>
  </ol>
</body>
</html>
```

## Question (cont.)

The following style rule has been added to the example:

```
<style>
body li li > b { color: red; }
</style>
```

Please circle clearly the part(s) of HTML that is affected by the above style rule. If you think nothing in the HTML can be affected by the rule, please write “Nothing” next to the above rule.

# Answer

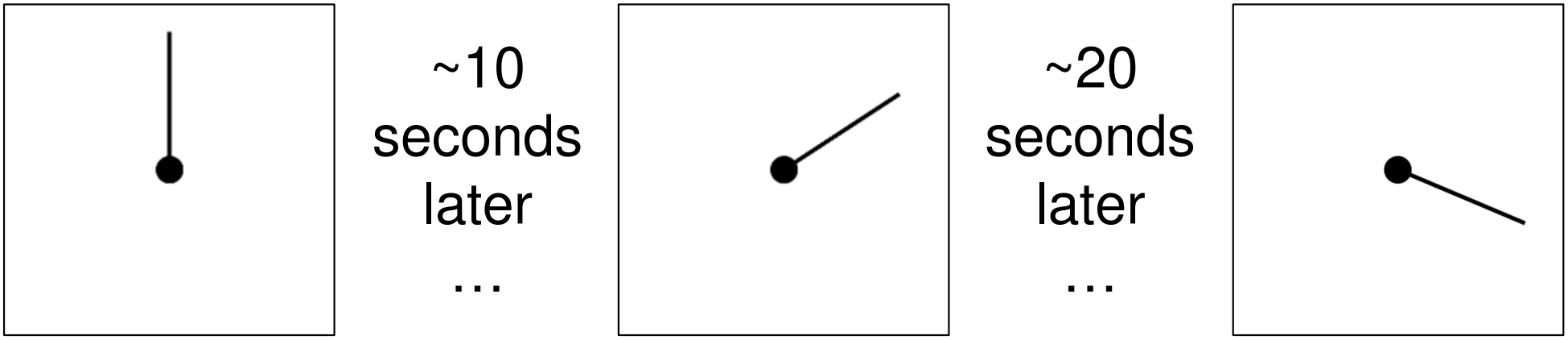
```
<!DOCTYPE html>
<html>
<head>
  <title>The Example Document</title>
</head>
<body>
  <ol>
    <li>Breakfast <b>$15.00</b></li>
    <li>Lunch <b>$25.00</b></li>
    <li>Dinner <b>$50.00</b>
      <ul>
        <li>Main course <b>$30.00</b></li>
        <li>Desert <b>$20.00</b></li>
      </ul>
    </li>
  </ol>
</body>
</html>
```

```
<style>
body li li > b { color: red; }
</style>
```

1. Breakfast **$15.00**
2. Lunch **$25.00**
3. Dinner **$50.00**
    - Main course **$30.00**
    - Desert **$20.00**

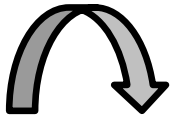
# Question

A stopwatch program has been created on a webpage so that it works like this:



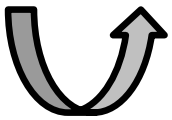

At the start of the program, the second hand points at 0 second. It then goes around the watch in 1 minute and stops when it points at 0 second again. The HTML code of the program is shown on the next page. Please fill in the blanks so that the program works as described.

# Question (cont.)



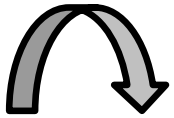
```
<!DOCTYPE html>
<html>
<head>
 <title>Stopwatch</title>
 <style>
 * { box-sizing: border-box; }
 @keyframes around {
  from { transform: rotate(0deg); }
  to   { transform: rotate(360deg); }
 }
 #watch { position: relative;
          width: 500px; height: 500px;
          border: 1px solid black; }
 #face  { width: 500px; height: 500px; }

 _____________________________ {
  animation-name:            _________________________;
  animation-duration:        _________________________;
  animation-timing-function: _________________________;
 }
 .hand { position: absolute; background-color: black; }
 </style>
</head>
```



```
<body>
 <div id="watch">
  <div id="face">
   <div class="hand"
        style="left: 230px;
        top: 230px; width: 40px;
        height: 40px;
        border-radius: 20px"></div>
   <div class="hand"
        style="left: 247px;
        top: 50px; width:6px;
        height:200px"></div>
  </div>
 </div>
</body>
</html>
```

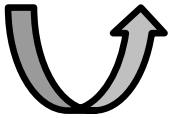
# Answer



```
<!DOCTYPE html>
<html>
<head>
 <title>Stopwatch</title>
 <style>
 * { box-sizing: border-box; }
 @keyframes around {
  from { transform: rotate(0deg); }
  to   { transform: rotate(360deg); }
 }
 #watch { position: relative;
          width: 500px; height: 500px;
          border: 1px solid black; }
 #face  { width: 500px; height: 500px; }
```

```
<body>
 <div id="watch">
  <div id="face">
   <div class="hand"
        style="left: 230px;
        top: 230px; width: 40px;
        height: 40px;
        border-radius: 20px"></div>
   <div class="hand"
        style="left: 247px;
        top: 50px; width:6px;
        height:200px"></div>
  </div>
 </div>
</body>
</html>
```

```
 #face { 
  animation-name:            around;
  animation-duration:          60s;
  animation-timing-function: linear;
 }
 .hand { position: absolute; background-color: black; }
 </style>
</head>
```



## Question

Here is some SVG code.

After loading this SVG document into a browser, how many nodes are there in the DOM of this document?

```
<svg xmlns="http://www.w3.org/2000/svg">
  <g id="my_group_name">
    <circle cx="100" cy="120" r="30"
            style="fill:red"/>
    <circle cx="200" cy="120" r="30"
            style="fill:red"/>
    <circle cx="150" cy="150" r="100"
       style="fill:none;stroke:blue;
              stroke-width:3"/>

  </g>
</svg>
```

A. 0

B. 1

C. 5

D. More than 5

## Answer

Here is some SVG code.

After loading this SVG document into a browser, how many nodes are there in the DOM of this document?

```
<svg xmlns="http://www.w3.org/2000/svg">
  <g id="my_group_name">
    <circle cx="100" cy="120" r="30"
            style="fill:red"/>
    <circle cx="200" cy="120" r="30"
            style="fill:red"/>
    <circle cx="150" cy="150" r="100"
       style="fill:none;stroke:blue;
              stroke-width:3"/>
  </g>
</svg>
```

A. 0

B. 1

C. 5

D. More than 5

## Question

Here is some content transmitted between a browser and a server.

```
HTTP/1.1 200 OK
Date: Tue, 8 May 2022 16:30:00 GMT
Server: Apache
Cache-Control: private
Connection: close
Content-Type: application/json

{ "crazy": "yes" }
```

Which one of the following statements is correct regarding the above content?

A. It is an HTTP GET request

B. It is an HTTP POST request

C. The HTTP request is successfully handled

D. The transmitted content is a JavaScript file

**Answer**

Here is some content transmitted between a browser and a server.

```
HTTP/1.1 200 OK
Date: Tue, 8 May 2022 16:30:00 GMT
Server: Apache
Cache-Control: private
Connection: close
Content-Type: application/json

{ "crazy": "yes" }
```

Which one of the following statements is correct regarding the above content?

A. It is an HTTP GET request

B. It is an HTTP POST request

C. The HTTP request is successfully handled

D. The transmitted content is a JavaScript file

# Question

An HTML page and a JSON file are stored in the same folder in the web server. The content of the JSON file, *rainbow.json*, is shown below.

```
{ "red":{"hex":"#FF0000","rgb":[255,0,0]},
  "orange":{"hex":"#FF8000","rgb":[255,128,0]},
  "yellow":{"hex":"#FFFF00","rgb":[255,255,0]},
  "green":{"hex":"#00FF00","rgb":[0,255,0]},
  "blue":{"hex":"#0000FF","rgb":[0,0,255]},
  "indigo":{"hex":"#4B0082","rgb":[75,0,130]} }
```

On the next page is the JavaScript code and body content of the HTML page. You need to fill in the blanks so that after loading the HTML page, the page immediately turns to have green background, based on the colour data returned by the AJAX call.

# Question (cont.)

```
{ "red":{"hex":"#FF0000","rgb":[255,0,0]},
  "orange":{"hex":"#FF8000","rgb":[255,128,0]},
  "yellow":{"hex":"#FFFF00","rgb":[255,255,0]},
  "green":{"hex":"#00FF00","rgb":[0,255,0]},
  "blue":{"hex":"#0000FF","rgb":[0,0,255]},
  "indigo":{"hex":"#4B0082","rgb":[75,0,130]} }
```

```
<!DOCTYPE html>
<html>
<head>
  <title>JSON</title>
  <script src="...jQuery source..."></script>
  <script>
  $(document).ready(function() {
    fetch("rainbow.json")
    .then((response) => response.json())
    .then((data) => {
      $("body").eq(0).css("_____________________________",

                          _____________________________);
    });
  });
  </script>
</head>
<body></body>
</html>
```

# Answer

```
{ "red":{"hex":"#FF0000","rgb":[255,0,0]},
  "orange":{"hex":"#FF8000","rgb":[255,128,0]},
  "yellow":{"hex":"#FFFF00","rgb":[255,255,0]},
  "green":{"hex":"#00FF00","rgb":[0,255,0]},
  "blue":{"hex":"#0000FF","rgb":[0,0,255]},
  "indigo":{"hex":"#4B0082","rgb":[75,0,130]} }
```

```
<!DOCTYPE html>
<html>
<head>
  <title>JSON</title>
  <script src="...jQuery source..."></script>
  <script>
  $(document).ready(function() {
    fetch("rainbow.json")
    .then((response) => response.json())
    .then((data) => {
      $("body").eq(0).css("background-color",
                          data.green.hex);
    });
  });
  </script>
</head>
<body></body>
</html>
```

*Or other equivalent answers*

# Question

Here is an HTML form.

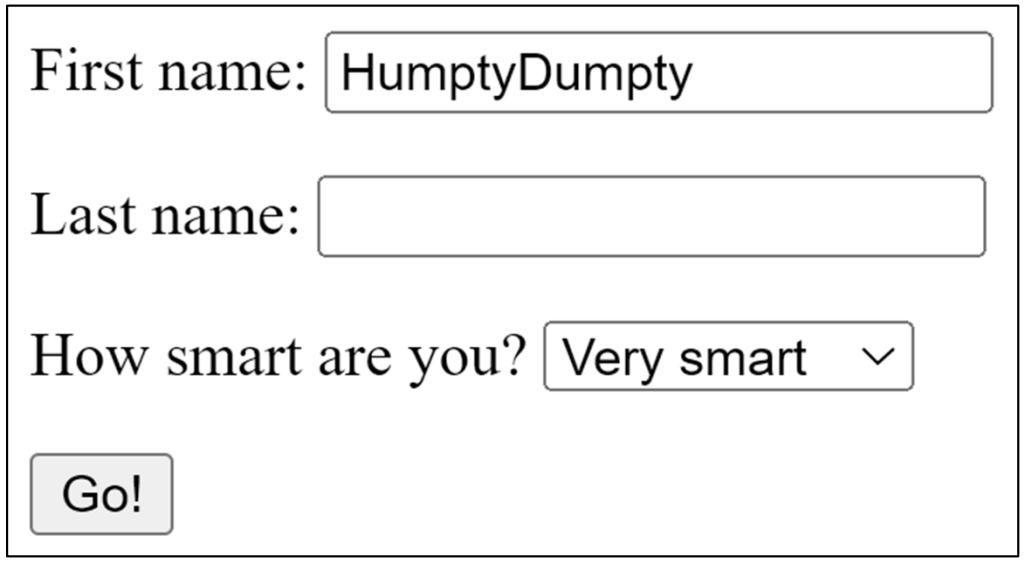
```
<!DOCTYPE html>
<html>
<head><title>Silly Form</title></head>
<body>
  <form method="get" action="/sillyform">
    <p>First name: <input type="text" name="first"></p>
    <p>Last name: <input type="text" name="last"></p>
    <p>How smart are you?
    <select name="smartness">
      <option value="5">Super smart</option>
      <option value="3">Very smart</option>
      <option value="4">Quite smart</option>
      <option value="2">Okay smart</option>
      <option value="1">Super dumb</option>
    </select>
    </p>
    <input type="hidden" name="silliness" value="5">
    <input type="submit" value="Go!">
  </form>
</body>
</html>
```

## Question (cont.)

The form data will be submitted to the following Express server:

```
const express = require("express");
const app = express();
app.use(express.static("./"));
app.get("/sillyform", (req, res) =>
        res.json(req.query));
app.listen(8000);
```

Here is an image showing some inputs entered into the form.

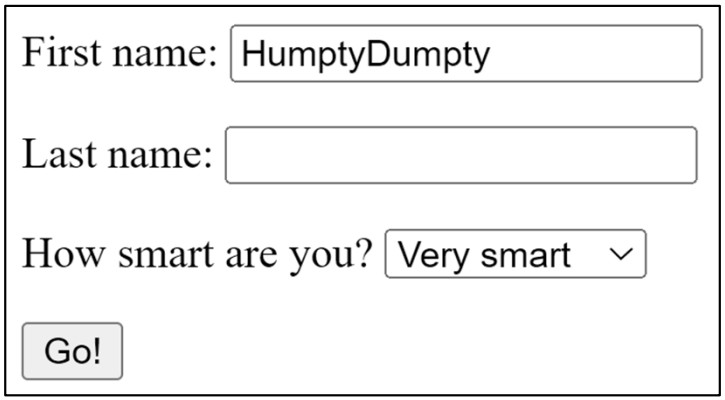


# Question (cont.)

If you press the ‘Go!’ button from the form, what is the display of the browser?

```
const express = require("express");
const app = express();
app.use(express.static("./"));
app.get("/sillyform", (req, res) =>
        res.json(req.query));
app.listen(8000);
```

```
<!DOCTYPE html>
<html>
<head><title>Silly Form</title></head>
<body>
  <form method="get" action="/sillyform">
    <p>First name: <input type="text" name="first"></p>
    <p>Last name: <input type="text" name="last"></p>
    <p>How smart are you?
    <select name="smartness">
      <option value="5">Super smart</option>
      <option value="3">Very smart</option>
      <option value="4">Quite smart</option>
      <option value="2">Okay smart</option>
      <option value="1">Super dumb</option>
    </select>
    </p>
    <input type="hidden" name="silliness" value="5">
    <input type="submit" value="Go!">
  </form>
</body>
</html>
```

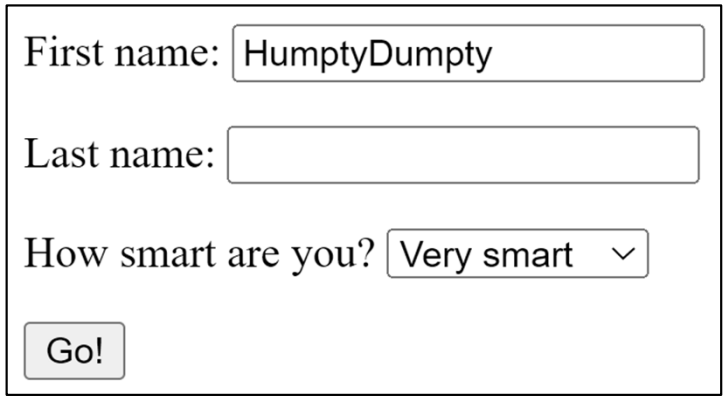


# Answer

If you press the ‘Go!’ button from the form, what is the display of the browser?

```
const express = require("express");
const app = express();
app.use(express.static("./"));
app.get("/sillyform", (req, res) =>
        res.json(req.query));
app.listen(8000);
```

```
<!DOCTYPE html>
<html>
<head><title>Silly Form</title></head>
<body>
  <form method="get" action="/sillyform">
    <p>First name: <input type="text" name="first"></p>
    <p>Last name: <input type="text" name="last"></p>
    <p>How smart are you?
    <select name="smartness">
      <option value="5">Super smart</option>
      <option value="3">Very smart</option>
      <option value="4">Quite smart</option>
      <option value="2">Okay smart</option>
      <option value="1">Super dumb</option>
    </select>
    </p>
    <input type="hidden" name="silliness" value="5">
    <input type="submit" value="Go!">
  </form>
</body>
</html>
```




```
{"first":"HumptyDumpty",
 "last":"",
 "smartness":"3",
 "silliness":"5"}
```

## Question

Here is an Express server file.

```
const express = require("express");
const app = express();
app.set("view engine", "ejs")

let ouch = 1;
app.get("/ouch", (res, req) => {
  if (res.query.ouch) {
    let { ouch } = res.query;
    req.render("ouch", { ouch });
    ouch++;
  }
  else
    req.redirect(`ouch?ouch=${ouch}`);
});
app.listen(8000)
```

## Question (cont.)

The EJS file, *ouch.ejs*, is shown below.

```
<% effects = ["BOOM!", "BAM!", "KA-POW!",
              "POW!", "ZAP"]; %>
<!DOCTYPE html>
<html>
<head>
  <title>Punch!</title>
</head>
<body>
  <p style="color: red">
  <%= effects[ouch % 5] %></p>
  <button onclick="location.reload()">
  Punch!</button>
</body>
</html>
```

## Question (cont.)

The EJS page can be accessed through:

http://localhost:8000/ouch

If you load the page using the above URL and then press the ‘Punch!’ button eight times, what is then the text displayed above the ‘Punch!’ button?

A. BOOM!

B. BAM!

C. KA-POW!

D. POW!

E. ZAP

# Answer

```
const express = require("express");
const app = express();
app.set("view engine", "ejs")

let ouch = 1;
app.get("/ouch", (res, req) => {
  if (res.query.ouch) {
    let { ouch } = res.query;
    req.render("ouch", { ouch });
    ouch++;
  }
  else
    req.redirect(`ouch?ouch=${ouch}`);
});
app.listen(8000)
```

```
<% effects = ["BOOM!", "BAM!",
              "KA-POW!", "POW!",
              "ZAP"]; %>
<!DOCTYPE html>
<html>
<head>
  <title>Punch!</title>
</head>
<body>
  <p style="color: red">
  <%= effects[ouch % 5] %></p>
  <button onclick="location.reload()">
  Punch!</button>
</body>
</html>
```

1. Load the page http://localhost:8000/ouch
2. Then press the button eight times

A. BOOM!
B. BAM!
C. KA-POW!
D. POW!
E. ZAP

# Question

A user tracking website *tracker.com* has made a simple ‘pixel tracker’ to track the behaviour of web users. The tracker can be accessed through:

http://tracker.com/track

The output of the tracker is a 1x1 transparent GIF pixel. The behaviours of users (i.e. the number of times they visited some websites) are stored in a cookie called ‘visit’.

## Question (cont.)

To put the tracker into your own website, you put it in an `<img>` tag and pass the tracker the id of your website.

For example, if the id of your website is ‘comp4021’, you can add a pixel tracker using the `<img>` tag shown below.

```
<img src="http://tracker.com/track?id=comp4021">
```

The code of the tracker is shown on the next page.

# Question (cont.)

```
const express = require("express");
const cookieParser = require("cookie-parser");
const app = express();

app.use(cookieParser());
app.get("/track", (req, res) => {
  const { id } = req.query;
  if (!id) {
    res.sendStatus(400);
    return;
  }
  let visit = {};
  if (req.cookies.visit)
    visit = JSON.parse(req.cookies.visit);
  if (id && visit[id])
    visit[id]++;
  else
    visit[id] = 1;

  res.cookie("visit", JSON.stringify(visit), {
    expires: new Date("2099-01-01")
  });

  res.sendFile(__dirname + "/pixel.gif");
});

app.listen(8000);
```

*Send a GIF file as output*

## Question (cont.)

If two companies, ‘My Food Store’ and ‘5 Stars Tutors’, put the pixel tracker into their websites, where will the ‘visit’ cookie be stored if some users access those websites?

A. The server of the ‘My Food Store’ website

B. The server of the ‘5 Stars Tutors’ website

C. The server of tracker.com

D. The users’ computers

## Answer

If two companies, ‘My Food Store’ and ‘5 Stars Tutors’, put the pixel tracker into their websites, where will the ‘visit’ cookie be stored if some users access those websites?

A. The server of the ‘My Food Store’ website

B. The server of the ‘5 Stars Tutors’ website

C. The server of tracker.com

D. The users’ computers

# Question

Based on the previous question, let's assume 'My Food Store' and '5 Stars Tutor' have used the pixel tracker with an id of 'myfood' and '5stars' respectively.

If a user accesses the website of 'My Food Store' three times and the website of '5 Stars Tutors' four times, what will be content of the cookie?

# Answer

```
{"myfood":3,
 "5stars":4}
```

```
const express = require("express");
const cookieParser = require("cookie-parser");
const app = express();

app.use(cookieParser());
app.get("/track", (req, res) => {
  const { id } = req.query;
  if (!id) {
    res.sendStatus(400);
    return;
  }
  let visit = {};
  if (req.cookies.visit)
    visit = JSON.parse(req.cookies.visit);
  if (id && visit[id])
    visit[id]++;
  else
    visit[id] = 1;

  res.cookie("visit", JSON.stringify(visit), {
    expires: new Date("2099-01-01")
  });

  res.sendFile(__dirname + "/pixel.gif");
});

app.listen(8000);
```

*Send a GIF file as output*