Delft University of Technology

CSE1500 Web & Database Technology

Resit WEB Exam

Date: 07/04/2022
Time Limit: 90 minutes

Instructions:

- This exam contains 14 pages (including this cover page) and 30 multiple-choice questions. Check to see if any pages are missing.
- All questions are worth 1 point.
- The usage of books, notes, old exams, and other written resources is explicitly FORBIDDEN during the exam. The use of electronic aids such as smartphones, laptops, etcetera, is ALSO NOT allowed.
- **There is only one right answer for each question.**
- The order of the answers on your answer form is not always A-B-C-D.
- **There are four possible answers for each question. You can select A, B, C, or D.** It is possible that answer options for a single question are placed cross two pages.
- Be sure to fill in all header information on the answer form.
- Some questions refer to source code listed a page earlier. **Feel free to disassemble your copy of the exam, so that you can work comfortably.**
- This exam copy is yours, you can take it home.

Good Luck!

Question 1

How many of the following statements about the body of an HTTP request are TRUE?

(1) The body is the final part of an HTTP request.

(2) Not all HTTP requests have a body.

(3) The length of the body is restricted by the `SIZE` attribute as stated in the http header.

A. 0

B. 1

C. 2

D. 3

Question 2

Which of the following statements about the `Cache-Control` header is TRUE?

A. The directive `public` means that any cache can store the HTTP response.

B. The directive `private` means that only caches that require authentication can store the HTTP response.

C. The directive `regional` means that only caches in the geographic vicinity of the user who made the HTTP request can store the HTTP response.

D. The directive `authenticated` means that only caches that require authentication can store the HTTP response.

Question 3

Which of the following statements about redirections in HTTP is TRUE?

A. Redirection is triggered by a server sending a special redirect response to a request. Redirect responses have status codes that start with 3, and a `Location` header holding the URL to redirect to.

B. Redirection is triggered by the client sending a special redirect request to the server to access the server holding a copy of the requested resource that is geographically closest to the client. The server responds with a redirect response which contains a `Redirect` header holding the web cache IP address that is closest to the client.

C. Redirection is triggered by the server when encountering a request for a resource that requires authentication. The `WWW-Authenticate` header redirects the client to a URL where the client can authenticate itself to the server.

D. Redirection is triggered by the client after encountering a 404 response from the server. The client requests an alternative URL for the resource from the server with the `Alt` header set to `True` in the HTTP request.

Question 4

How many of the following statements about Punycode are TRUE?

(1) Punycode allows URLs with unicode characters to be translated nonreversibly into an ASCII string.

(2) Punycode allows URLs with ASCII characters to be translated nonreversibly into unicode characters.

(3) A URL in ASCII format remains the same after Punycode encoding.

(4) A URL containing mixed scripts cannot be transformed into a valid Punycode representation.

A. 1

B. 2

C. 3

D. 4

Question 5

How many of the following statements about HTTP basic authentication are TRUE?

(1) The `realm` describes the protection area: if several web resources on the same server require authentication within the same realm, a single user/password combination should be sufficient to access all of them.

(2) The client sends username and password encrypted to the server via the `HTTP-Authorization` header field.

(3) Base-64 encoding of username and password provides an additional layer of encryption.

(4) The server sends a `401` status code to indicate to a client that the requested web resource requires authentication.

A. 1

B. 2

C. 3

D. 4

Question 6

Consider the following telnet exchange on the terminal. Which of the following statements about it is TRUE?

```
bash-3.2$ telnet microsoft.com 80
Trying 104.215.148.265 ...
Connected to microsoft.com.
Escape character is '^]'.
HEAD / HTTP/1.1
host:microsoft.com

HTTP/1.1 200 OK
...

Connection closed by foreign host.
```

A. The IP address shown here is not a valid IPv4 address.

B. The Domain Name System server resolution step is missing in this telnet exchange.

C. The initial `telnet` command is invalid as the port number should be appended only to the `host` header value.

D. The response cannot have status code `200` if the HTTP method is `HEAD`.

Question 7

Consider the URL below:

```
ftp://anonymous:mypass@ftp.csx.cam.ac.uk/?date=today&localtime=utc
```

What URL component is not present in this URL?

A. `<scheme>`

B. `<host>`

C. `<path>`

D. `<query>`

Question 8

What does the *First Input Delay* (FID) metric measure?

A. FID measures the time between the client making an HTTP request and the server sending all subsequent cascading HTTP responses.

B. FID measures the time between the first input provided by the server via WebSockets and the second input provided by the server.

C. FID measures the time between a user first interacting with a page and the browser processing corresponding event handlers.

D. FID measures the time it takes to establish the first long polling request-response cycle between client and server.

Question 9

When executing the following piece of JavaScript in the browser's web console, what will be the output on the console?

```
amount = 10;
function changeColorShade(amount){
    rgb = [0, 0, 0];
    return function(rgb){
        return [Math.min(rgb[0]*amount, 255),
                Math.min(rgb[1]*amount, 255),
                Math.min(rgb[2]*amount, 255)]
    }
}
lighter = changeColorShade(1.2);
darker = changeColorShade(0.8);

console.log(lighter([200,10,60]))
console.log(darker([200,10,60]))
```

A. The output on the console is:

```
[ 255, 255, 255 ]
[ 255, 255, 255 ]
```

B. The output on the console is:

```
[ 0, 0, 0 ]
[ 0, 0, 0 ]
```

C. The output on the console is:

```
[ 240, 12, 72 ]
[ 160, 8, 48 ]
```

D. The output on the console is:

```
[ 200, 10, 60 ]
[ 200, 10, 60 ]
```

Question 10

When executing the following piece of JavaScript in the browser's web console, what will be the output on the console?

```
1  window.used = true
2
3  function Color(n, m, u){
4      this.name = n;
5      this.mood = m;
6      this.used = u;
7
8      this["using"] = function(){
9          this.used = true;
10     }
11     this.hasBeenUsed = function(){
12         return this.used;
13     }
14 }
15
16 let c = new Color("tomato", "happy", false);
17
18 Color.prototype.using = function(){
19     used = 1;
20 }
21
22 console.log(c.hasBeenUsed())
23 c.using()
24 console.log(c.hasBeenUsed())
```

A. `true`
`true`

B. `false`
`false`

C. `true`
`false`

D. `false`
`true`

Question 11

In JavaScript, the scopes of values and expressions depend on where and how they are declared. For the following four descriptions of variable declarations, select the correct scope.

`var` declared within a function	(1)
`var` declared outside of a function	(2)
`let` and `const`	(3)
variable declaration without `var`, `let`, `const`	(4)

A. (1) local, (2) global, (3) block, (4) global

B. (1) global, (2) global, (3) local, (4) block

C. (1) block, (2) local, (3) local, (4) global

D. (1) block, (2) block, (3) local, (4) global

Question 12

In JavaScript, what does the *object literal* notation refer to?

A. An object initializer which is a comma-delimited list of pairs of property names and associated values of an object, enclosed in curly braces.

B. An object initializer using the `Object.create` method where property names are enclosed in square brackets.

C. An object initializer that makes use of an immediate invoked function expression.

D. An object initializer where the object constructor is a normal function that is prepended by the `new` keyword.

Question 13

Which of the statements below about the following piece of code is TRUE?

```
1  function Color(n) {
2    this.name = n;
3  }
4
5  Color.prototype.getName = function () {
6    return this.name;
7  };
8
9  function ColorMood(n, m) {
10   Color.call(this, n);
11   this.mood = m;
12 }
13
14 ColorMood.prototype = Object.create(Color.prototype)
15 ColorMood.prototype.constructor = ColorMood;
16
17 ColorMood.prototype.getMood = function () {
18   return this.mood;
19 };
20
21 function ColorMoodUsage(n, m, u) {
22   ColorMood.call(this, n, m);
23   this.usage = u;
24 }
25
26 ColorMoodUsage.prototype = Object.create(ColorMood.prototype)
27 ColorMoodUsage.prototype.constructor = ColorMoodUsage;
28
29 let c1 = new Color("seagreen");
30 let c2 = new ColorMood("gold", "happy");
31 let c3 = new ColorMoodUsage("tomato", "friendly", 0);
32
33 ColorMoodUsage.prototype.incrUsage = function () {
34   return ++this.usage;
35 };
36
37 console.log(c3.getName());
38 console.log(c3.getMood());
39 console.log(c3.incrUsage());
```

A. Object `c3` has a direct link to one prototype object.

B. Object `c3` tries to access properties (`getName`, `getMood`) that the object does not have. Due to a broken prototype chain, no matching properties are found in the chain.

C. Property `incrUsage` was added to the prototype after the creation of object `c3` and thus `c3` cannot access this property.

D. As the properties `name` and `mood` have not been defined on the prototype of `c3`, they are not accessible to object `c3`.

Question 14

Which of the following is not a method of the browser's `document` object?

A. `getElementsByClassName`

B. `getElementsByTagName`

C. `getElementsByID`

D. `getElementsByName`

Question 15

Consider the web page defined below. Assume a user opens the web page in a modern browser. The user presses key 1 three times and then presses key 2 two times. What is the font size of the two paragraphs after these actions?

```
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <meta charset="UTF-8" />
5      <title>Smaller / bigger</title>
6      <style>
7        body {
8          font-size: 20px;
9        }
10     </style>
11   </head>
12   <body>
13     <p>
14       This is the first paragraph.
15     </p>
16
17     <p>
18       This is the second paragraph.
19     </p>
20
21     <script>
22       function getSize(e){
23         return parseInt(window.getComputedStyle(e).fontSize);
24       }
25
26       document.body.addEventListener("keydown", function(e){
27         if(e.code == 'Digit1'){ //key "1"
28           document.body.style.fontSize=2 * getSize(this) + "px";
29         }
30       })
31
32       document.body.addEventListener("keydown", function(e){
33         if(e.code == 'Digit2'){ //key "2"
34           document.body.style.fontSize = 0.5 * getSize(this) + "px";
35         }
36       })
37     </script>
38   </body>
39 </html>
```

A. The paragraphs' font size remains at `20px` as event listeners cannot be attached directly to `document.body`.

B. The font size of the paragraphs cannot be determined as their font size is never set in the first place.

C. The font size of the paragraphs is `5px` as the second event listener (line 32) overrides the first one (line 26).

D. The font size of the paragraphs is `40px` as both event listeners are valid and attachable directly to `document.body`.

Question 16

Consider the source code of Question 15 one more time. In line 28 the keyword `this` is used. Which object does `this` refer to when the user presses keys 1 or 2?

A. `window`

B. `document`

C. `document.body`

D. `p`

Question 17

Consider the web page source code below. Assume a user opens the web page in a modern browser. The user clicks on the `Add Paragraph` button once. In which place will `<p>New Paragraph</p>` be added?

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8" />
    <title>Adding a paragraph</title>
  </head>

  <body>
    <p>Paragraph 1</p>
    <p>Paragraph 2</p>

    <button class="button">Add Paragraph</button>

    <script>
      document.getElementsByClassName("button")[0].onclick = addParagraph;

      function addParagraph() {
        let p = document.createElement("p");
        p.innerHTML = "New Paragraph";
        document.body.appendChild(p);
      }
    </script>
  </body>
</html>
```

A. Immediately before `<p>Paragraph 1</p>`.

B. Immediately after `<p>Paragraph 2</p>`.

C. Immediately after `<button class="button">Add Paragraph</button>`.

D. Nowhere. The DOM will not change as the event listener is not actually attached to the `<button>` element.

Question 18

Given two files `bar.js` and `foo.js` as defined next, what is the output on the console after running `node foo.js`?

```
module.exports = function () { //bar.js
  return [
    function () {
      return "gold";
    }(),
    function () {
      return "silver";
    }(),
    function () {
      return "bronze";
    }(),
  ];
};
```

```
const medalsOld = require("./bar.js")(); //foo.js
console.log(medalsOld[0]);

medalsOld.push(function () {
  return "copper";
}());

const medalsNew = require("./bar.js")();
console.log(medalsNew[3]);
```

A. gold
   copper

B. undefined
   undefined

C. gold
   undefined

D. gold
   bronze

Question 19

Consider a Node.js script that is used to help developers of an e-commerce application to keep an eye on the number of entries in `pricelist.json` while they are at home. The file `pricelist.json` looks as follows (note that we here show three products with their prices, but the actual file contains many more):

```
[
  { "product": "Astro Bot", "price": "14.15" },
  { "product": "Wimbledon", "price": "29.90" },
  { "product": "Train Simulator", "price": "49.90" }
]
```

The Node.js script below is used to keep track of the number of entries in the price list. It is stored in a file called `pricelist-watcher.js` and started as follows on the server (which has IP address `8.8.8.8`): `node pricelist-watcher.js 4001`. The developers in turn connect to the server via telnet using the following command: `telnet 8.8.8.8 4001`. The idea is for them to receive the number of products approximately every five seconds as a printout on the terminal. What actually happens?

```
const fs = require("fs");
const net = require("net");

const file = "pricelist.json";
const port = process.argv[2];

const server = net.createServer(function (connection) {
  setInterval(function () {
    fs.readFile(file, function (err, data) {
      const entries = JSON.parse(data);
      connection.write(`Number of products in the system is ${entries.length}\n`);
    });
  }, 5000);

  connection.on("close", function () {});
});

server.listen(port, function () {
  console.log("Waiting for clients to connect.");
});
```

Note: assume that the file ***`pricelist.json`*** *contains exactly three products.*

A. Approximately five seconds after connecting, a single message is printed on the terminal `Number of products in the system is 3`. Afterwards no further messages are printed. This behavior is the same for all clients that connect to the server via telnet.

B. No message is printed on the terminal. This behavior is the same for all clients that connect to the server via telnet.

C. Approximately every five seconds, the message `Number of products in the system is 3` is printed on the terminal. This behavior is the same for all clients that connect to the server via telnet.

D. The behavior differs depending on when a client connects to the server. The first client that connects receives the message `Number of products in the system is 0` approximately every five seconds. Clients that connect later receive no message.

Question 20

Consider the following four statements about Ajax:

(1) Ajax is a Node.js mechanism that enables the dynamic loading of content without having to reload the page manually.

(2) Ajax revolves around the use of the `XMLHttpRequest` object to communicate between client and servers.

(3) Using Ajax, servers can send several HTTP responses in reply to a single HTTP request.

(4) Ajax reduces the HTTP request/response cycle overhead as a complete HTTP message is no longer required for the communication between client and server.

Which of these statements are TRUE?

A. Only 2)

B. Only 3) and 4)

C. All four statements are true.

D. None. All four statements are false.

Question 21

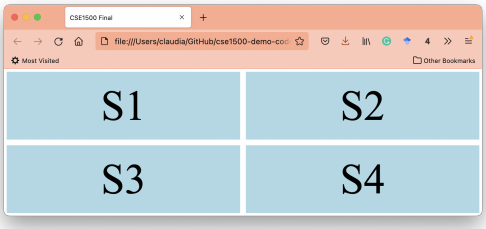
Consider the following HTML:

```
<!DOCTYPE html>
<html lang="en">

<head>
    <title>CSE1500 Final</title>
    <style>
        * {
            margin: 0;
        }

        #main {
            display: grid;
            grid-template-columns: 1fr 1fr;
            grid-template-rows: 1fr 1fr;
        }

        .square {
            position: relative;
            width: calc(50vw - 10px);
            height: calc(50vh - 10px);
            background-color: lightblue;
            color: black;
```

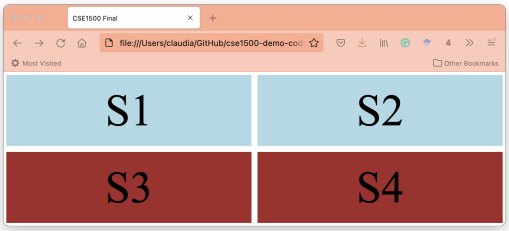
```
            border: 5px solid white;
            font-size: 75px;
            display: flex;
            align-items: center;
            justify-content: center;
        }

        div + .square {
            background-color: orange;
        }

        .square + .square {
            background-color: lightblue;
        }

        .square + .square + .square {
            background-color: brown;
        }
    </style>
</head>

<body>
    <div id="main">
        <div class="square odd">S1</div>
        <div class="square even">S2</div>
        <div class="square odd">S3</div>
        <div class="square even">S4</div>
    </div>
</body>

</html>
```

How will its rendering look like?

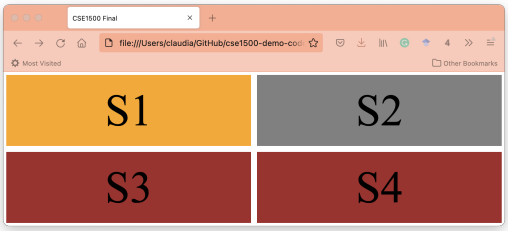
Q21: Answer A.



Q21: Answer B.



Q21: Answer C.



Q21: Answer D.

Question 22

Consider the HTML of Question 21 again. Which of the following selectors selects only the two `<div>` elements whose inner HTML is set to `S2` and `S4` respectively?

A. `div:nth-child(2n) { /* ... */ }`

B. `div:nth-of-type(2n+1) { /* ... */ }`

C. `div:nth-child(2n+1) { /* ... */ }`

D. `div:nth-of-type(2n) { /* ... */ }`

Question 23

Which of the following statements about pseudo-element `::before` is TRUE? *Note: in the answers that follow, we use* `<e>` *as a placeholder element.*

A. `e::before` creates a pseudo-element that is the first child of element `<e>`.

B. `e::before` creates a pseudo-element that is the first parent of `<e>`.

C. `e::before` creates a pseudo-element that is the first sibling of `<e>`.

D. `e::before` creates a pseudo-element that is the last child of `<e>`'s parent.

Question 24

Which of the following statements about DOM elements is FALSE?

A. A single DOM element can contain multiple classes.

B. A class attribute can be added to any DOM element.

C. The content attribute is required for block-level DOM elements.

D. Block-level DOM elements can be placed within other block-level DOM elements.

Question 25

Consider the following route handler:

```
app.get("/w(ho)+(hat)+am?i*", function (req, res) {
  res.end("ROUTE MATCHED!")
});
```

Assume that a Node.js script containing this route handler (and all other necessary boilerplate code) is started on the local machine, port 3000. Now consider these four URLs:

- `http://localhost:3000/whoamiii`
- `http://localhost:3000/whatai`
- `http://localhost:3000/whatamidoinghere`
- `http://localhost:3000/whowhatami`

How many of them match the route defined above?

A. 0

B. 1

C. 2

D. 3

Question 26

Which of the following statements about cookies is TRUE?

A. Secure cookies can only be sent via HTTPS, thus ensuring the cookies' encryption before transmission.

B. Transient cookies are also called permanent cookies and only exist in the browser's local storage.

C. If a cookie has no explicit expiration date, it is a persistent cookie.

D. Third-party cookies are cookies that belong to a subdomain of the domain that is one shown in the browser's address bar.

Question 27

Consider the following Node.js script:

```
1  const express = require("express");
2
3  function logger(request, response, next) {
4    console.log(`${new Date()}, ${request.method}, ${request.url}`);
5    next();
6  }
7
8  function changeRequest(request, response, next){
9    request.query.logged="true"
10   next()
11 }
12
13 function delimiter(request, response, next) {
14   console.log("------------------");
15   next();
16 }
17
18 const app = express();
19 app.use(logger);
20 app.use(changeRequest)
21 app.use(delimiter);
22 app.get("*", function(req, res){
23   console.log(req.query)
24   res.end("Request received. Logging status: "+req.query.logged)
25 })
26 app.listen(3001);
```

Assume the script was started on `localhost`. Entering the URL `http://localhost:3001/hello` into the browser's address bar and pressing `Enter` has which effect?

A. The browser displays *Request received. Logging status: true.*

B. The browser displays *Request received. Logging status: undefined* as it is not possible to change a `request` object as done here in line 9.

C. The browser displays nothing. The request is sent to the server but the server never sends a response due to the incorrect use of the `next()` function.

D. The browser displays nothing. The request is sent to the server but there is no route defined for path `/hello`.

Question 28

Assume that a server sent a number of cookies to a client in response to the client's HTTP request. Now another HTTP request by the same client is received and the server wants to clear all cookies from the client. How can this be achieved?

A. The server includes the header `Set-Cookie: clear` in the HTTP response. This informs the browser that all cookies previously sent by the server are invalid and need to be removed from its cookie storage.

B. The server includes the header `Clear-Cookies: true` in the HTTP response. This informs the browser that all cookies previously sent by the server are invalid and need to be removed from its cookie storage.

C. For each cookie the server wants to clear from the client, the server sends the cookie to the client with an expiration date in the past. This informs the browser that this cookie has become invalid and the browser deletes the cookie from its cookie storage.

D. The server has no possibility to clear any of the cookies it sent to the client. The client always respects the original expiration date (as set by the server when the cookie was first sent to the client) of each cookie.

Question 29

What is the purpose of signing a cookie with an HMAC string?

A. The cookie is now encrypted.

B. The cookie is now compressed.

C. Tampering with the cookie value can be detected.

D. Tampering with the cookie's flags can be detected.

Question 30

How many of the following statements should be followed in order to ensure secure authentication and session management?

(1) Session IDs should not have a timeout.

(2) Session IDs should not be visible in URLs.

(3) Session IDs should be invalidated on the server after the user ends the session.

(4) Session IDs should not be reused.

A. 1

B. 2

C. 3

D. 4

END OF EXAM.