Instituto Superior de Engenharia de Lisboa

Licenciatura em Engenharia Informática e de Computadores e Licenciatura em Engenharia Informática, Redes e Telecomunicações

Introdução à Programação na Web / Programação na Internet

Inverno de 2023/2024

* Teste Global de Época de Recurso (25 de Janeiro de 2024) - Duração 1h30

NÚMERO _____ **NOME** _____

GRUPO 1 [4 valores]

Nas seguintes questões, selecione a única opção verdadeira. Uma resposta correta conta a totalidade da cotação da pergunta; uma resposta incorreta desconta 25% da cotação da pergunta ao total do grupo; uma questão não respondida conta 0 valores.

1. Qual destes mecanismos é suportado pela linguagem JavaScript?
   A. Funções de ordem superior
   B. Closures
   C. Protótipos
   D. Todas as anteriores

2. Ao analisar a URL *http://www.gloriososlb.com/players?name=picanhas&number=9#biography*, qual das seguintes afirmações é verdadeira?
   A. O path neste URL é *www.gloriososlb.com*
   B. A query string neste URL é *biography*
   C. O esquema usado é HTTPS
   D. Nenhuma das opções anteriores

3. Ao lidar com um pedido para um recurso que o cliente tem permissões para aceder, contudo não apresenta credenciais válidas, o *status code* que deve ser retornado é:
   A. 400
   B. 403
   C. 404
   D. 401

4. Qual das seguintes afirmações, relativas aos métodos HTTP, é verdadeira:
   A. O método POST é um método *safe* e idempotente
   B. O método GET não é *safe*, mas é idempotente
   C. O método PUT não é safe, mas é idempotente
   D. O método DELETE não é *safe*, nem idempotente

5. Qual das seguintes afirmações descreve corretamente a Fetch API em JavaScript?
   A. A Fetch API é usada para realizar operações de manipulação de DOM
   B. A Fetch API é exclusivamente projetada para interações assíncronas com bases de dados
   C. A Fetch API fornece uma interface para enviar pedidos HTTP e receber respostas de forma assíncrona
   D. A Fetch API é usada para realizar transições em páginas web

6. Qual das seguintes afirmações melhor descreve o Node.js?
   A. Node.js é uma versão da linguagem JavaScript
   B. Node.js é um ambiente de execução JavaScript
   C. Node.js é uma base de dados documental
   D. Node.js é uma biblioteca para desenvolvimento em JavaScript

7. Qual é o principal propósito do header `Authorization` do protocolo HTTP?
   A. Especificar o tipo de conteúdo enviado nos pedidos HTTP.
   B. Fornecer credenciais de autenticação do utilizador
   C. Enviar um cookie do servidor para o utilizador
   D. Indicar ao browser como apresentar a página web

NÚMERO _____ **NOME** _____

GRUPO 2 [4 valores]

Faça a correspondência entre o que está no lado esquerdo da tabela (identificado com números) e o que lhe está relacionado no lado direito da tabela (identificado por letras). Se por exemplo considerar que a opção 1 do lado esquerdo está relacionada com a opção c do lado direito, responda 1-c. Cada elemento da coluna da esquerda só tem **uma** correspondência da coluna da direita, e vice-versa.
Uma resposta correta conta a totalidade da cotação da opção; uma resposta incorreta desconta 25% da cotação da opção ao total da questão; uma opção não respondida conta 0 valores.

1. Em JavaScript:

1	Uma arrow function	**a**	retorna sempre uma Promise
2	Uma função é *callback* se	**b**	é passada como argumento a outra função
3	Uma função regular sem 'async'	**c**	é sempre anónima
4	Uma função definida como 'async'	**d**	é executada sem pausas até retornar

2. Relativamente aos *Headers* do Protocolo HTTP:

1	Content-Type	**a**	Apenas pode ser enviado nos pedidos
2	Cookie	**b**	Pode ser enviado nos pedidos e nas respostas
3	Location	**c**	Apenas pode ser enviado nos respostas
4	Set-Cookie	**d**	Serve guardar informação do servidor no cliente

3. No contexto do Handlebars:

1	A expressão `@index`	**a**	tem de ser utilizada dentro de um bloco {{#each}}
2	A expressão `{{#unless}}`	**b**	permite alterar o contexto do bloco
3	A expressão `{{#with}}`	**c**	permite iterar sobre uma lista
4	A expressão `{{#each}}`	**d**	permite renderizar um bloco condicionalmente

4. Relativamente ao mecanismo de Promises do JavaScript:

1	O método `then()`	**a**	retorna uma promise que é resolvida quando todas as promises do argumento passado tenham sido resolvidas
2	O método `catch()`	**b**	executa o *callback* quando a promise é rejeitada
3	O método `finally()`	**c**	executa o *callback* quando a promise é resolvida com sucesso
4	O método `Promise.all()`	**d**	executa o *callback* independentemente do resultado da promise

5. Em HTML:

1	O elemento `head`	**a**	tem de ter um valor único num documento
2	O elemento `form`	**b**	contém metadados sobre o documento, incluindo título e links para scripts e folhas de estilos
3	O atributo `id`	**c**	pode ser partilhado por múltiplos elementos
4	O atributo `class`	**d**	pode conter múltiplos elementos `input`

6. No contexto do express.js e do passport.js:

1	Um middleware	**a**	associa um ou mais middlewares ao caminho especificado
2	`express.urlencoded()` é um middleware que	**b**	auxilia na autenticação de uma aplicação em Node.js
3	`app.use()` é uma função que	**c**	tem acesso aos objetos que representam o pedido e resposta HTTP
4	O Passport é um middleware que	**d**	faz parse dos pedidos recebidos

Instituto Superior de Engenharia de Lisboa

Licenciatura em Engenharia Informática e de Computadores e Licenciatura em Engenharia Informática, Redes e Telecomunicações

Introdução à Programação na Web / Programação na Internet

Inverno de 2023/2024

** Teste Global de Época de Recurso (25 de Janeiro de 2024) - Duração 1h30

NÚMERO _____ **NOME** _____

GRUPO 1 [4 valores]

Nas seguintes questões, selecione a única opção verdadeira. Uma resposta correta conta a totalidade da cotação da pergunta; uma resposta incorreta desconta 25% da cotação da pergunta ao total do grupo; uma questão não respondida conta 0 valores.

1. Qual é o principal propósito do header `Authorization` do protocolo HTTP?
   A. Especificar o tipo de conteúdo enviado nos pedidos HTTP.
   B. Fornecer credenciais de autenticação do utilizador
   C. Enviar um cookie do servidor para o utilizador
   D. Indicar ao browser como apresentar a página web

2. Ao lidar com um pedido para um recurso que o cliente tem permissões para aceder, contudo não apresenta credenciais válidas, o *status code* que deve ser retornado é:
   A. 400
   B. 403
   C. 404
   D. 401

3. Qual das seguintes afirmações descreve corretamente a Fetch API em JavaScript?
   A. A Fetch API é usada para realizar operações de manipulação de DOM
   B. A Fetch API é exclusivamente projetada para interações assíncronas com bases de dados
   C. A Fetch API fornece uma interface para enviar pedidos HTTP e receber respostas de forma assíncrona
   D. A Fetch API é usada para realizar transições em páginas web

4. Qual destes mecanismos é suportado pela linguagem JavaScript?
   A. Funções de ordem superior
   B. *Closures*
   C. Protótipos
   D. Todas as anteriores

5. Qual das seguintes afirmações, relativas aos métodos HTTP, é verdadeira:
   A. O método POST é um método *safe* e idempotente
   B. O método GET não é *safe*, mas é idempotente
   C. O método PUT não é safe, mas é idempotente
   D. O método DELETE não é *safe*, nem idempotente

6. Ao analisar a URL *http://www.gloriososlb.com/players?name=picanhas&number=9#biography*, qual das seguintes afirmações é verdadeira?
   A. O path neste URL é *www.gloriososlb.com*
   B. A query string neste URL é *#biography*
   C. O esquema usado é HTTPS
   D. Nenhuma das opções anteriores

7. Qual das seguintes afirmações melhor descreve o Node.js?
   A. Node.js é uma versão da linguagem JavaScript
   B. Node.js é um ambiente de execução JavaScript
   C. Node.js é uma base de dados documental
   D. Node.js é uma biblioteca para desenvolvimento em JavaScript

Instituto Superior de Engenharia de Lisboa

Licenciatura em Engenharia Informática e de Computadores e Licenciatura em Engenharia Informática, Redes e Telecomunicações

Introdução à Programação na Web / Programação na Internet

Inverno de 2023/2024

** Teste Global de Época de Recurso (25 de Janeiro de 2024) - Duração 1h30

NÚMERO _____ **NOME** _____

GRUPO 2 [4 valores]

Faça a correspondência entre o que está no lado esquerdo da tabela (identificado com números) e o que lhe está relacionado no lado direito da tabela (identificado por letras). Se por exemplo considerar que a opção 1 do lado esquerdo está relacionada com a opção c do lado direito, responda 1-c. Cada elemento da coluna da esquerda só tem **uma** correspondência da coluna da direita, e vice-versa.
Uma resposta correta conta a totalidade da cotação da opção; uma resposta incorreta desconta 25% da cotação da opção ao total da questão; uma opção não respondida conta 0 valores.

1. Relativamente aos *Headers* do Protocolo HTTP:

1	Content-Type	**a**	Apenas pode ser enviado nos pedidos
2	Cookie	**b**	Pode ser enviado nos pedidos e nas respostas
3	Location	**c**	Apenas pode ser enviado nos respostas
4	Set-Cookie	**d**	Serve guardar informação do servidor no cliente

2. No contexto do Handlebars:

1	A expressão `@index`	**a**	tem de ser utilizada dentro de um bloco {{#each}}
2	A expressão `{{#unless}}`	**b**	permite alterar o contexto do bloco
3	A expressão `{{#with}}`	**c**	permite iterar sobre uma lista
4	A expressão `{{#each}}`	**d**	permite renderizar um bloco condicionalmente

3. Relativamente ao mecanismo de Promises do JavaScript:

1	O método `then()`	**a**	retorna uma promise que é resolvida quando todas as promises do argumento passado tenham sido resolvidas
2	O método `catch()`	**b**	executa o *callback* quando a promise é rejeitada
3	O método `finally()`	**c**	executa o *callback* quando a promise é resolvida com sucesso
4	O método `Promise.all()`	**d**	executa o *callback* independentemente do resultado da promise

4. Em HTML:

1	O elemento `head`	**a**	tem de ter um valor único num documento
2	O elemento `form`	**b**	contém metadados sobre o documento, incluindo título e links para scripts e folhas de estilos
3	O atributo `id`	**c**	pode ser partilhado por múltiplos elementos
4	O atributo `class`	**d**	pode conter múltiplos elementos `input`

5. No contexto do express.js e do passport.js:

1	Um middleware	**a**	associa um ou mais middlewares ao caminho especificado
2	`express.urlencoded()` é um middleware que	**b**	auxilia na autenticação de uma aplicação em Node.js
3	`app.use()` é uma função que	**c**	tem acesso aos objetos que representam o pedido e resposta HTTP
4	O Passport é um middleware que	**d**	faz parse dos pedidos recebidos

6. Em JavaScript:

1	Uma arrow function	**a**	retorna sempre uma Promise
2	Uma função é *callback* se	**b**	é passada como argumento a outra função
3	Uma função regular sem 'async'	**c**	é sempre anónima
4	Uma função definida como 'async'	**d**	é executada sem pausas até retornar

Instituto Superior de Engenharia de Lisboa

Licenciatura em Engenharia Informática e de Computadores e Licenciatura em Engenharia Informática, Redes e Telecomunicações

Introdução à Programação na Web / Programação na Internet

Inverno de 2023/2024

*** Teste Global de Época de Recurso (25 de Janeiro de 2024) - Duração 1h30

NÚMERO _____ **NOME** _____

GRUPO 1 [4 valores]

Nas seguintes questões, selecione a única opção verdadeira. Uma resposta correta conta a totalidade da cotação da pergunta; uma resposta incorreta desconta 25% da cotação da pergunta ao total do grupo; uma questão não respondida conta 0 valores.

1. Ao analisar a URL *http://www.gloriososlb.com/players?name=picanhas&number=9#biography*, qual das seguintes afirmações é verdadeira?
   A. O esquema usado é HTTPS
   B. A query string neste URL é *#biography*
   C. O path neste URL é *www.gloriososlb.com*
   D. Nenhuma das opções anteriores

2. Qual das seguintes afirmações melhor descreve o Node.js?
   A. Node.js é uma biblioteca para desenvolvimento em JavaScript
   B. Node.js é uma versão da linguagem JavaScript
   C. Node.js é um ambiente de execução para JavaScript
   D. Node.js é uma base de dados documental

3. Qual destes mecanismos é suportado pela linguagem JavaScript?
   A. Funções de ordem superior
   B. *Closures*
   C. Protótipos
   D. Todas as anteriores

4. Ao lidar com um pedido para um recurso que o cliente tem permissões para aceder, contudo não apresenta credenciais válidas, o *status code* que deve ser retornado é:
   A. 404
   B. 400
   C. 401
   D. 403

5. Qual das seguintes afirmações, relativas aos métodos HTTP, é verdadeira:
   A. O método PUT não é safe, mas é idempotente
   B. O método POST é um método *safe* e idempotente
   C. O método GET não é *safe*, mas é idempotente
   D. O método DELETE não é *safe*, nem idempotente

6. Qual das seguintes afirmações descreve corretamente a Fetch API em JavaScript?
   A. A Fetch API é usada para realizar transições em páginas web
   B. A Fetch API fornece uma interface para enviar pedidos HTTP e receber respostas de forma assíncrona
   C. A Fetch API é usada para realizar operações de manipulação de DOM
   D. A Fetch API é exclusivamente projetada para interações assíncronas com bases de dados

7. Qual é o principal propósito do header `Authorization` do protocolo HTTP?
   A. Fornecer credenciais de autenticação do utilizador
   B. Indicar ao browser como apresentar a página web
   C. Especificar o tipo de conteúdo enviado nos pedidos HTTP.
   D. Enviar um cookie do servidor para o utilizador

NÚMERO _____ **NOME** _____

GRUPO 2 [4 valores]

Faça a correspondência entre o que está no lado esquerdo da tabela (identificado com números) e o que lhe está relacionado no lado direito da tabela (identificado por letras). Se por exemplo considerar que a opção 1 do lado esquerdo está relacionada com a opção c do lado direito, responda 1-c. Cada elemento da coluna da esquerda só tem **uma** correspondência da coluna da direita, e vice-versa.
Uma resposta correta conta a totalidade da cotação da opção; uma resposta incorreta desconta 25% da cotação da opção ao total da questão; uma opção não respondida conta 0 valores.

1. Em HTML:

1	O elemento `head`	**a**	tem de ter um valor único num documento
2	O elemento `form`	**b**	contém metadados sobre o documento, incluindo título e links para scripts e folhas de estilos
3	O atributo `id`	**c**	pode ser partilhado por múltiplos elementos
4	O atributo `class`	**d**	pode conter múltiplos elementos `input`

2. Relativamente aos *Headers* do Protocolo HTTP:

1	Content-Type	**a**	Apenas pode ser enviado nos pedidos
2	Cookie	**b**	Pode ser enviado nos pedidos e nas respostas
3	Location	**c**	Apenas pode ser enviado nos respostas
4	Set-Cookie	**d**	Serve guardar informação do servidor no cliente

3. Em JavaScript:

1	Uma arrow function	**a**	retorna sempre uma Promise
2	Uma função é *callback* se	**b**	é passada como argumento a outra função
3	Uma função regular sem 'async'	**c**	é sempre anónima
4	Uma função definida como 'async'	**d**	é executada sem pausas até retornar

4. Relativamente ao mecanismo de Promises do JavaScript:

1	O método `then()`	**a**	retorna uma promise que é resolvida quando todas as promises do argumento passado tenham sido resolvidas
2	O método `catch()`	**b**	executa o *callback* quando a promise é rejeitada
3	O método `finally()`	**c**	executa o *callback* quando a promise é resolvida com sucesso
4	O método `Promise.all()`	**d**	executa o *callback* independentemente do resultado da promise

5. No contexto do express.js e do passport.js:

1	Um middleware	**a**	associa um ou mais middlewares ao caminho especificado
2	`express.urlencoded()` é um middleware que	**b**	auxilia na autenticação de uma aplicação em Node.js
3	`app.use()` é uma função que	**c**	tem acesso aos objetos que representam o pedido e resposta HTTP
4	O Passport é um middleware que	**d**	faz parse dos pedidos recebidos

6. No contexto do Handlebars:

1	A expressão `@index`	**a**	tem de ser utilizada dentro de um bloco {{#each}}
2	A expressão `{{#unless}}`	**b**	permite alterar o contexto do bloco
3	A expressão `{{#with}}`	**c**	permite iterar sobre uma lista
4	A expressão `{{#each}}`	**d**	permite renderizar um bloco condicionalmente

---

GRUPO 3 [6 valores]

1. [4] Considere o seguinte código em JavaScript (em acima) e duas as listagens da sua utilização (em baixo).

```
1  function waitFor(obj, timeMs = 2000) {
2   return new Promise((resolve, reject) => {
3     if (obj)
4         return setTimeout(
5            () => { resolve(obj) }, timeMs)
6       return reject('Invalid object')
7   })
8  }
9
```

```
function success(obj) {
  console.log(`Success "${obj}"`)
}

function error(err) {
  console.log(`Error "${err}"`)
}
```

```
// Listagem 1
await waitFor("SLB")
    .then(success)
    .catch(error)
await waitFor()
    .then(success)
    .catch(error)
```

```
// Listagem 2
async function test() {
   for (let i = 0; i < 3; i++) {
       waitFor(`Time elapsed ${i}`).
          then(console.log)
   }
}
async function test1() {
  for (let i = 0; i < 3; i++) {
     const msg =
        await waitFor(`Time elapsed ${i}`)
     console.log(msg)
  }
}
```

   a. [1.5] Indique qual o resultado na consola da execução da listagem 1, bem como o tempo (aproximado) em que as mensagens aparecem.

   b. [1.5] Que diferenças existiriam se os 2 await fossem retirados da listagem 1. Justifique num máximo de 2 linhas.

   c. [1.5] Indique qual o resultado na consola da execução da listagem 2, bem como o tempo (aproximado) em que as mensagens aparecem.

2. [1.5] Considere o seguinte código em JavaScript. O que aparece na consola quando este for executado?

```
1  function f(str) {
2     if(str == "SLB") { return "Glorioso" }
3     console.log("Nothing to say....")
4  }
5  async function f1(str) { return f(str) }
```

```
console.log(f("SLB"))
console.log(f("Other"))
console.log("-----------------")
console.log(f1("SLB"))
console.log(f1("Other"))
```

GRUPO 4 [6 valores]

1. [6] Pretende-se implementar um *middleware* express que processa pedidos com *body* no formato CSV (*Comma Separated Values*). Este formato tem o `Content-Type application/csv`. Este, coloca na propriedade `body` do objeto `Request`, um objeto com a propriedade `lines`, que contém um *array* de *arrays*, em que cada *array* corresponde a uma linha do csv e cada posição nesse *array* contém o valor de cada uma das colunas nessa linha. Se o parâmetro `hasColumnRow` for `true`, considera a 1ª linha como contendo o nome das colunas, que fica na propriedade `columns`.

csv-mw.mjs

```
1  export default function (hasColumnsRow = false) {
2     return function middleware(req, rsp, next) {
3         if (!req.get("Content-Type").includes(_________________))  // 1
4             return __________   // 2
5
6         getRequestBody().then(__________)   // 3
7
8         // Returns a Promise that is resolved with the text of the Request body
9         function getRequestBody() { . . . }
10        function createBodyObj(rawBody) {
11            let lines = rawBody.split(/\n/).map(s => s.split(_____).map(s => s.trim())) // 4
12            let obj = req.body = {}
13            if (hasColumnsRow) {
14                obj.columns = __________   // 5
15                lines = lines.slice(1)
16            }
17            __________ = lines // 6
18            next()
19        }
20    }
21 }
```

server.mjs

```
1  import express from "express";
2  import csv from "./csv-mw.mjs"
3  const app = express()
4
5  app.use(__________)  // 1
6  app._____("/process-csv", processCsv) // 2
7  const PORT = 1904
8  app.listen(PORT, ()=> console.log(`Server listening in port ${PORT}`))
9
10 function processCsv(req, rsp) {
11  rsp.___________(req.body)            // 3
12 }
13
```

a. [3] Complete a implementação do módulo csv-mw.mjs.

b. [1.5] Complete a implementação do módulo server.mjs.

c. [1.5] Apresente o pedido HTTP que provoca a seguinte resposta, no formato JSON.

```
{ "columns": [ "Num", "Nome", "Nota"],
  "lines": [
    [ "123", "Joao","12"],
    [ "234", "Ana", "12"]
  ]
}
```

Os docentes,
Luís Falcão e Valderi Leithardt