```
abstract class Filter<T extends Number>{
    public abstract boolean isValid(T a);
}
```
6406532306834. ✗

```
interface Filter<T extends Number>{
    public boolean isValid(T a);
}
```
6406532306835. ✓

```
interface Filter<T extends Number>{
    public abstract boolean isValid(T a);
    public abstract boolean predicate(T a);
}
```
6406532306836. ✗

AppDev2

Section Id :	64065348509
Section Number :	11
Section type :	Online
Mandatory or Optional :	Mandatory
Number of Questions :	17
Number of Questions to be attempted :	17
Section Marks :	50
Display Number Panel :	Yes
Group All Questions :	No
Enable Mark as Answered Mark for Review and Clear Response :	Yes
Maximum Instruction Time :	0
Sub-Section Number :	1
Sub-Section Id :	640653100852
Question Shuffling Allowed :	No
Is Section Default? :	null

Question Number : 161 Question Id : 640653689578 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 0

Question Label : Multiple Choice Question

THIS IS QUESTION PAPER FOR THE SUBJECT "DIPLOMA LEVEL : MODERN APPLICATION DEVELOPMENT II (COMPUTER BASED EXAM)"

ARE YOU SURE YOU HAVE TO WRITE EXAM FOR THIS SUBJECT?
CROSS CHECK YOUR HALL TICKET TO CONFIRM THE SUBJECTS TO BE WRITTEN.

(IF IT IS NOT THE CORRECT SUBJECT, PLS CHECK THE SECTION AT THE TOP FOR THE SUBJECTS REGISTERED BY YOU)

Options :

6406532306837. ✓ YES

6406532306838. ✗ NO

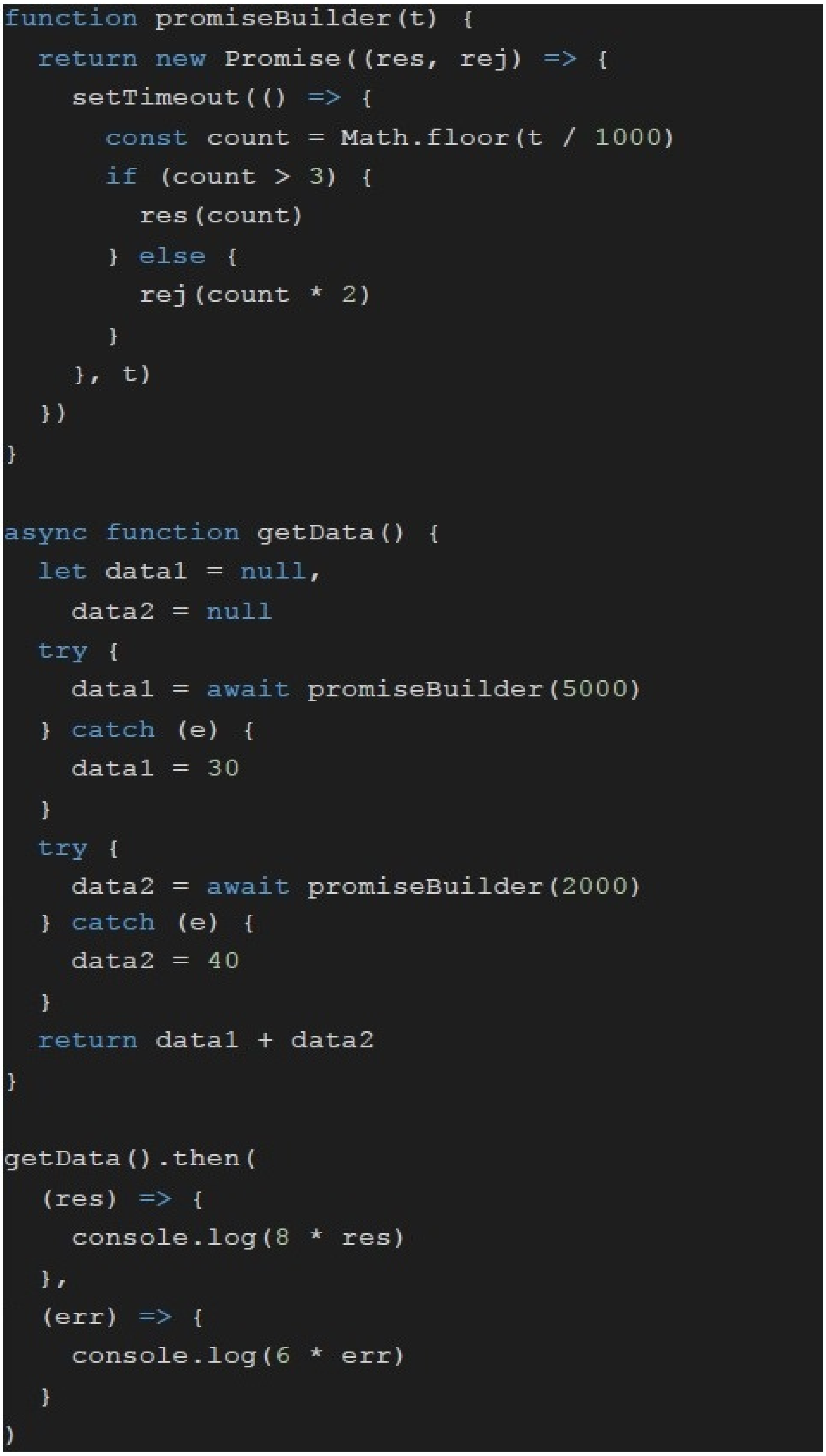
Sub-Section Number :	2
Sub-Section Id :	640653100853
Question Shuffling Allowed :	Yes
Is Section Default? :	null

Question Number : 162 Question Id : 640653689579 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 3

Question Label : Multiple Choice Question

Consider the following JavaScript code.



```
function promiseBuilder(t) {
  return new Promise((res, rej) => {
    setTimeout(() => {
      const count = Math.floor(t / 1000)
      if (count > 3) {
        res(count)
      } else {
        rej(count * 2)
      }
    }, t)
  })
}

async function getData() {
  let data1 = null,
    data2 = null
  try {
    data1 = await promiseBuilder(5000)
  } catch (e) {
    data1 = 30
  }
  try {
    data2 = await promiseBuilder(2000)
  } catch (e) {
    data2 = 40
  }
  return data1 + data2
}

getData().then(
  (res) => {
    console.log(8 * res)
  },
  (err) => {
    console.log(6 * err)
  }
)
```

What will be logged on to the console?

Options :

6406532306839. ✖ 56

6406532306840. ✖ 272

6406532306841.

✓ 360

6406532306842. ✗ 560

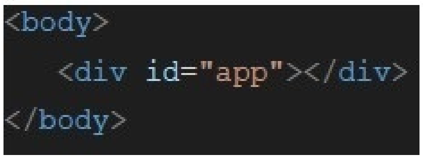
Question Number : 163 Question Id : 640653689581 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 3

Question Label : Multiple Choice Question

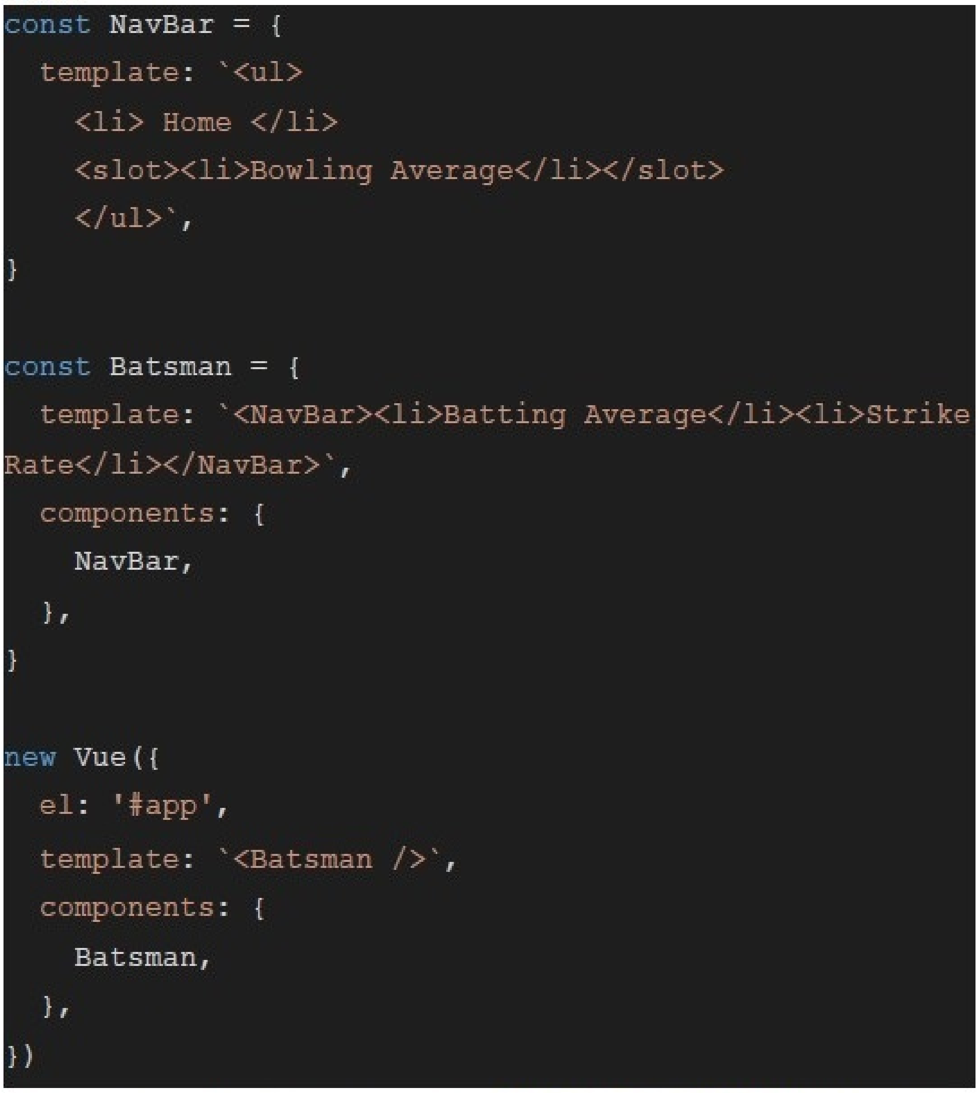
Consider the Vue application with markup index.html and JavaScript app.js

index.html



```
<body>
   <div id="app"></div>
</body>
```

app.js



```
const NavBar = {
  template: `<ul>
    <li> Home </li>
    <slot><li>Bowling Average</li></slot>
    </ul>`,
}

const Batsman = {
  template: `<NavBar><li>Batting Average</li><li>Strike
Rate</li></NavBar>`,
  components: {
    NavBar,
  },
}

new Vue({
  el: '#app',
  template: `<Batsman />`,
  components: {
    Batsman,
  },
})
```

Suppose the application is running on "http://localhost:8080" . What will be rendered by the browser?

Options :

6406532306847. ✖ ● Home

● Bowling Average

6406532306848. ✖ ● Home

● Bowling Average

● Batting Average

6406532306849. ✓ ● Home

● Batting Average

● Strike Rate

6406532306850. ✖ ● Bowling Average

● Batting Average

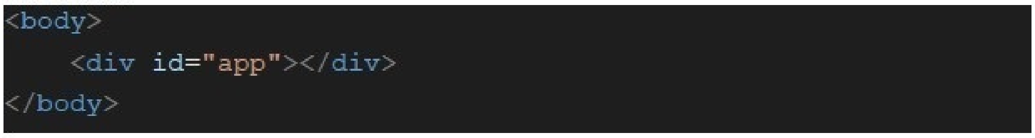
Question Number : 164 Question Id : 640653689583 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 3

Question Label : Multiple Choice Question

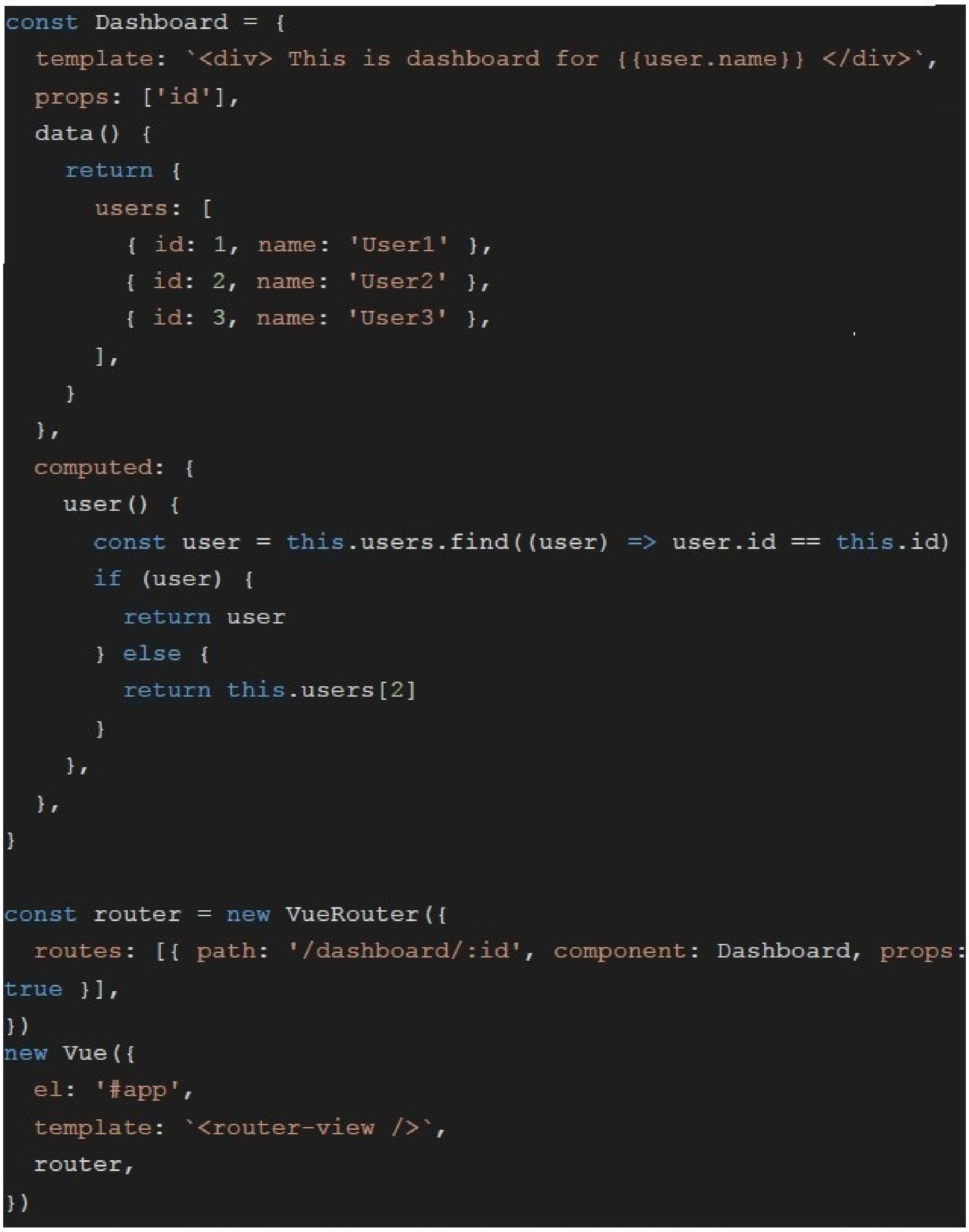
Consider the following Vue app with markup index.html and JavaScript app.js.

index.html



```
<body>
    <div id="app"></div>
</body>
```

app.js



```
const Dashboard = {
  template: `<div> This is dashboard for {{user.name}} </div>`,
  props: ['id'],
  data() {
    return {
      users: [
        { id: 1, name: 'User1' },
        { id: 2, name: 'User2' },
        { id: 3, name: 'User3' },
      ],
    }
  },
  computed: {
    user() {
      const user = this.users.find((user) => user.id == this.id)
      if (user) {
        return user
      } else {
        return this.users[2]
      }
    },
  },
}

const router = new VueRouter({
  routes: [{ path: '/dashboard/:id', component: Dashboard, props:
true }],
})
new Vue({
  el: '#app',
  template: `<router-view />`,
  router,
})
```

Suppose the application is running on "http://localhost:8080" . What will be rendered inside the router-view component of root component for the URL "http://127.0.0.1:8080/#/dashboard/8" ?

Options :

6406532306855. ✖ This is dashboard for User1

6406532306856. ✖ This is dashboard for User2

6406532306857. ✔ This is dashboard for User3

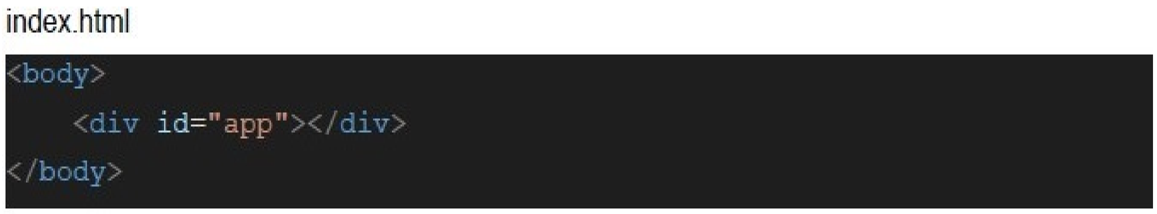
6406532306858. ✖ None of these

Question Number : 165 Question Id : 640653689586 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 3

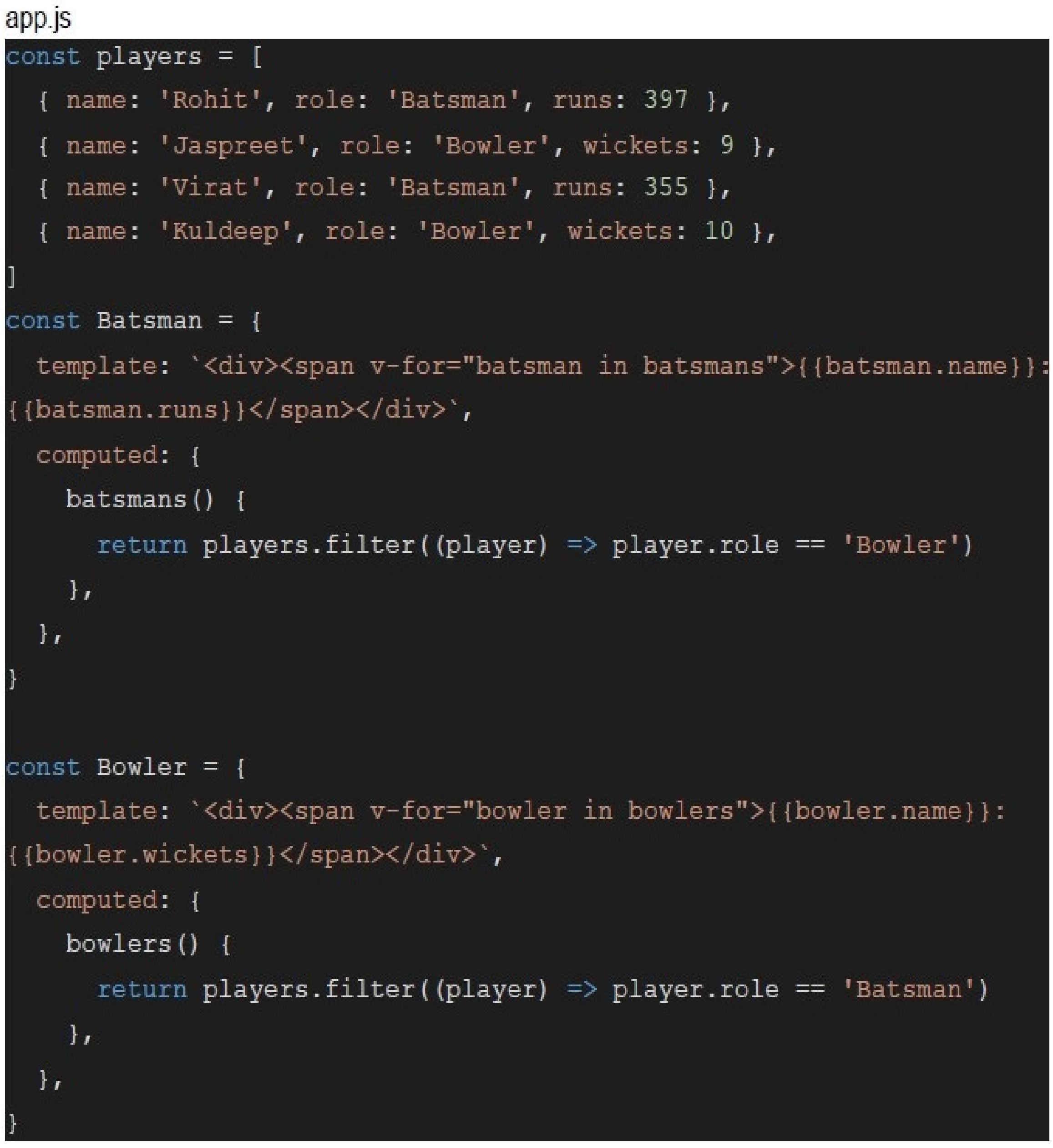
Question Label : Multiple Choice Question

Consider the following Vue application with JavaScript app.js and markup index.html



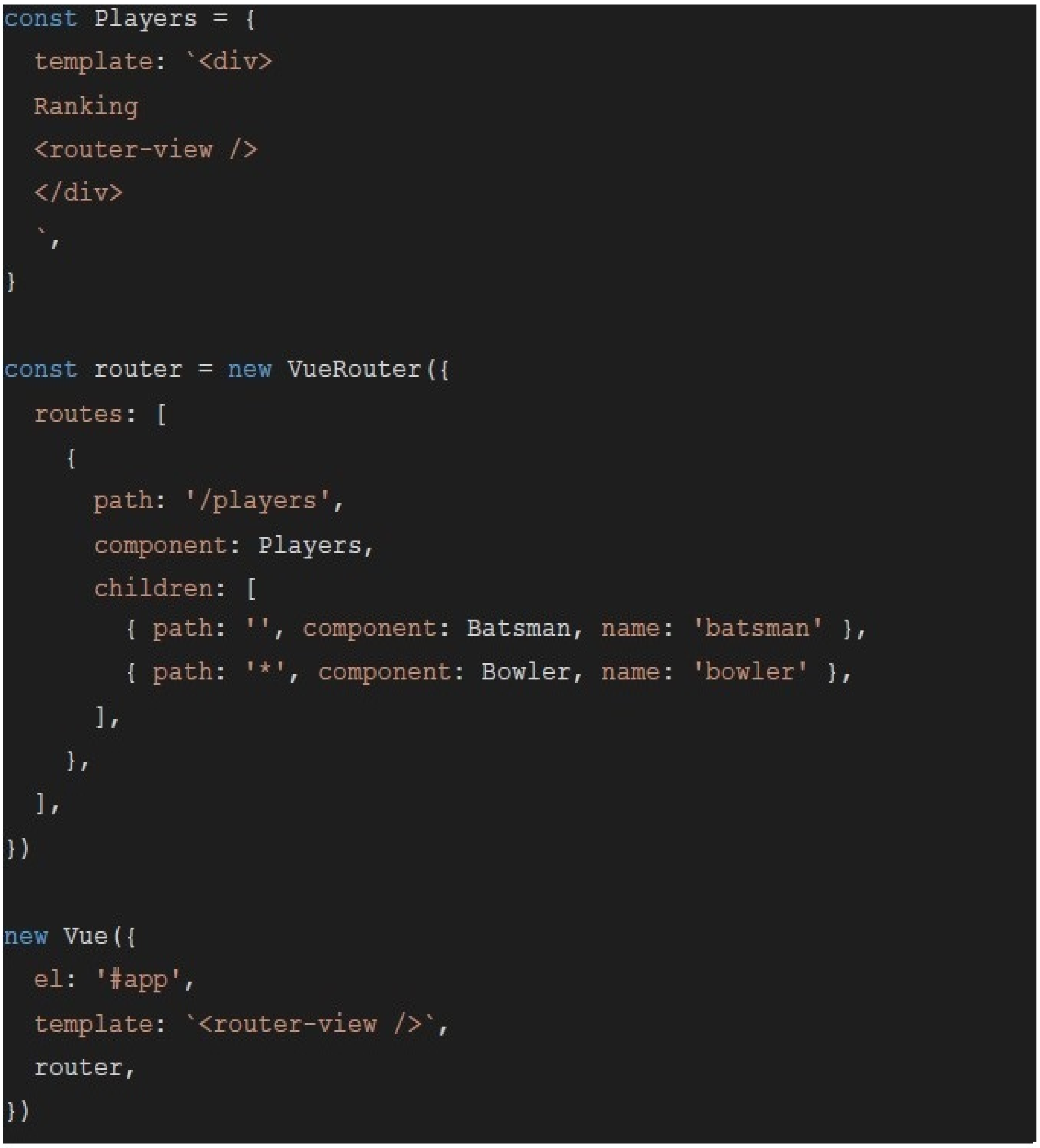
index.html

```html
<body>
    <div id="app"></div>
</body>
```

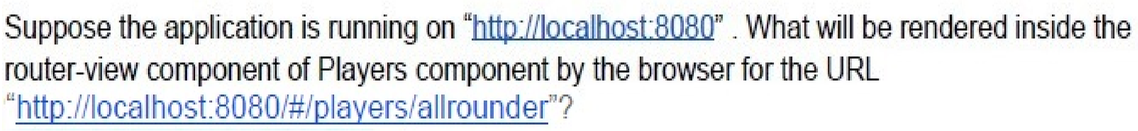


app.js

```javascript
const players = [
  { name: 'Rohit', role: 'Batsman', runs: 397 },
  { name: 'Jaspreet', role: 'Bowler', wickets: 9 },
  { name: 'Virat', role: 'Batsman', runs: 355 },
  { name: 'Kuldeep', role: 'Bowler', wickets: 10 },
]
const Batsman = {
  template: `<div><span v-for="batsman in batsmans">{{batsman.name}}:
{{batsman.runs}}</span></div>`,
  computed: {
    batsmans() {
      return players.filter((player) => player.role == 'Bowler')
    },
  },
}

const Bowler = {
  template: `<div><span v-for="bowler in bowlers">{{bowler.name}}:
{{bowler.wickets}}</span></div>`,
  computed: {
    bowlers() {
      return players.filter((player) => player.role == 'Batsman')
    },
  },
}
```



```
const Players = {
  template: `<div>
  Ranking
  <router-view />
  </div>
  `,
}

const router = new VueRouter({
  routes: [
    {
      path: '/players',
      component: Players,
      children: [
        { path: '', component: Batsman, name: 'batsman' },
        { path: '*', component: Bowler, name: 'bowler' },
      ],
    },
  ],
})

new Vue({
  el: '#app',
  template: `<router-view />`,
  router,
})
```



Suppose the application is running on "http://localhost:8080" . What will be rendered inside the router-view component of Players component by the browser for the URL "http://localhost:8080/#/players/allrounder"?

Options :

6406532306867. ✗ Jaspreet: Kuldeep:

6406532306868. ✗ Jaspreet: 9 Kuldeep: 10

6406532306869. ✓ Rohit: Virat:

6406532306870. ✗ Rohit:397 Virat:355

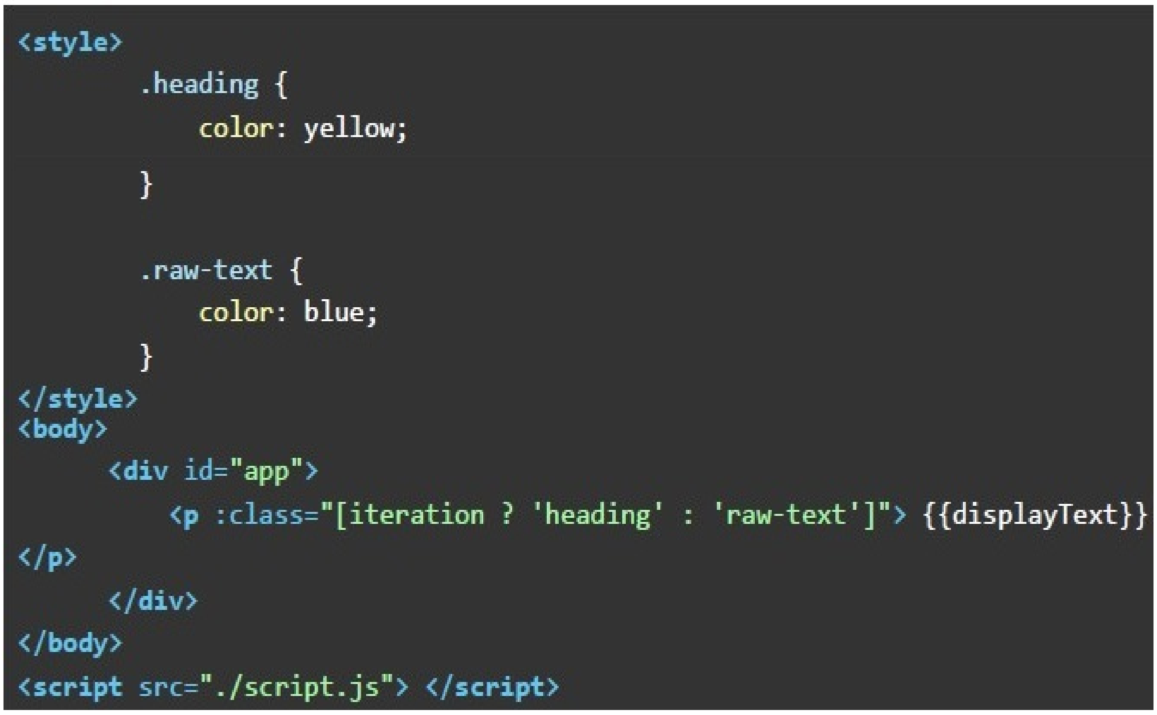
Question Number : 166 Question Id : 640653689588 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 3

Question Label : Multiple Choice Question

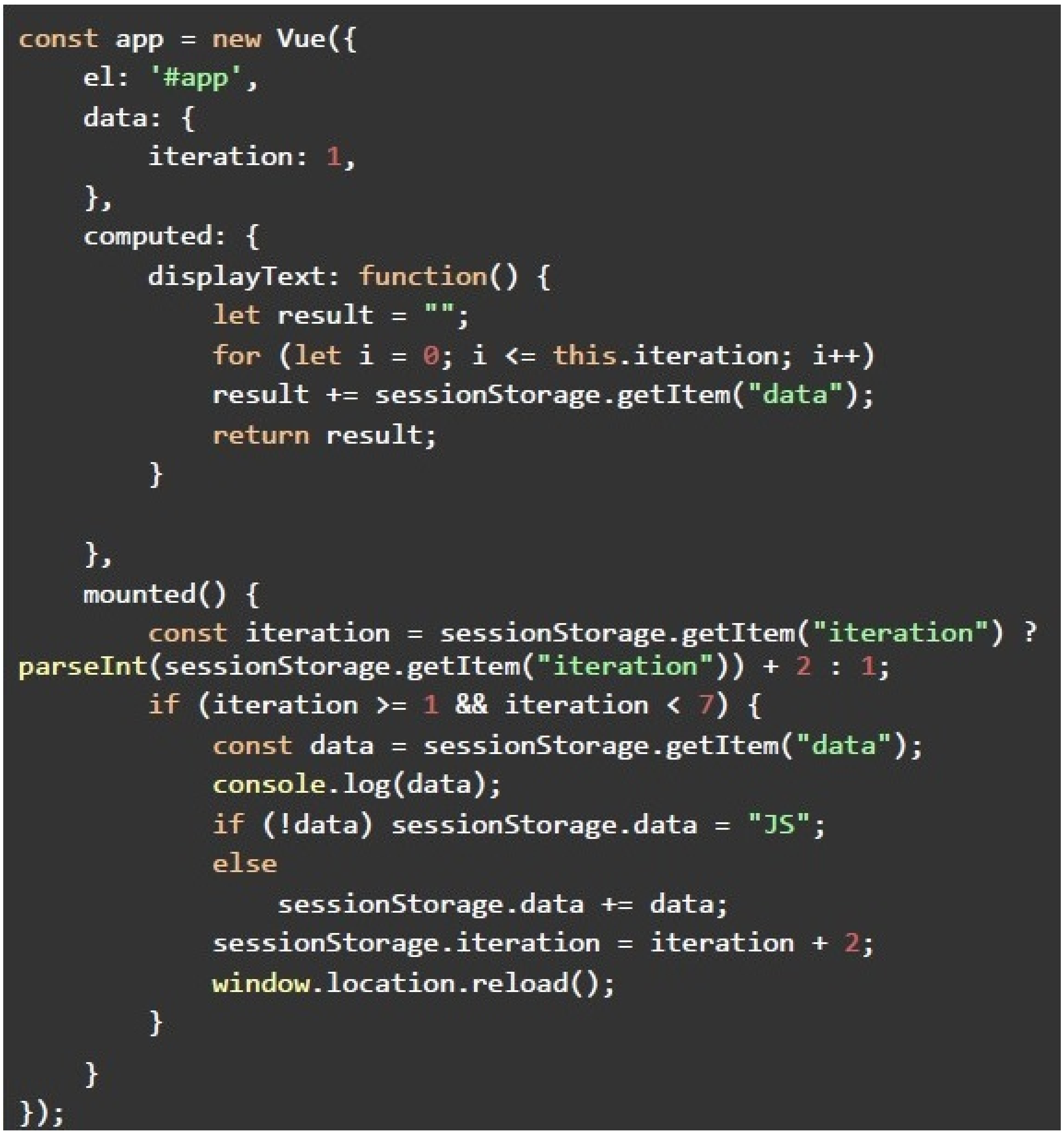
Consider the following Vue application with markup “index.html” and javascript file “app.js”.

index.html:



```
<style>
        .heading {
            color: yellow;
        }

        .raw-text {
            color: blue;
        }
</style>
<body>
      <div id="app">
          <p :class="[iteration ? 'heading' : 'raw-text']"> {{displayText}}
</p>
      </div>
</body>
<script src="./script.js"> </script>
```

app.js:



```
const app = new Vue({
    el: '#app',
    data: {
        iteration: 1,
    },
    computed: {
        displayText: function() {
            let result = "";
            for (let i = 0; i <= this.iteration; i++)
            result += sessionStorage.getItem("data");
            return result;
        }

    },
    mounted() {
        const iteration = sessionStorage.getItem("iteration") ?
parseInt(sessionStorage.getItem("iteration")) + 2 : 1;
        if (iteration >= 1 && iteration < 7) {
            const data = sessionStorage.getItem("data");
            console.log(data);
            if (!data) sessionStorage.data = "JS";
            else
                sessionStorage.data += data;
            sessionStorage.iteration = iteration + 2;
            window.location.reload();
        }
    }
});
```

If you open the 'index.html' file in a browser, what content will be displayed on the browser screen after the automatic reloading of the browser tab stops?

Options :

6406532306878. ✖ Black

6406532306879. ✖ Blue

6406532306880. ✔ Yellow

6406532306881. ✖ None of these

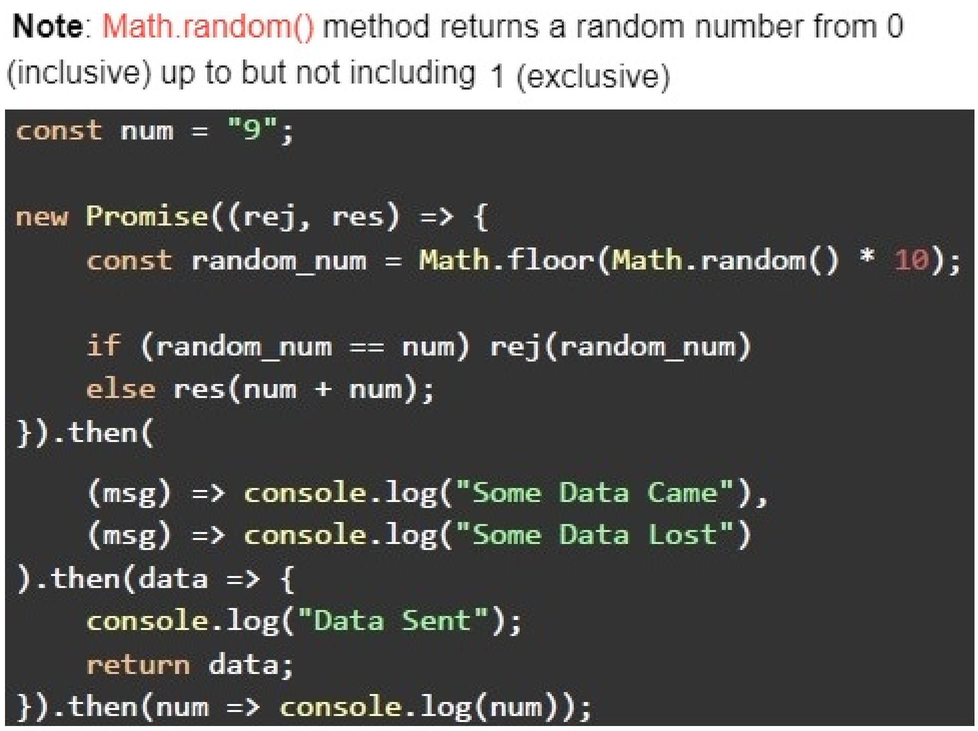
Question Number : 167 Question Id : 640653689591 Question Type : MCQ Is Question

Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 3

Question Label : Multiple Choice Question

Consider the below javascript program, and predict the output assuming the variable “random_num” always has a value greater than 8.



Note: Math.random() method returns a random number from 0 (inclusive) up to but not including 1 (exclusive)

```
const num = "9";

new Promise((rej, res) => {
    const random_num = Math.floor(Math.random() * 10);

    if (random_num == num) rej(random_num)
    else res(num + num);
}).then(
    (msg) => console.log("Some Data Came"),
    (msg) => console.log("Some Data Lost")
).then(data => {
    console.log("Data Sent");
    return data;
}).then(num => console.log(num));
```

Options :

6406532306890. ✖ Some Data Came

Some Data Lost

Some Data Lost

6406532306891. ✖ Some Data Came

Data Sent

Data Sent

6406532306892. ✓ Some Data Came

Data Sent

undefined

6406532306893. ✖ Some Data Lost

Data Sent

undefined

6406532306894. ✖ Some Data Lost

undefined

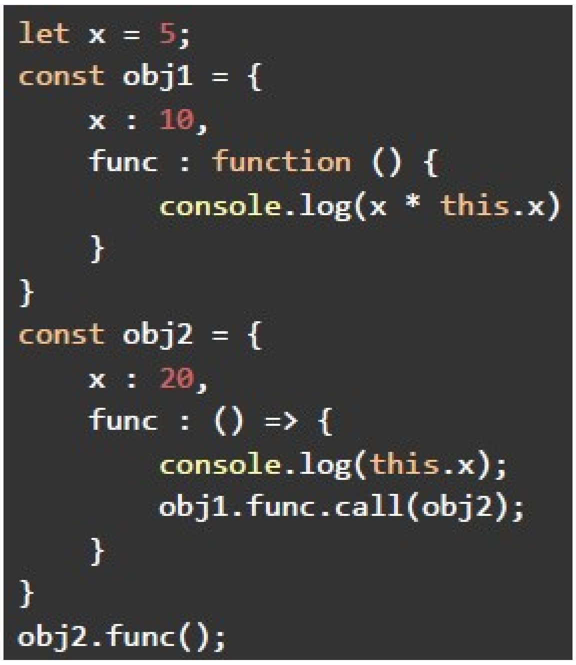
Data Sent

Question Number : 168 Question Id : 640653689594 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 3

Question Label : Multiple Choice Question

Consider the below javascript program, and predict the output if executed.



```
let x = 5;
const obj1 = {
    x : 10,
    func : function () {
        console.log(x * this.x)
    }
}
const obj2 = {
    x : 20,
    func : () => {
        console.log(this.x);
        obj1.func.call(obj2);
    }
}
obj2.func();
```

Options :

6406532306903. ✖ 20

100

6406532306904. ✖ 5

NaN

6406532306905. ✖ 20

NaN

6406532306906. ✖ 5

50

6406532306907. ✓ undefined

100

6406532306908. ✗ undefined

50

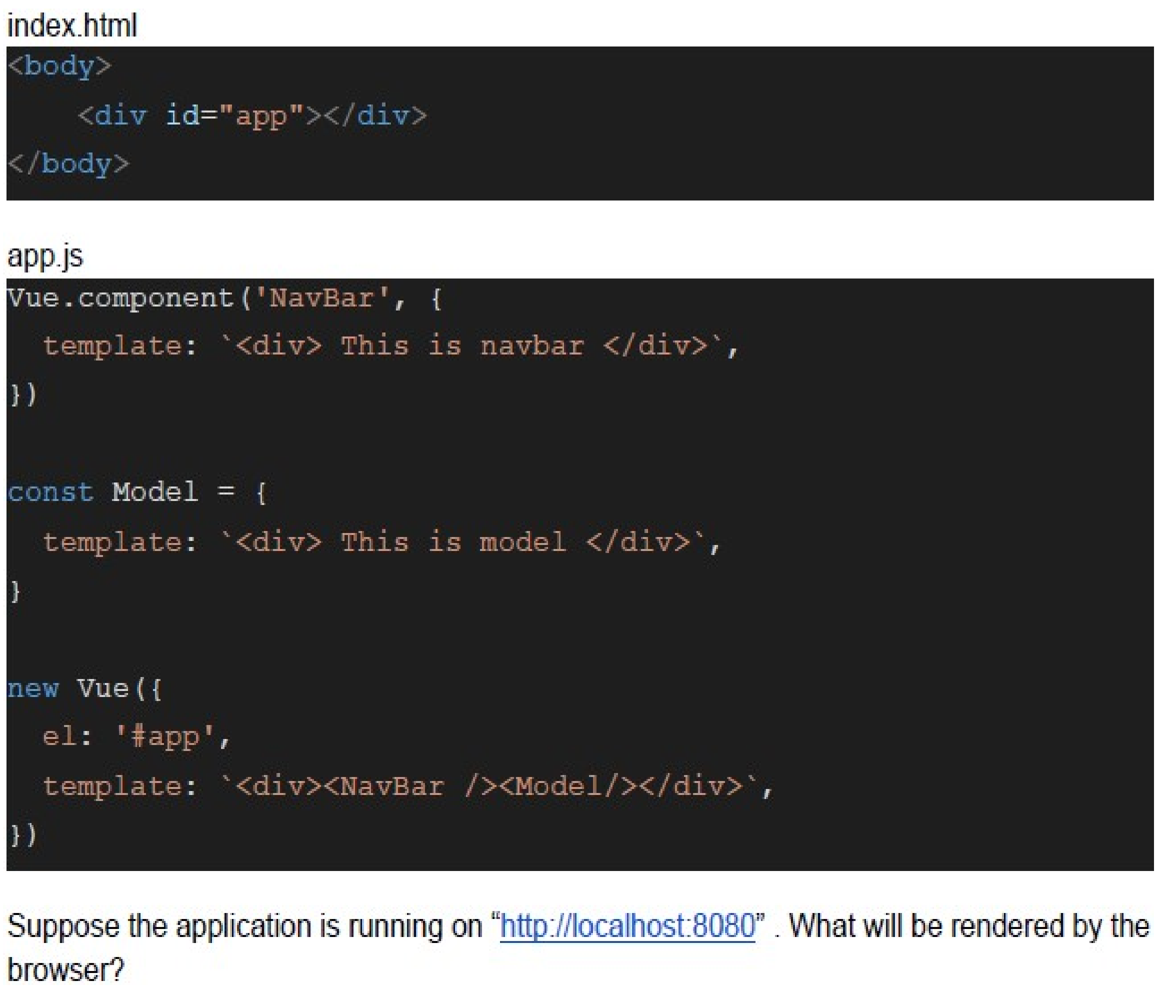
Sub-Section Number :	3
Sub-Section Id :	640653100854
Question Shuffling Allowed :	Yes
Is Section Default? :	null

Question Number : 169 Question Id : 640653689580 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 2

Question Label : Multiple Choice Question

Consider the Vue app with markup index.html and script app.js



index.html

```html
<body>
    <div id="app"></div>
</body>
```

app.js

```javascript
Vue.component('NavBar', {
  template: `<div> This is navbar </div>`,
})

const Model = {
  template: `<div> This is model </div>`,
}

new Vue({
  el: '#app',
  template: `<div><NavBar /><Model/></div>`,
})
```

Suppose the application is running on "http://localhost:8080" . What will be rendered by the browser?

Options :

6406532306843. ✓ This is navbar

6406532306844. ✗ This is model

6406532306845. ✗ This is navbar
This is model

6406532306846. ✗ None of these

Question Number : 170 Question Id : 640653689592 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 2

Question Label : Multiple Choice Question

Which of the following statements is false regarding promise in javascript?

Options :

6406532306895. ✗ A promise has 3 states - pending, fulfilled & rejected.

6406532306896. ✗ The promises can be chained to perform dependent tasks.

6406532306897. ✓ Promises can have only two states, Fulfilled and Rejected.

6406532306898. ✗ A promise constructor accepts an executor function, which in itself received 2 parameters for resolve and reject functions.

Sub-Section Number :	4
Sub-Section Id :	640653100855
Question Shuffling Allowed :	Yes
Is Section Default? :	null

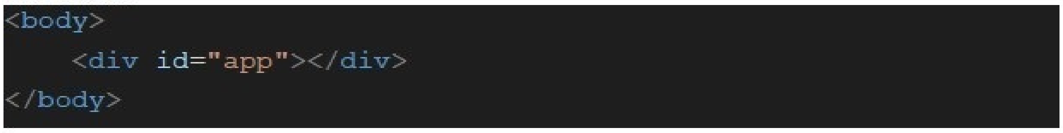
Question Number : 171 Question Id : 640653689582 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 4.5

Question Label : Multiple Choice Question

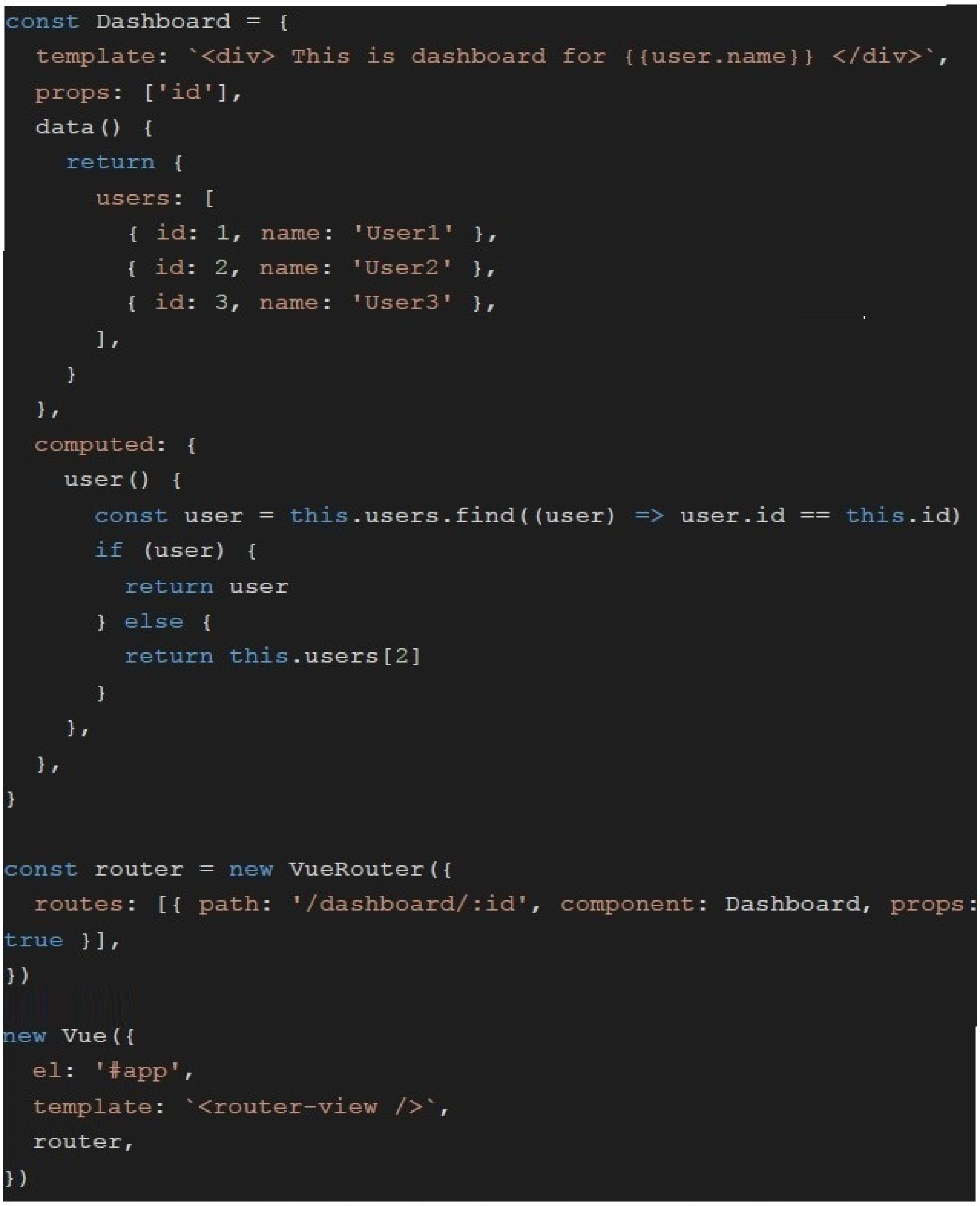
Consider the following Vue app with markup index.html and JavaScript app.js.

index.html



```html
<body>
    <div id="app"></div>
</body>
```

app.js



```javascript
const Dashboard = {
  template: `<div> This is dashboard for {{user.name}} </div>`,
  props: ['id'],
  data() {
    return {
      users: [
        { id: 1, name: 'User1' },
        { id: 2, name: 'User2' },
        { id: 3, name: 'User3' },
      ],
    }
  },
  computed: {
    user() {
      const user = this.users.find((user) => user.id == this.id)
      if (user) {
        return user
      } else {
        return this.users[2]
      }
    },
  },
}

const router = new VueRouter({
  routes: [{ path: '/dashboard/:id', component: Dashboard, props:
true }],
})

new Vue({
  el: '#app',
  template: `<router-view />`,
  router,
})
```

Suppose the application is running on "http://localhost:8080" . What will be rendered inside the router-view component of root component for the URL "http://localhost:8080/dashboard/1" ?

Options :

6406532306851. ✓ This is dashboard for User1

6406532306852. ✖ This is dashboard for User2

6406532306853. ✖ This is dashboard for User3

6406532306854. ✖ None of these

Question Number : 172 Question Id : 640653689585 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 4.5

Question Label : Multiple Choice Question

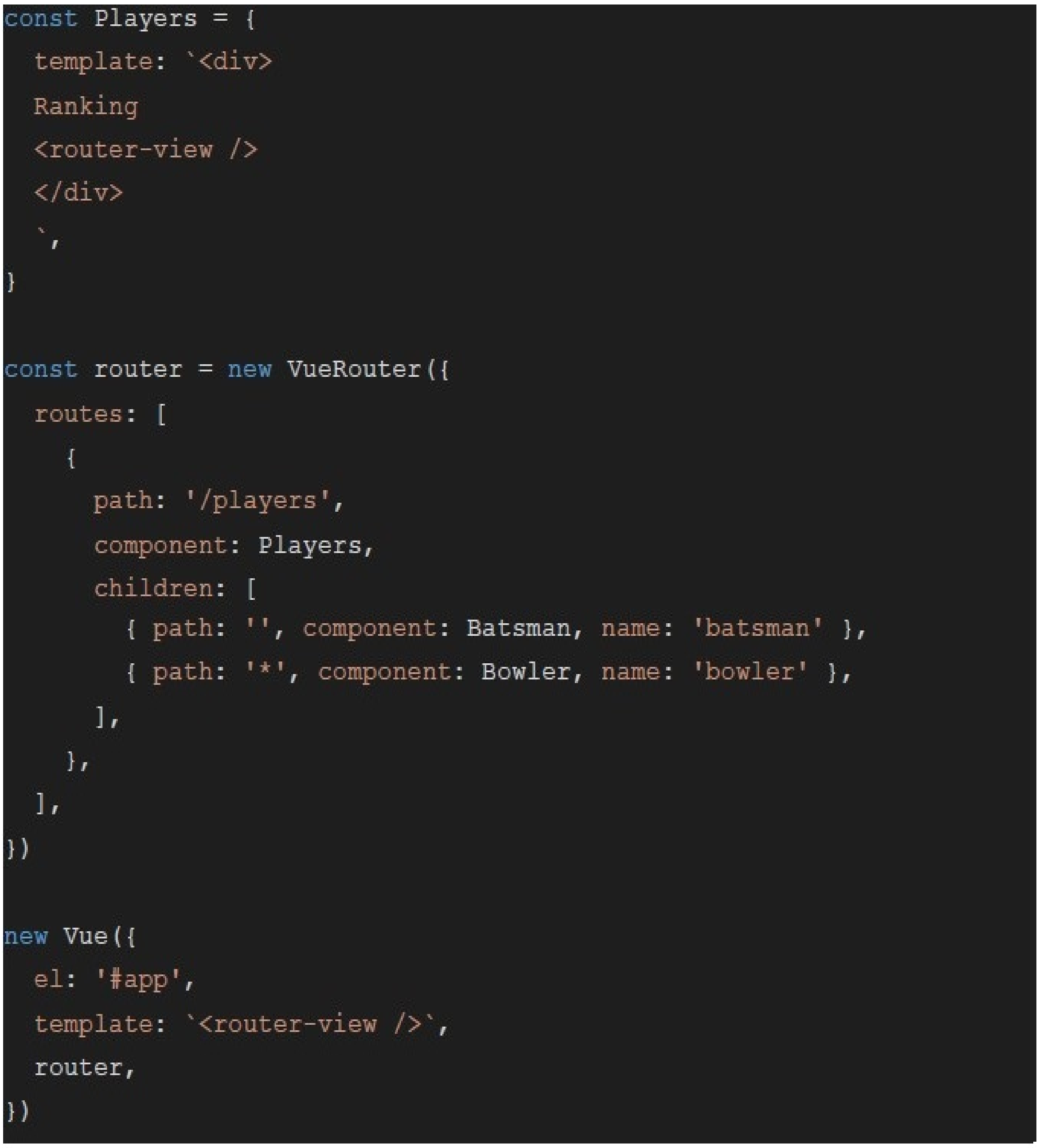
Consider the following Vue application with JavaScript app.js and markup index.html

index.html

```html
<body>
    <div id="app"></div>
</body>
```

app.js

```javascript
const players = [
  { name: 'Rohit', role: 'Batsman', runs: 397 },
  { name: 'Jaspreet', role: 'Bowler', wickets: 9 },
  { name: 'Virat', role: 'Batsman', runs: 355 },
  { name: 'Kuldeep', role: 'Bowler', wickets: 10 },
]
const Batsman = {
  template: `<div><span v-for="batsman in batsmans">{{batsman.name}}:
{{batsman.runs}}</span></div>`,
  computed: {
    batsmans() {
      return players.filter((player) => player.role == 'Bowler')
    },
  },
}

const Bowler = {
  template: `<div><span v-for="bowler in bowlers">{{bowler.name}}:
{{bowler.wickets}}</span></div>`,
  computed: {
    bowlers() {
      return players.filter((player) => player.role == 'Batsman')
    },
  },
}
```



```
const Players = {
  template: `<div>
  Ranking
  <router-view />
  </div>
  `,
}

const router = new VueRouter({
  routes: [
    {
      path: '/players',
      component: Players,
      children: [
        { path: '', component: Batsman, name: 'batsman' },
        { path: '*', component: Bowler, name: 'bowler' },
      ],
    },
  ],
})

new Vue({
  el: '#app',
  template: `<router-view />`,
  router,
})
```

Suppose the application is running on “http://localhost:8080” . What will be rendered inside the router-view component of Players component by the browser for the URL “http://localhost:8080/#/players” ?

Options :

6406532306863. ✔ Jaspreet: Kuldeep:

6406532306864. ✖ Jaspreet: 9 Kuldeep: 10

6406532306865. ✖ Rohit: Virat:

6406532306866. ✖ Rohit:397 Virat:355

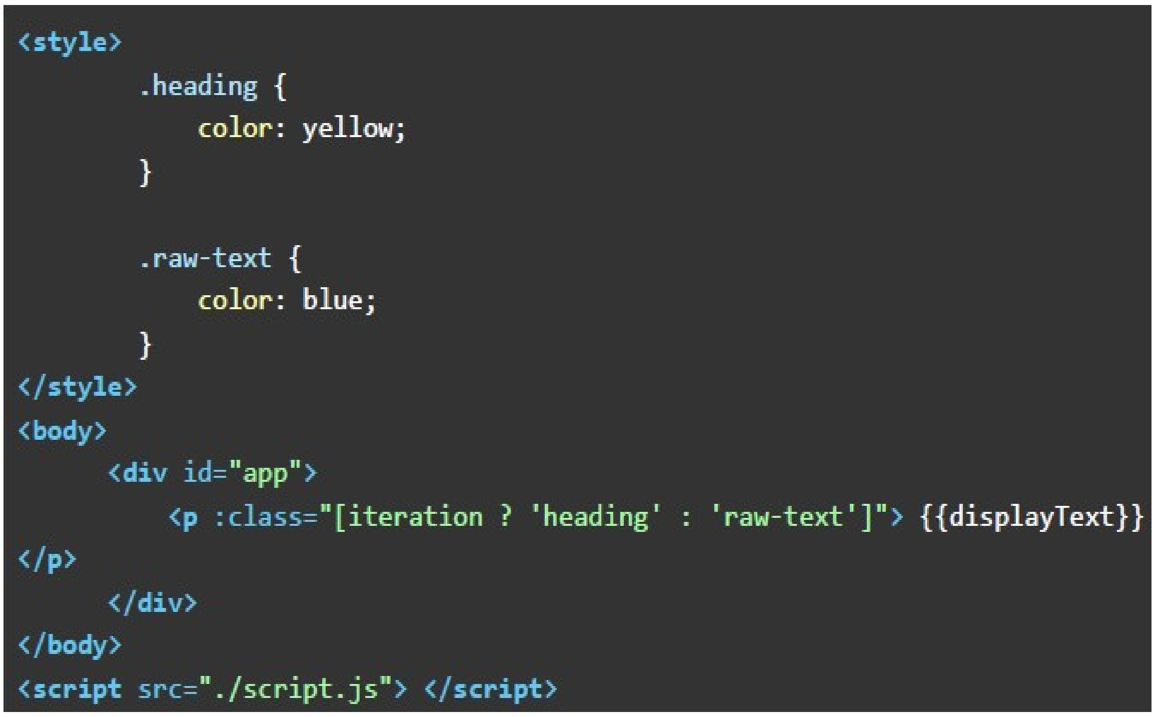
Question Number : 173 Question Id : 640653689587 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 4.5

Question Label : Multiple Choice Question

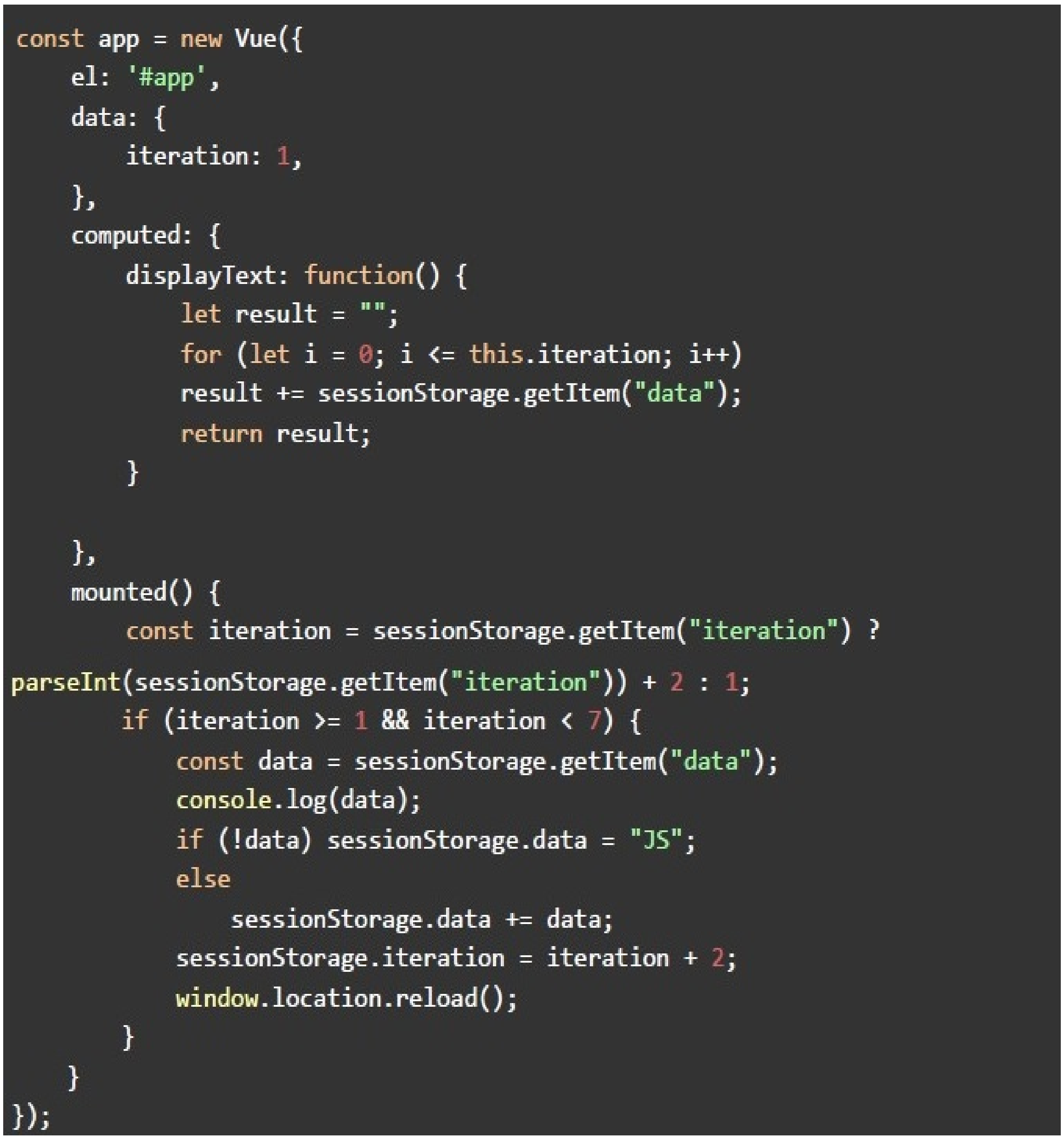
Consider the following Vue application with markup “index.html” and javascript file “app.js”.

index.html:



```
<style>
        .heading {
            color: yellow;
        }

        .raw-text {
            color: blue;
        }
</style>
<body>
      <div id="app">
          <p :class="[iteration ? 'heading' : 'raw-text']"> {{displayText}}
</p>
      </div>
</body>
<script src="./script.js"> </script>
```

app.js:



```
const app = new Vue({
    el: '#app',
    data: {
        iteration: 1,
    },
    computed: {
        displayText: function() {
            let result = "";
            for (let i = 0; i <= this.iteration; i++)
            result += sessionStorage.getItem("data");
            return result;
        }

    },
    mounted() {
        const iteration = sessionStorage.getItem("iteration") ?
parseInt(sessionStorage.getItem("iteration")) + 2 : 1;
        if (iteration >= 1 && iteration < 7) {
            const data = sessionStorage.getItem("data");
            console.log(data);
            if (!data) sessionStorage.data = "JS";
            else
                sessionStorage.data += data;
            sessionStorage.iteration = iteration + 2;
            window.location.reload();
        }
    }
});
```

Suppose you open the file “index.html” in a browser, what will be shown on the browser screen post which browser tab will not be reloaded programmatically?

If you open the 'index.html' file in a browser, what content will be displayed on the browser screen after the automatic reloading of the browser tab stops?

Options :

6406532306871. ✖ JSJS

6406532306872. ✖ JSJSJS

6406532306873. ✓ JSJSJSJS

6406532306874. ✖ NaNNaNNaN

6406532306875. ✖ NaNNaN

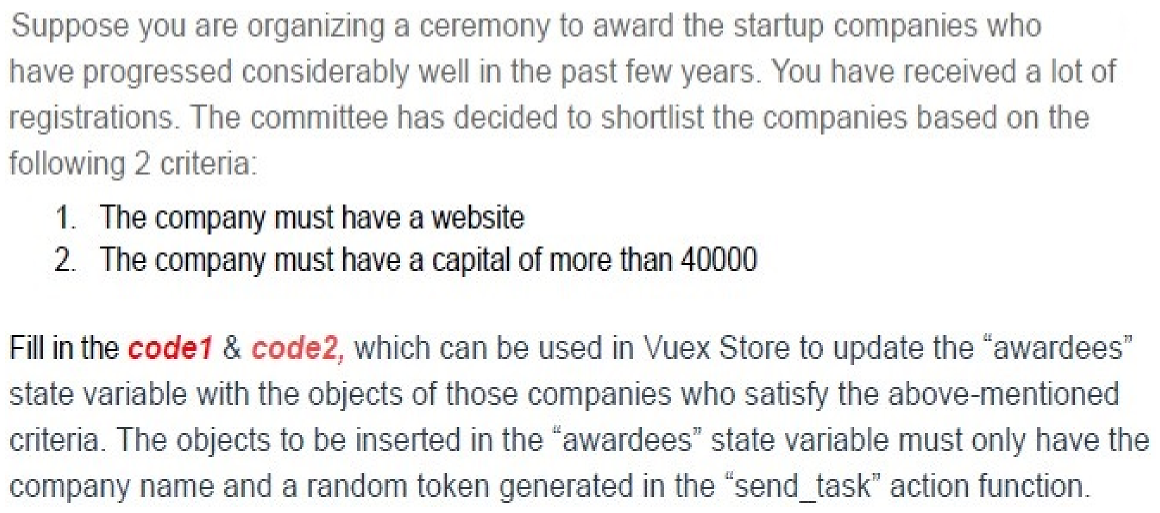
6406532306876. ✖ NaNNaNNaNNaN

6406532306877. ✖ None of these

Question Number : 174 Question Id : 640653689589 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 4.5

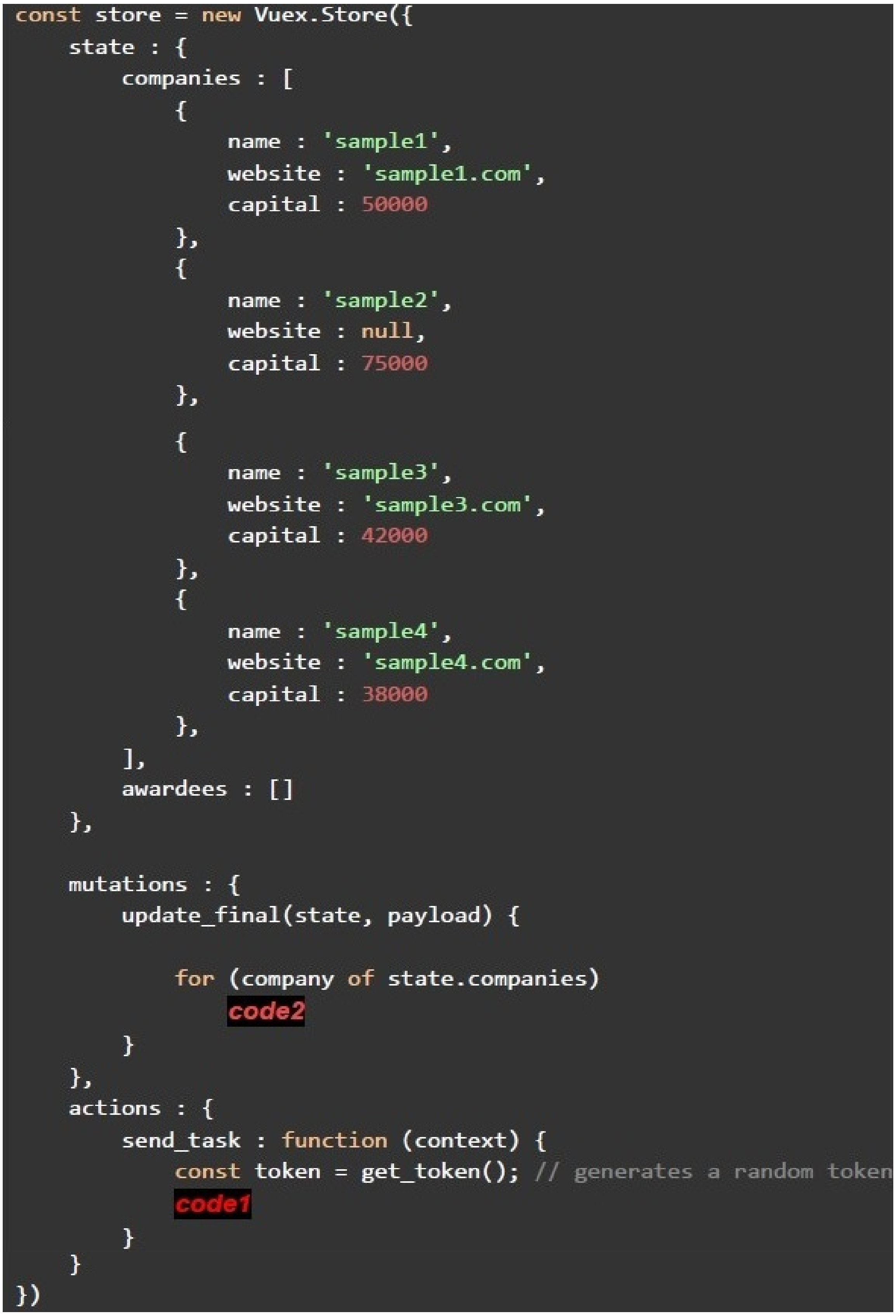
Question Label : Multiple Choice Question



Suppose you are organizing a ceremony to award the startup companies who have progressed considerably well in the past few years. You have received a lot of registrations. The committee has decided to shortlist the companies based on the following 2 criteria:

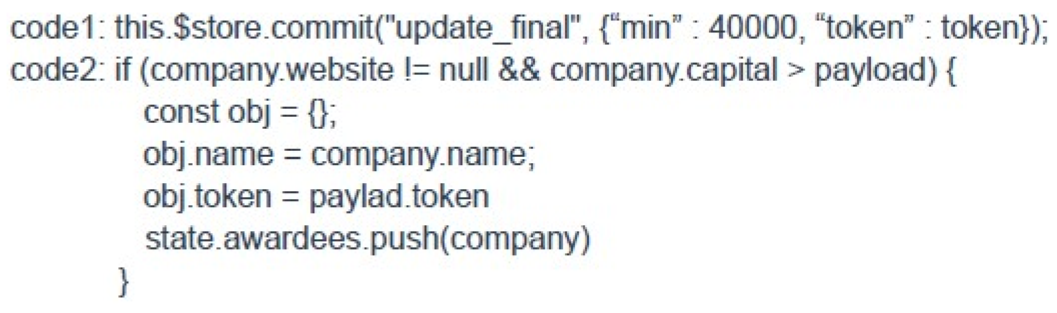
1. The company must have a website
2. The company must have a capital of more than 40000

Fill in the ***code1*** & ***code2***, which can be used in Vuex Store to update the “awardees” state variable with the objects of those companies who satisfy the above-mentioned criteria. The objects to be inserted in the “awardees” state variable must only have the company name and a random token generated in the “send_task” action function.



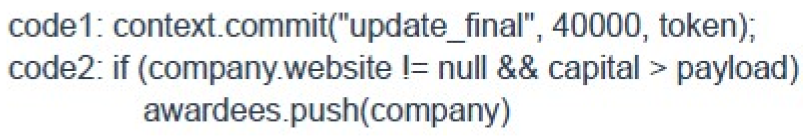
```
const store = new Vuex.Store({
    state : {
        companies : [
            {
                name : 'sample1',
                website : 'sample1.com',
                capital : 50000
            },
            {
                name : 'sample2',
                website : null,
                capital : 75000
            },

            {
                name : 'sample3',
                website : 'sample3.com',
                capital : 42000
            },
            {
                name : 'sample4',
                website : 'sample4.com',
                capital : 38000
            },
        ],
        awardees : []
    },

    mutations : {
        update_final(state, payload) {

            for (company of state.companies)
                code2
        }
    },
    actions : {
        send_task : function (context) {
            const token = get_token(); // generates a random token
            code1
        }
    }
})
```

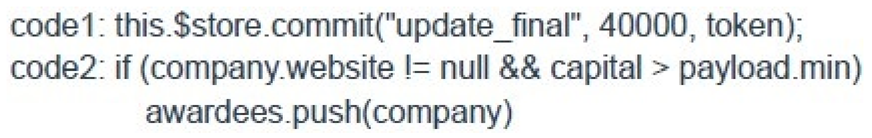
Options :

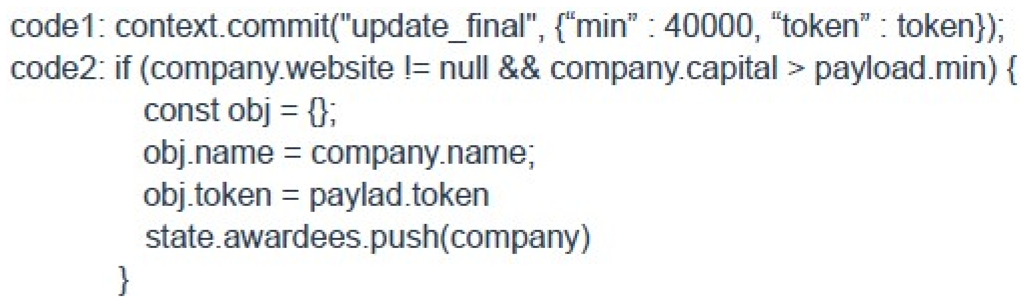


```
code1: this.$store.commit("update_final", {“min” : 40000, “token” : token});
code2: if (company.website != null && company.capital > payload) {
           const obj = {};
           obj.name = company.name;
           obj.token = paylad.token
           state.awardees.push(company)
       }
```

6406532306882.

6406532306883. ✗ 
```
code1: context.commit("update_final", 40000, token);
code2: if (company.website != null && capital > payload)
            awardees.push(company)
```

6406532306884. ✗ 
```
code1: this.$store.commit("update_final", 40000, token);
code2: if (company.website != null && capital > payload.min)
            awardees.push(company)
```

6406532306885. ✓ 
```
code1: context.commit("update_final", {"min" : 40000, "token" : token});
code2: if (company.website != null && company.capital > payload.min) {
            const obj = {};
            obj.name = company.name;
            obj.token = paylad.token
            state.awardees.push(company)
        }
```

Sub-Section Number : 5

Sub-Section Id : 640653100856

Question Shuffling Allowed : Yes

Is Section Default? : null

Question Number : 175 Question Id : 640653689584 Question Type : MSQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 2 Max. Selectable Options : 0

Question Label : Multiple Select Question

Which of the following is/are true regarding JSON Web Tokens?

Options :

6406532306859. ✓ Each token has three components, header, payload and signature

6406532306860. ✗ Each token has two components, payload and signature.

6406532306861. ✓ Each component of the token is encoded using Base-64.

6406532306862. ✓ Components of the token are joined using period.

Question Number : 176 Question Id : 640653689590 Question Type : MSQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 2 Max. Selectable Options : 0

Question Label : Multiple Select Question

Which of the following statement(s) is/are true, regarding REST and GraphQL?

Options :

6406532306886. ✓ GraphQL helps to fetch exactly the same data which is needed, and avoids over fetching as well as under fetching

6406532306887. ✓ In general, a GraphQL response always returns 200 status code, with the "error" field containing the errors (if any).

6406532306888. ✓ A REST API provides multiple endpoints to access multiple resources.

6406532306889. ✗ None of these.

Sub-Section Number :	6
Sub-Section Id :	640653100857
Question Shuffling Allowed :	Yes
Is Section Default? :	null

Question Number : 177 Question Id : 640653689593 Question Type : MSQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 3 Max. Selectable Options : 0

Question Label : Multiple Select Question

Which of the following statement(s) is/are correct using Vuex?

Options :

6406532306899. ✓ Vuex is a state management library for Vue.js applications.

6406532306900. ✗ Using Vuex always becomes difficult when an application scales.

6406532306901. ✓ Vuex provides actions for performing asynchronous operations.

6406532306902. ✖ The Vuex getters are not reactive.

MLT

Section Id :	64065348510
Section Number :	12
Section type :	Online
Mandatory or Optional :	Mandatory
Number of Questions :	12
Number of Questions to be attempted :	12
Section Marks :	50
Display Number Panel :	Yes
Group All Questions :	No
Enable Mark as Answered Mark for Review and Clear Response :	Yes
Maximum Instruction Time :	0
Sub-Section Number :	1
Sub-Section Id :	640653100858
Question Shuffling Allowed :	No
Is Section Default? :	null

Question Number : 178 Question Id : 640653689595 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 0

Question Label : Multiple Choice Question

THIS IS QUESTION PAPER FOR THE SUBJECT "DIPLOMA LEVEL : MACHINE LEARNING TECHNIQUES (COMPUTER BASED EXAM)"

ARE YOU SURE YOU HAVE TO WRITE EXAM FOR THIS SUBJECT?