Started on	Friday, 8 September 2023, 7:53 AM
State	Finished
Completed on	Friday, 8 September 2023, 7:55 AM
Time taken	2 mins 8 secs
Marks	12.00/12.00
Grade	**10.00** out of 10.00 (**100**%)

QUESTION 1

Correct

Mark 1.00 out of 1.00

What is the output of the following code:

```
function job() {
    return new Promise(function(resolve, reject) {
        reject();
    });
}

let promise = job();

promise
.then(function() {
    console.log('Success 1');
})
.then(function() {
    console.log('Success 2');
})
.then(function() {
    console.log('Success 3');
})
.catch(function() {
    console.log('Error 1');
})
.then(function() {
    console.log('Success 4');
});
```

Select one:

[✓ a. Error 1, Success 4] ✓

b. Success 1, Success 2, Success 3, Success 4

c. Success 1, Success 2, Success 3, Error 1, Success 4

d. Success 1, Error 1

e. Error 1

f. Error 1, Success 1, Success 2, Success 3, Success 4

QUESTION 2

Correct

Mark 1.00 out of 1.00

What is the output of the following code:

```
function job(state) {
    return new Promise(function(resolve, reject) {
        if (state) {
            resolve('success');
        } else {
            reject('error');
        }
    });
}

let promise = job(true);

promise
.then(function(data) {
    console.log(data);
    return job(false);
})
.catch(function(error) {
    console.log(error);
    return 'Error caught';
})
.then(function(data) {
    console.log(data);
    return job(true);
})
.catch(function(error) {
    console.log(error);
});
```

Select one:

- a. success, success
- b. error, Error caught, success
- c. error, success, Error caught
- [✓ d. success, error, Error caught] ✓
- e. success, error, success, error

QUESTION 3

Correct

Mark 1.00 out of 1.00

What are 2 native functions to run code asynchronously in JavaScript ?

Select one or more:

- ☐ a. interval
- ☑ b. setInterval ✓
- ☐ c. repeat
- ☐ d. delay
- ☐ e. startInterval
- ☐ f. timeout
- ☑ g. setTimeout ✓

QUESTION 4

Correct

Mark 1.00 out of 1.00

What is the output of the following code?

```
let fs = require('fs');

console.log('1');

fs.readFile('test.txt', 'utf8', function(error, data) {
    if (error) {
        throw error;
    }
    console.log('2');
});

console.log('3');
```

Select one:

- ◉ a. 1 3 2 ✓
- ○ b. 3 2 1
- ○ c. 2 3 1
- ○ d. 2 1 3
- ○ e. 1 2 3

QUESTION 5

Correct

Mark 1.00 out of 1.00

What is the function to stop an interval timer?

Select one:

[✓ a. clearInterval] ✓

b. clearTimer

c. shutdownTimer

d. stopTimer

QUESTION 6

Correct

Mark 1.00 out of 1.00

What will be displayed to the console when calling the `f()` function?

```
async function f() {
  let result = 'first!';
  let promise = new Promise((resolve, reject) => {
    setTimeout(() => resolve('done!'), 1000);
  });

  result = await promise;

  console.log(result);
}

f();
```

Select one:

a. first!

b. JavaScript error

[✓ c. done!] ✓

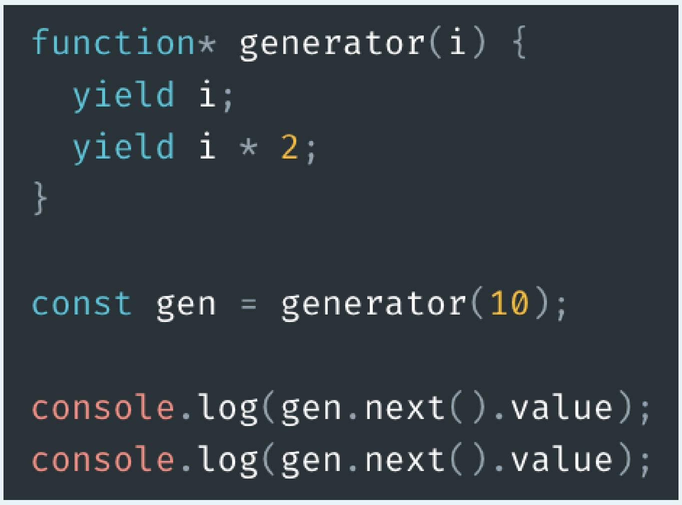
d. Something else

QUESTION 7

Correct

Mark 1.00 out of 1.00

What is the output of the following code?



```
function* generator(i) {
  yield i;
  yield i * 2;
}

const gen = generator(10);

console.log(gen.next().value);
console.log(gen.next().value);
```

Select one:

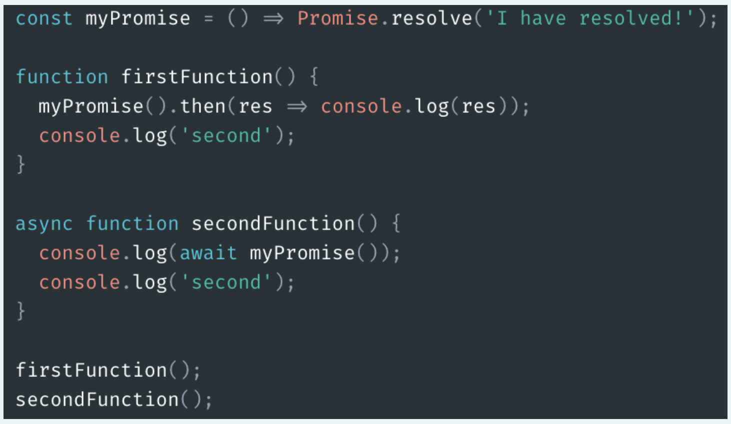
- a. 0, 10 and 10, 20
- b. [0, 10], [10, 20]
- [✓ c. 10, 20] ✓
- d. 20, 20

QUESTION 8

Correct

Mark 1.00 out of 1.00

What is the output of the following code?



```
const myPromise = () => Promise.resolve('I have resolved!');

function firstFunction() {
  myPromise().then(res => console.log(res));
  console.log('second');
}

async function secondFunction() {
  console.log(await myPromise());
  console.log('second');
}

firstFunction();
secondFunction();
```

Select one:

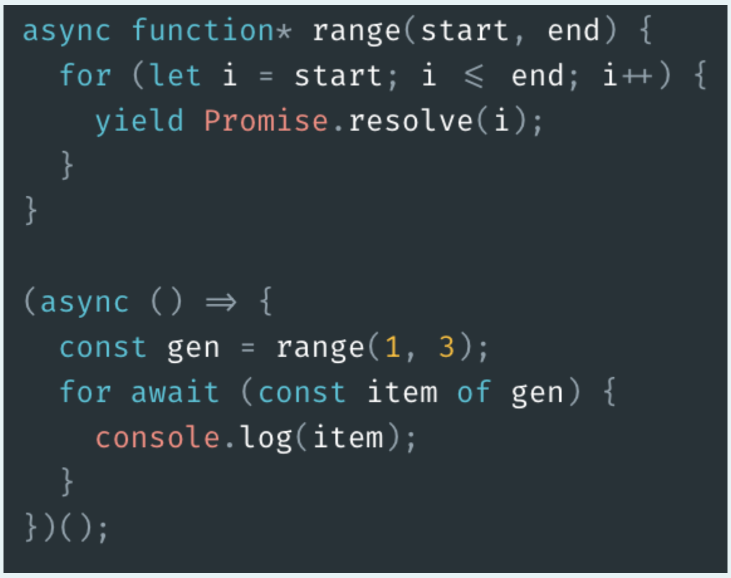
- a. I have resolved!, second and I have resolved!, second
- [✓ b. second, I have resolved! and I have resolved!, second] ✓
- c. I have resolved!, second and second, I have resolved!
- d. second, I have resolved! and second, I have resolved!

QUESTION 9

Correct

Mark 1.00 out of 1.00

What's the output of the following code?



```
async function* range(start, end) {
  for (let i = start; i <= end; i++) {
    yield Promise.resolve(i);
  }
}

(async () => {
  const gen = range(1, 3);
  for await (const item of gen) {
    console.log(item);
  }
})();
```

Select one:

- [✓ a. 1 2 3] ✓
- b. Promise {1} Promise {2} Promise {3}
- c. undefined undefined undefined
- d. Promise {<pending>} Promise {<pending>} Promise {<pending>}

QUESTION 10

Correct

Mark 1.00 out of 1.00

Select the missing line of code

```
const teams = [
  { name: 'Team 1', members: ['Paul', 'Lisa'] },
  { name: 'Team 2', members: ['Laura', 'Tim'] },
];

function* getMembers(members) {
  for (let i = 0; i < members.length; i++) {
    yield members[i];
  }
}

function* getTeams(teams) {
  for (let i = 0; i < teams.length; i++) {
    // ✨ SOMETHING IS MISSING HERE ✨
  }
}

const obj = getTeams(teams);
obj.next(); // { value: "Paul", done: false }
obj.next(); // { value: "Lisa", done: false }
```

Select one:

- a. return yield getMembers(teams[i].members)
- b. yield getMembers(teams[i].members)
- c. return getMembers(teams[i].members)
- [✓ d. yield* getMembers(teams[i].members)] ✓

QUESTION 11

Correct

Mark 1.00 out of 1.00

What types of errors can we handle with the try..catch construct?

Select one:

- a. logical errors
- [✓ b. runtime errors] ✓
- c. critical errors
- d. syntax errors
- e. all of the above

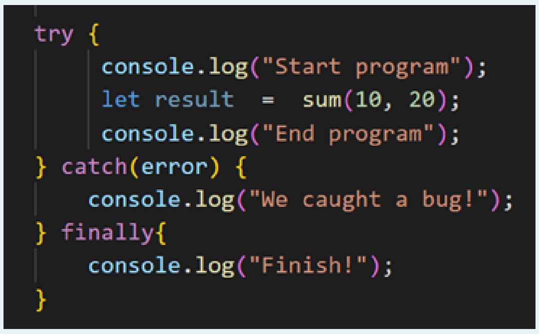
QUESTION 12

Correct

Mark 1.00 out of 1.00

What will be the result of executing?

(*sum* was not defined neither before nor after the given code.)



```
try {
     console.log("Start program");
     let result  =  sum(10, 20);
     console.log("End program");
} catch(error) {
    console.log("We caught a bug!");
} finally{
    console.log("Finish!");
}
```

Select one:

a.
Start program
We caught a bug!

b.
Start program
End program
We caught a bug!

[✓ c.] ✓
Start program
We caught a bug!
Finish!

d.
Start program
Finish!

e.
Start program
End program
Finish!

f.
Start program
 End program
We caught a bug!
Finish!

◄ Tasks. Asynchronous JS, Closures, Exceptions

Jump to...

Slides. Scrum Fundamental ►