

Você entendeu o tratamento de erros?

Maria observou que durante as aulas, o instrutor adicionou ao final do arquivo `custom-express.js` os dois *middlewares* de tratamento de erros. De modo que o código de forma resumida ficou como a seguir:

```
// parte inicial do código omitida.

const rotas = require('../app/rotas/rotas');
rotas(app);

app.use(function (req, resp, next) {
  return resp.status(404).marko(
    // caminho para o template omitido.
  );
});

app.use(function (erro, req, resp, next) {
  return resp.status(500).marko(
    // caminho para o template omitido.
  );
});

module.exports = app;
```

Sobre a ordem da declaração dos *middlewares* podemos afirmar corretamente:

Selecione 2 alternativas

A O código funcionaria como esperado qualquer que fosse a ordem de declaração! Portanto, não seria necessário que os *middlewares* de tratamento de erro fossem os últimos a serem declarados.

B Maria também poderia fazer da seguinte forma:

```
// parte inicial do código omitida.

app.use(function (req, resp, next) {
  return resp.status(404).marko(
    // caminho para o template omitido.
  );
});

const rotas = require('../app/rotas/rotas');
rotas(app);

app.use(function (erro, req, resp, next) {
  return resp.status(500).marko(
    // caminho para o template omitido.
  );
});

module.exports = app;
```

C Os *middlewares* de tratamento de erro precisam vir necessariamente ao final de toda configuração.

D A configuração poderia ser como a seguir e também funcionaria corretamente.
// parte inicial do código omitida.

```
const rotas = require('../app/rotas/rotas');
rotas(app);

app.use(function (erro, req, resp, next) {
  return resp.status(500).marko(
    // caminho para o template omitido.
  );
});

app.use(function (req, resp, next) {
  return resp.status(404).marko(
    // caminho para o template omitido.
  );
});

module.exports = app;
```