During this quiz, you will be experimenting with mongodb, so you will have to have a database running.

To enter the mongo shell use whatever MongoDB shell you have installed. (mongo, mongosh, or VS Code extension)

Use the following commands in the mongo shell to create the database, insert documents, and execute a query:

use testDB  
coll = db.getCollection("newCollection")  
coll.find()  
coll.insert({ vehicle: "plane", speed: "480mph" })  
coll.insert({ vehicle: "car", speed: "120mph" })  
coll.insert({ vehicle: "train", speed: "120mph" })  
coll.find()

**Question 1**

**2 / 2 pts**

What is returned in response to the mongo request:

coll.find({speed:"480mph"})

The \* indicates that it should be different for everyone so we can't put it there.

**Correct!**



{"\_id":ObjectID("\*"), "vehicle":"plane","speed":"480mph"}



{"\_id":ObjectID("\*"), "vehicle":"car","speed":"480mph"}



{"\_id":ObjectID("\*"), "vehicle":"car","speed":"120mph"}



{"\_id":ObjectID("\*"), "vehicle":"plane","speed":"120mph"}

**Question 2**

**5 / 5 pts**

Now we want to access our Mongo database using JavaScript.

Create a JavaScript file named mongoquiz.js and paste the following code into it.

const { MongoClient } = require('mongodb');  
  
const dbName = 'astro';  
const colName = 'nebulae';  
  
const docs = [  
  { ngc: 'NGC 7293', name: 'Helix', type: 'planetary', location: 'Aquila' },  
  { ngc: 'NGC 6543', name: "Cat's Eye", type: 'planetary', location: 'Draco' },  
  { ngc: 'NGC 1952', name: 'Crab', type: 'supernova', location: 'Taurus' },  
];  
  
async function main() {  
 const url = `<yourMongoDBConnectionStringHere>`;  
  const client = new MongoClient(url);  
  
  try {  
    await client.connect();  
  
    // Create the collection if it don't exist  
    const exists = await client  
      .db(dbName)  
      .listCollections({ name: colName })  
      .hasNext();  
    if (!exists) {  
      await client.db(dbName).collection(colName).insertMany(docs);  
   }  
  
    // Do a query  
    const result = await client  
      .db(dbName)  
      .collection(colName)  
      .findOne({ type: 'supernova' });  
    console.log('Found', result);  
  } finally {  
    await client.close();  
  }  
}  
  
main().catch(console.error);

Replace the value for yourMongoDBConnectionStringHere with a value that works with your MongoDB server. Save the file and run it with

node mongoquiz.js

Which nebulae name was output?



Helix



Cat's Eye**ct!**



Crab



Taurus

**Question 3**

**3 / 3 pts**

We can now use a JavaScript package called Mongoose to make it a little easier to work with Mongo, abstract a collection to have a set schema, and add methods associated with the schema.

Create a JavaScript file name mongoose.js and paste the following code into it.

const mongoose = require('mongoose');  
  
async function main() {  
  
// Connect to the server and set the database to cats  
await mongoose.connect(`<yourMongoDBConnectionStringHere>/cats`);  
  
// Create a schema to represent the collection  
const kittySchema = new mongoose.Schema({  
 name: String,  
 });  
  
// Create a function that is associated with the schema  
kittySchema.methods.speak = function speak() {  
 console.log('Meow name is ' + (this.name || 'secret'));  
};  
  
// Bind the schema to a Mongo collection name  
const Kitten = mongoose.model('kitten', kittySchema);  
  
// Add some kittens  
await new Kitten({ name: 'hairball' }).save();  
await new Kitten({ name: 'fluffy' }).save();  
await new Kitten().save();  
  
// Do a query on the kitten collection and speak the name of each kitten that is found  
const kittens = await Kitten.find();  
kittens.forEach((kitten) => {  
 kitten.speak();  
});  
}  
  
main().catch(console.error);

Replace the value for yourMongoDBConnectionStringHere with a value that work with your MongoDB server. Save the file and run it with

node mongoose.js

What cat names were output?



fluffy



hairball, fluffy, secret



hairball, fluffy



hairball



secret