National University of Computer and Emerging Sciences, Lahore Campus

Course:	Web Programming	Course Code:	CS-406
Program:	BS (Computer Science)	Semester:	Fall 2018
Duration:	10 Minutes	Total Marks:	10
Date:	24-Oct-18	Weight	-
Section:	CS-A	Page(s):	2
Quiz:	3	Reg. No.	

Instruction/Notes:

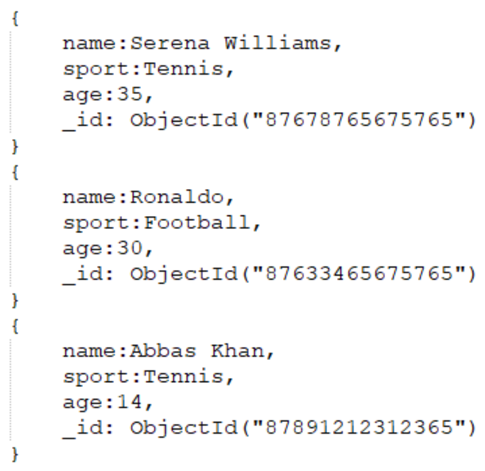
Write the correct answer in the answer box provided. Each MCQ is of one 1 mark.

1) Which of the following is the correct URI to connect to a MongoDB database called WebProg with user credentials:
   a. http://localhost:27017/WebProg
   b. mongodb://localhost:27017/WebProg
   c. mongodb://username:pass@localhost:2700/WebProg
   d. none of the above
2) Consider the following code block and identify the incorrect statement

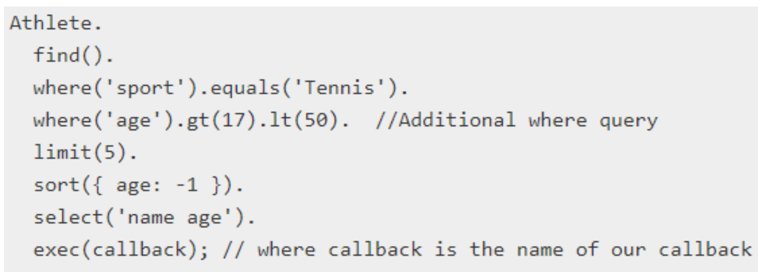
```
var locationSchema = new mongoose.Schema({
  name: {type: String, required: true},
  address: String,
  rating: {type: Number, "default": 0, min: 0, max: 5},
  facilities: String,
  reviews: [reviewSchema]
});
```

   a. Schema is a constructor function of Mongoose used to define a documents schema
   b. reviewSchema is nested inside locationSchema which means that review is a subdocument of location
   c. facilities can contain multiple strings
   d. address is an optional path
3) We can create new data in the database in an express application using the following request method
   a. GET
   b. POST
   c. PUT
   d. DELETE
4) If we want to specify the userId and bookId route parameters for a url, we use the following
   a. app.get('/users/userId:/books/bookId:',callback)
   b. app.get('/users/:userId/books/:bookId',callback)
   c. both a and b
   d. none of the above
5) If we want to access the bookId encoded in the URL such as http://localhost:3000/users/34/books/007, we can do this using
   a. res.params.bookId
   b. req.paramameters.bookId
   c. req.params.bookId
   d. res.parameters.bookId
6) Suppose we compile an Athlete model using the code: var Athlete=mongoose.model('Athlete', athleteSchema) and we need to get the names and ages of all athletes who play tennis. We do this by
   a. athleteSchema.findAll({'sport':'tennis'}, 'name age', callback)
   b. athleteSchema.find({'sport':'tennis'}, 'name age', callback)
   c. Athlete.findAll({'sport':'tennis'}, 'name age', callback)
   d. Athlete.find({'sport':'tennis'}, 'name age', callback)

7) Suppose we have the following 3 documents in our database, the code fragment shown will result in the following documents being retrieved:



```
{
    name:Serena Williams,
    sport:Tennis,
    age:35,
    _id: ObjectId("87678765675765")
}
{
    name:Ronaldo,
    sport:Football,
    age:30,
    _id: ObjectId("87633465675765")
}
{
    name:Abbas Khan,
    sport:Tennis,
    age:14,
    _id: ObjectId("87891212312365")
}
```



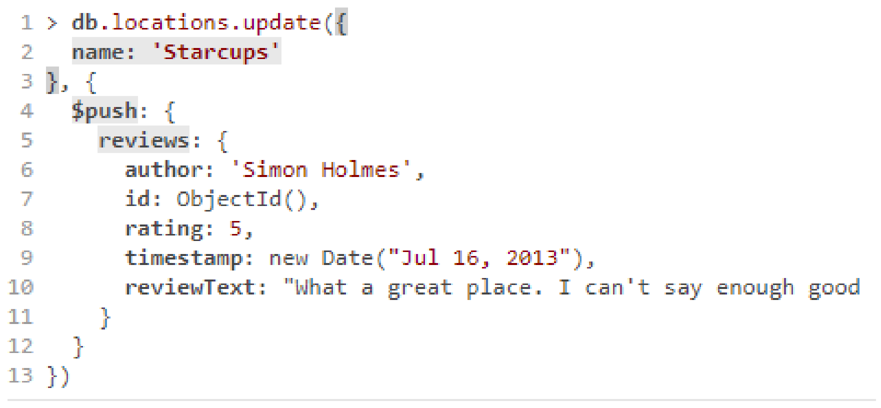
```
Athlete.
  find().
  where('sport').equals('Tennis').
  where('age').gt(17).lt(50).  //Additional where query
  limit(5).
  sort({ age: -1 }).
  select('name age').
  exec(callback); // where callback is the name of our callback
```

a. 1 and 2
b. Only 1
c. Only 3
d. None of the above

8) You can test your Node application database on localhost as well as an online database service provider
a. True
b. False

9) The NODE_ENV variable for a heroku deployment is set as production by default
a. True
b. False

10) The following code finds a location document having name property as Starcups and pushes a subdocument into the reviews path



```
> db.locations.update({
  name: 'Starcups'
}, {
  $push: {
    reviews: {
      author: 'Simon Holmes',
      id: ObjectId(),
      rating: 5,
      timestamp: new Date("Jul 16, 2013"),
      reviewText: "What a great place. I can't say enough good
    }
  }
})
```

a. True
b. False

Answer Box:

1	2	3	4	5	6	7	8	9	10
c	c	b	b	c	d	b	a	a	a