INDEX

PRACTICAL NO :- 1

➢ <u>MONGO DB BASICS</u>

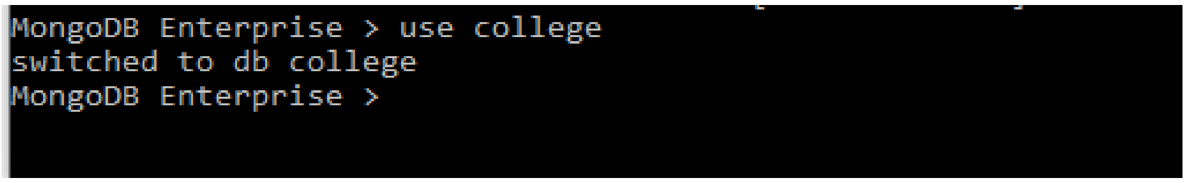
a. Write a MongoDB query to create and drop database.

Syntax:-

```
use DATABASE_NAME
```

Query:-

>use college



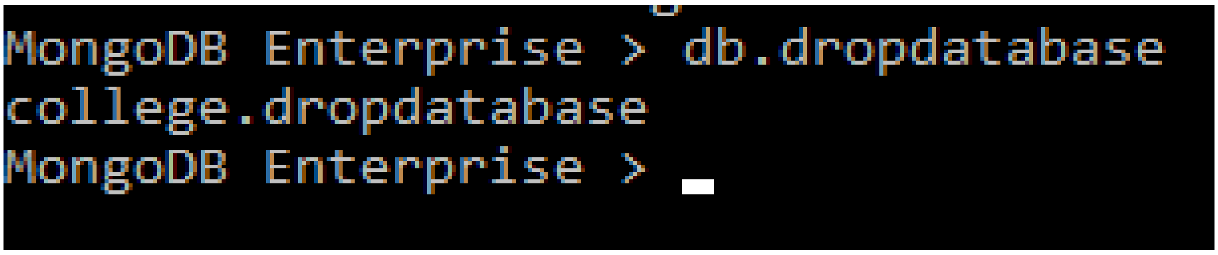
```
MongoDB Enterprise > use college
switched to db college
MongoDB Enterprise >
```

Syntax:-

```
db.dropDatabase()
```

Query:-

> db.dropDatabase()



```
MongoDB Enterprise > db.dropdatabase
college.dropdatabase
MongoDB Enterprise > _
```

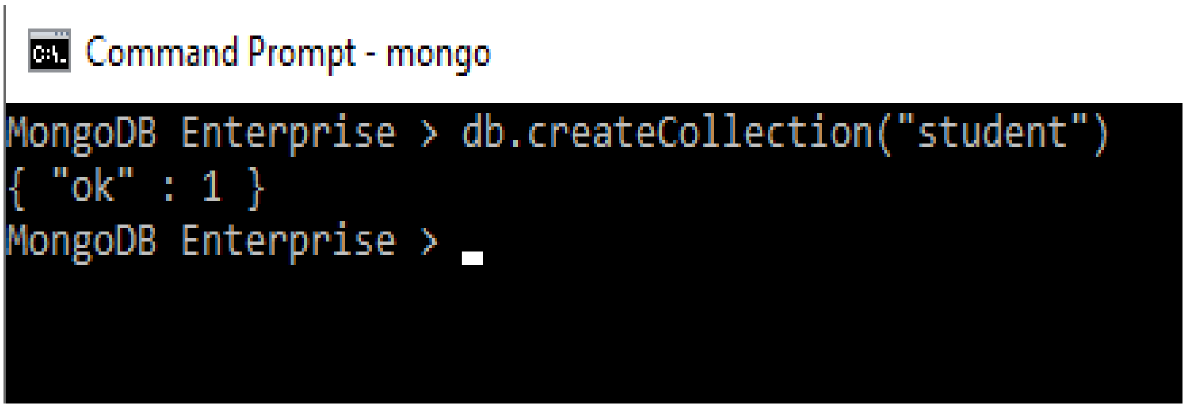
b. Write a MongoDB query to create, display and drop collection

Syntax:-

```
db.createCollection(name, options)
```

Query:-

```
>db.createCollection("student")
```



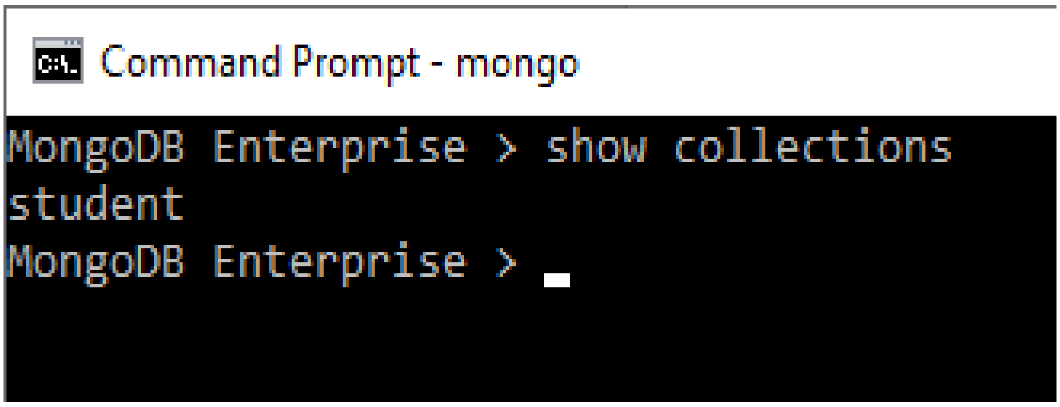
```
MongoDB Enterprise > db.createCollection("student")
{ "ok" : 1 }
MongoDB Enterprise > _
```

Syntax:-

```
show collections
```

Query:-

```
>show collections
```



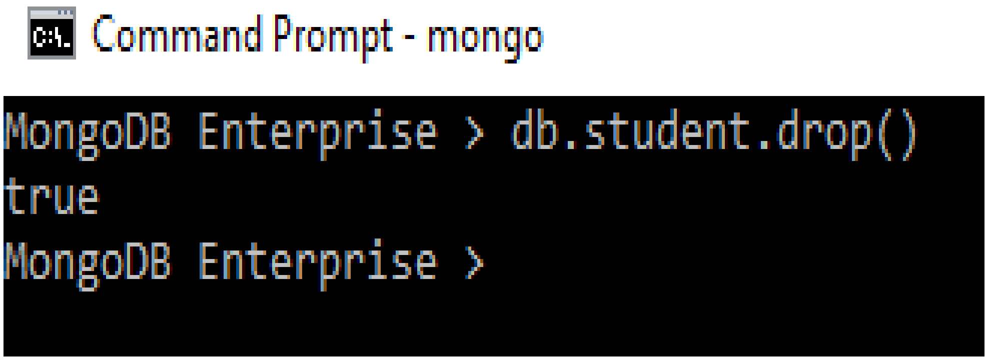
```
MongoDB Enterprise > show collections
student
MongoDB Enterprise > _
```

Syntax:-

```
db.COLLECTION_NAME.drop()
```

Query:-

>db.student.drop()



```
MongoDB Enterprise > db.student.drop()
true
MongoDB Enterprise >
```

c. **Write a MongoDB query to insert, query, update and delete a document.**

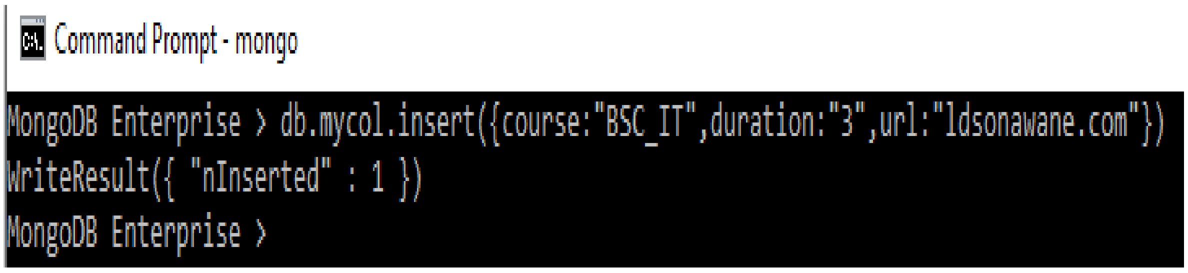
INSERT DOCUMENT

Syntax:-

```
db.COLLECTION_NAME.insert(document)
```

Query:-

>db.mycol.insert({course:"BSC_IT",duration:"3", url:"ldsonawane.com"})



```
Command Prompt - mongo
MongoDB Enterprise > db.mycol.insert({course:"BSC_IT",duration:"3",url:"ldsonawane.com"})
WriteResult({ "nInserted" : 1 })
MongoDB Enterprise >
```

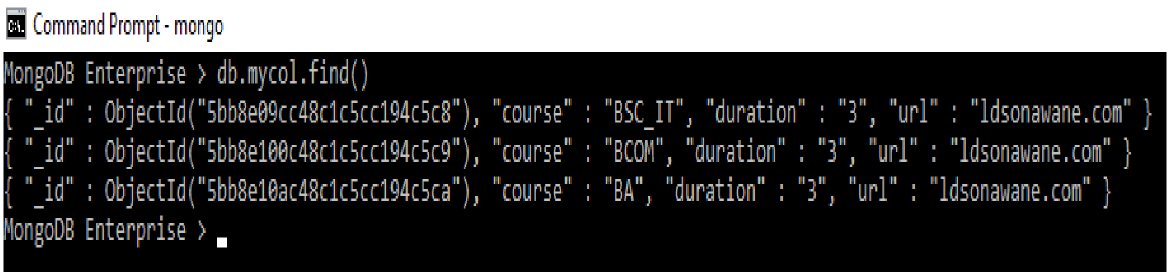
QUERY DOCUMENT

Syntax:-

```
>db.COLLECTION_NAME.find()
```

Query:-

>db.mycol.find()


Command Prompt - mongo

```
MongoDB Enterprise > db.mycol.find()
{ "_id" : ObjectId("5bb8e09cc48c1c5cc194c5c8"), "course" : "BSC_IT", "duration" : "3", "url" : "ldsonawane.com" }
{ "_id" : ObjectId("5bb8e100c48c1c5cc194c5c9"), "course" : "BCOM", "duration" : "3", "url" : "ldsonawane.com" }
{ "_id" : ObjectId("5bb8e10ac48c1c5cc194c5ca"), "course" : "BA", "duration" : "3", "url" : "ldsonawane.com" }
MongoDB Enterprise >
```

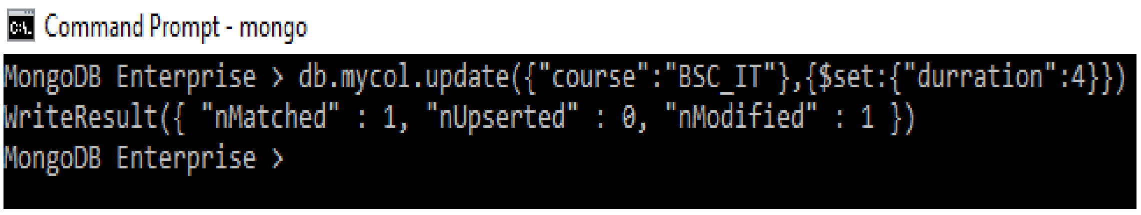
UPDATE DOCUMENT

Syntax:-

```
>db.COLLECTION_NAME.update(SELECTION_CRITERIA,
UPDATED_DATA)
```

Query:-

>db.mycol.update({"course":"BSC_IT"},{$set:{"durration":4}})


Command Prompt - mongo

```
MongoDB Enterprise > db.mycol.update({"course":"BSC_IT"},{$set:{"durration":4}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
MongoDB Enterprise >
```

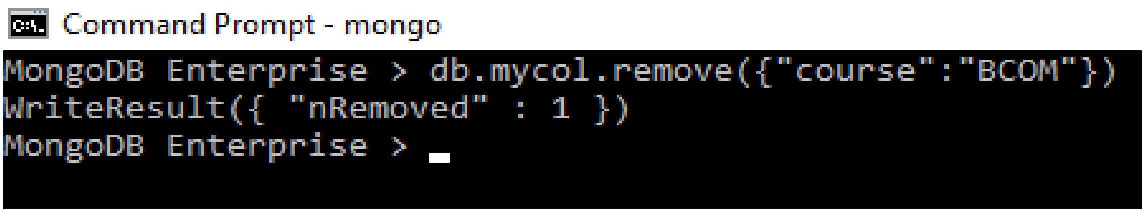
DELETE DOCUMENT

Syntax:-

```
>db.COLLECTION_NAME.remove(DELLETION_CRITTERIA)
```

Query:-

>db.mycol.remove({"course":"BCOM"})



```
MongoDB Enterprise > db.mycol.remove({"course":"BCOM"})
WriteResult({ "nRemoved" : 1 })
MongoDB Enterprise > _
```

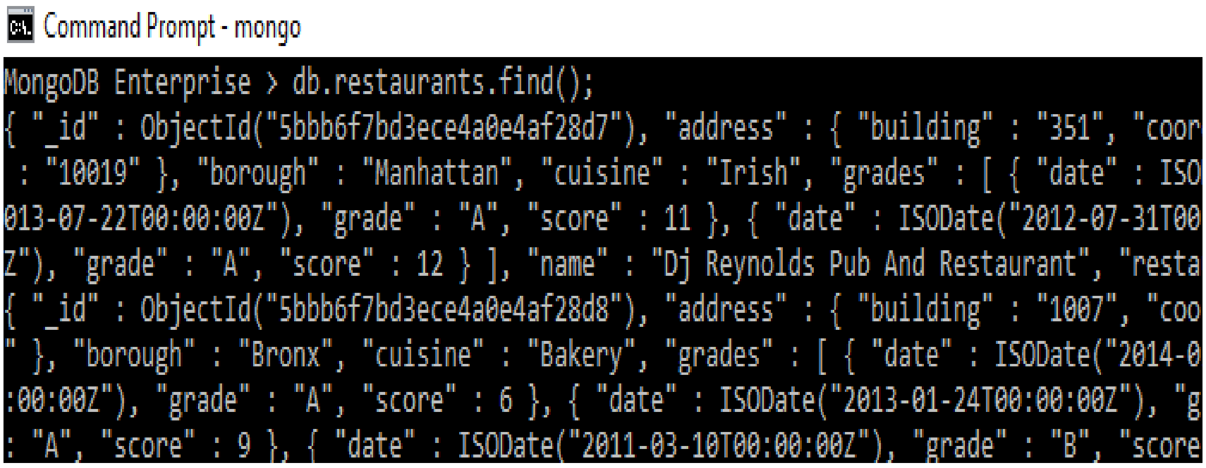
PRACTICAL NO:-2

➢SIMPLE MONGODB QUERIES

- Write a mongo DB query to display all the documents in the collection restaurants:

Query:-

```
db.restaurants.find()
```



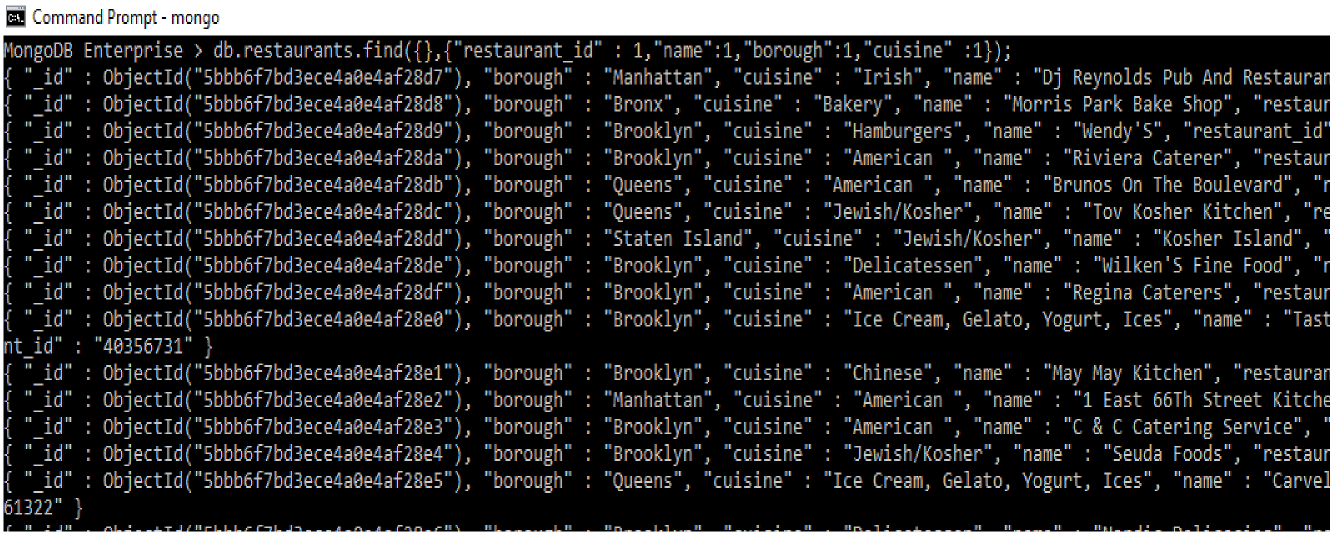
Command Prompt - mongo

```
MongoDB Enterprise > db.restaurants.find();
{ "_id" : ObjectId("5bbb6f7bd3ece4a0e4af28d7"), "address" : { "building" : "351", "coor
 : "10019" }, "borough" : "Manhattan", "cuisine" : "Irish", "grades" : [ { "date" : ISO
013-07-22T00:00:00Z"), "grade" : "A", "score" : 11 }, { "date" : ISODate("2012-07-31T00
Z"), "grade" : "A", "score" : 12 } ], "name" : "Dj Reynolds Pub And Restaurant", "resta
{ "_id" : ObjectId("5bbb6f7bd3ece4a0e4af28d8"), "address" : { "building" : "1007", "coo
" }, "borough" : "Bronx", "cuisine" : "Bakery", "grades" : [ { "date" : ISODate("2014-0
:00:00Z"), "grade" : "A", "score" : 6 }, { "date" : ISODate("2013-01-24T00:00:00Z"), "g
: "A", "score" : 9 }, { "date" : ISODate("2011-03-10T00:00:00Z"), "grade" : "B", "score
```

- Write a MongoDB query to display the fields restaurant_id, name, borough and cuisine for all the documents in the collection restaurant.

Query:-

```
        db.restaurants.find({},{"restaurant_id" :
1,"name":1,"borough":1,"cuisine" :1});
```



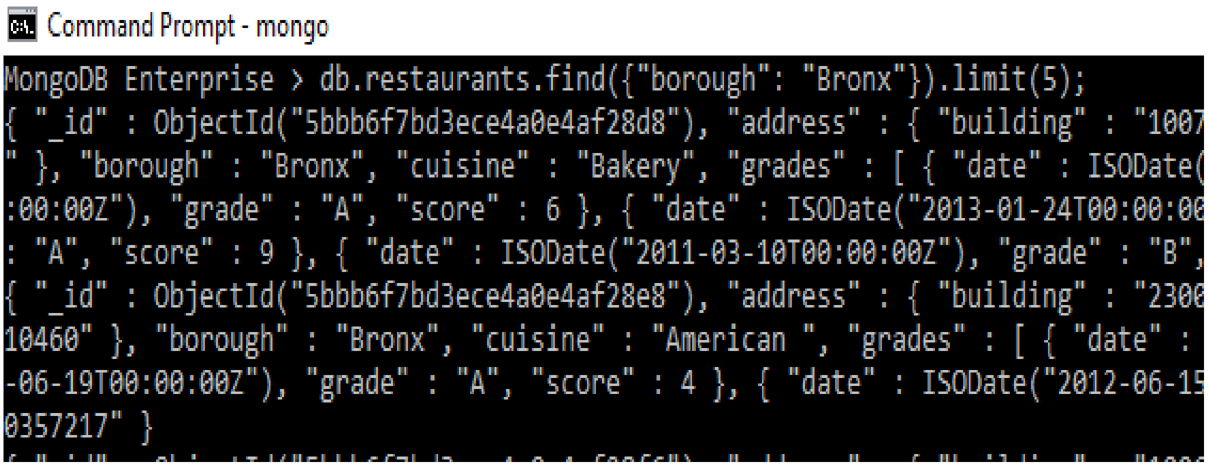
Command Prompt - mongo

```
MongoDB Enterprise > db.restaurants.find({},{"restaurant_id" : 1,"name":1,"borough":1,"cuisine" :1});
{ "_id" : ObjectId("5bbb6f7bd3ece4a0e4af28d7"), "borough" : "Manhattan", "cuisine" : "Irish", "name" : "Dj Reynolds Pub And Restauran
{ "_id" : ObjectId("5bbb6f7bd3ece4a0e4af28d8"), "borough" : "Bronx", "cuisine" : "Bakery", "name" : "Morris Park Bake Shop", "restaur
{ "_id" : ObjectId("5bbb6f7bd3ece4a0e4af28d9"), "borough" : "Brooklyn", "cuisine" : "Hamburgers", "name" : "Wendy'S", "restaurant_id"
{ "_id" : ObjectId("5bbb6f7bd3ece4a0e4af28da"), "borough" : "Brooklyn", "cuisine" : "American ", "name" : "Riviera Caterer", "restaur
{ "_id" : ObjectId("5bbb6f7bd3ece4a0e4af28db"), "borough" : "Queens", "cuisine" : "American ", "name" : "Brunos On The Boulevard", "r
{ "_id" : ObjectId("5bbb6f7bd3ece4a0e4af28dc"), "borough" : "Queens", "cuisine" : "Jewish/Kosher", "name" : "Tov Kosher Kitchen", "re
{ "_id" : ObjectId("5bbb6f7bd3ece4a0e4af28dd"), "borough" : "Staten Island", "cuisine" : "Jewish/Kosher", "name" : "Kosher Island", "
{ "_id" : ObjectId("5bbb6f7bd3ece4a0e4af28de"), "borough" : "Brooklyn", "cuisine" : "Delicatessen", "name" : "Wilken'S Fine Food", "r
{ "_id" : ObjectId("5bbb6f7bd3ece4a0e4af28df"), "borough" : "Brooklyn", "cuisine" : "American ", "name" : "Regina Caterers", "restaur
{ "_id" : ObjectId("5bbb6f7bd3ece4a0e4af28e0"), "borough" : "Brooklyn", "cuisine" : "Ice Cream, Gelato, Yogurt, Ices", "name" : "Tast
nt_id" : "40356731" }
{ "_id" : ObjectId("5bbb6f7bd3ece4a0e4af28e1"), "borough" : "Brooklyn", "cuisine" : "Chinese", "name" : "May May Kitchen", "restauran
{ "_id" : ObjectId("5bbb6f7bd3ece4a0e4af28e2"), "borough" : "Manhattan", "cuisine" : "American ", "name" : "1 East 66Th Street Kitche
{ "_id" : ObjectId("5bbb6f7bd3ece4a0e4af28e3"), "borough" : "Brooklyn", "cuisine" : "American ", "name" : "C & C Catering Service", "
{ "_id" : ObjectId("5bbb6f7bd3ece4a0e4af28e4"), "borough" : "Brooklyn", "cuisine" : "Jewish/Kosher", "name" : "Seuda Foods", "restaur
{ "_id" : ObjectId("5bbb6f7bd3ece4a0e4af28e5"), "borough" : "Queens", "cuisine" : "Ice Cream, Gelato, Yogurt, Ices", "name" : "Carvel
61322" }
```

- Write a MongoDB query to display the first 5 restaurant which is in the borough Bronx

Query:-

```
        db.restaurants.find({"borough": "Bronx"}).limit(5);
```



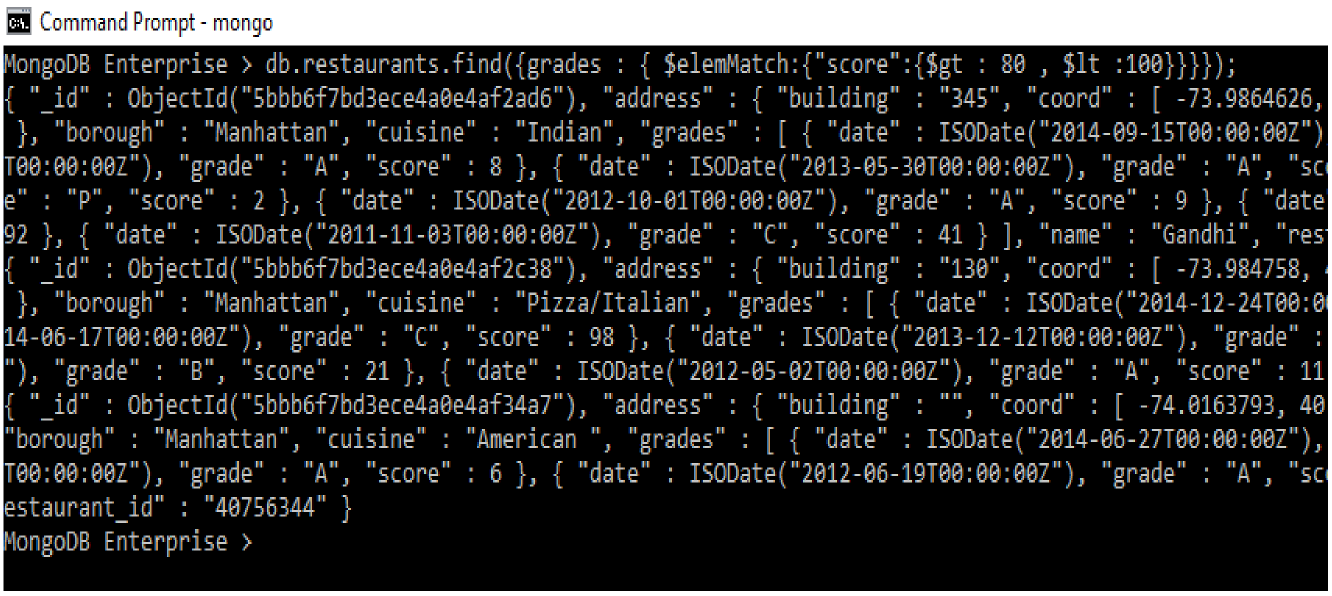
Command Prompt - mongo

```
MongoDB Enterprise > db.restaurants.find({"borough": "Bronx"}).limit(5);
{ "_id" : ObjectId("5bbb6f7bd3ece4a0e4af28d8"), "address" : { "building" : "1007
" }, "borough" : "Bronx", "cuisine" : "Bakery", "grades" : [ { "date" : ISODate(
:00:00Z"), "grade" : "A", "score" : 6 }, { "date" : ISODate("2013-01-24T00:00:00
: "A", "score" : 9 }, { "date" : ISODate("2011-03-10T00:00:00Z"), "grade" : "B",
{ "_id" : ObjectId("5bbb6f7bd3ece4a0e4af28e8"), "address" : { "building" : "2300
10460" }, "borough" : "Bronx", "cuisine" : "American ", "grades" : [ { "date" :
-06-19T00:00:00Z"), "grade" : "A", "score" : 4 }, { "date" : ISODate("2012-06-15
0357217" }
```

- Write a MongoDB query to find the restaurants that achieved a score is more than 80 but less than 100

Query:-

db.restaurants.find({grades : { $elemMatch:{"score":{$gt : 80 , $lt :100}}}});



Command Prompt - mongo

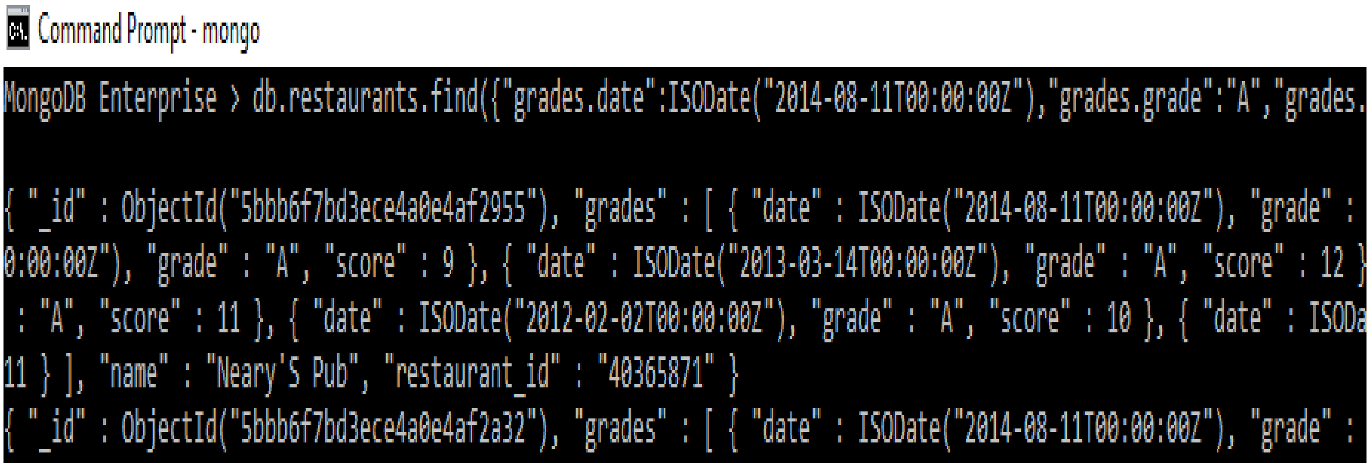
```
MongoDB Enterprise > db.restaurants.find({grades : { $elemMatch:{"score":{$gt : 80 , $lt :100}}}});
{ "_id" : ObjectId("5bbb6f7bd3ece4a0e4af2ad6"), "address" : { "building" : "345", "coord" : [ -73.9864626,
 }, "borough" : "Manhattan", "cuisine" : "Indian", "grades" : [ { "date" : ISODate("2014-09-15T00:00:00Z")
T00:00:00Z"), "grade" : "A", "score" : 8 }, { "date" : ISODate("2013-05-30T00:00:00Z"), "grade" : "A", "sc
e" : "P", "score" : 2 }, { "date" : ISODate("2012-10-01T00:00:00Z"), "grade" : "A", "score" : 9 }, { "date
92 }, { "date" : ISODate("2011-11-03T00:00:00Z"), "grade" : "C", "score" : 41 } ], "name" : "Gandhi", "res
{ "_id" : ObjectId("5bbb6f7bd3ece4a0e4af2c38"), "address" : { "building" : "130", "coord" : [ -73.984758,
 }, "borough" : "Manhattan", "cuisine" : "Pizza/Italian", "grades" : [ { "date" : ISODate("2014-12-24T00:0
14-06-17T00:00:00Z"), "grade" : "C", "score" : 98 }, { "date" : ISODate("2013-12-12T00:00:00Z"), "grade" :
"), "grade" : "B", "score" : 21 }, { "date" : ISODate("2012-05-02T00:00:00Z"), "grade" : "A", "score" : 11
{ "_id" : ObjectId("5bbb6f7bd3ece4a0e4af34a7"), "address" : { "building" : "", "coord" : [ -74.0163793, 40
"borough" : "Manhattan", "cuisine" : "American ", "grades" : [ { "date" : ISODate("2014-06-27T00:00:00Z"),
T00:00:00Z"), "grade" : "A", "score" : 6 }, { "date" : ISODate("2012-06-19T00:00:00Z"), "grade" : "A", "sc
estaurant_id" : "40756344" }
MongoDB Enterprise >
```

- Write a MongoDB query to find the restaurant Id, name, and grades for those restaurants which achieved a grade of "A" and scored 11 on an

ISODate "2014-08-11T00:00:00Z" among many of survey dates.

Query:-

```
        db.restaurants.find({"grades.date":ISODate("2014-08-11T00:00:00Z"),"grades.grade":"A","grades.score" :11},{"restaurant_id":1,"name":1,"grades":1});
```



```
Command Prompt - mongo
MongoDB Enterprise > db.restaurants.find({"grades.date":ISODate("2014-08-11T00:00:00Z"),"grades.grade":"A","grades.

{ "_id" : ObjectId("5bbb6f7bd3ece4a0e4af2955"), "grades" : [ { "date" : ISODate("2014-08-11T00:00:00Z"), "grade" :
0:00:00Z"), "grade" : "A", "score" : 9 }, { "date" : ISODate("2013-03-14T00:00:00Z"), "grade" : "A", "score" : 12 }
 : "A", "score" : 11 }, { "date" : ISODate("2012-02-02T00:00:00Z"), "grade" : "A", "score" : 10 }, { "date" : ISODa
11 } ], "name" : "Neary'S Pub", "restaurant_id" : "40365871" }
{ "_id" : ObjectId("5bbb6f7bd3ece4a0e4af2a32"), "grades" : [ { "date" : ISODate("2014-08-11T00:00:00Z"), "grade" :
```

PRACTICAL NO:-3

➢Implementing Aggregation

a. **Write a MongoDB query to use sum, avg, min and max expression.**

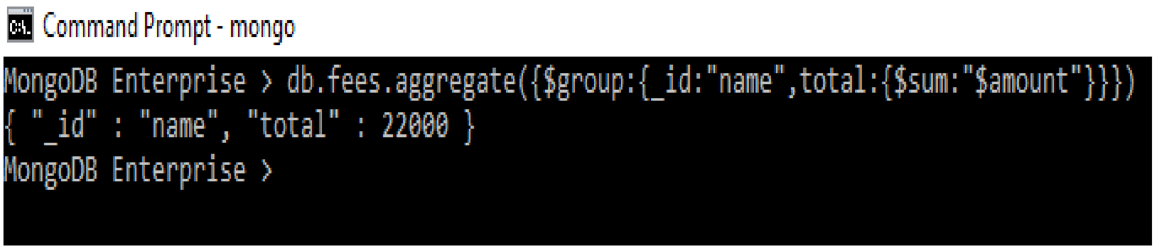
SUM

Syntax:-

```
>db.COLLECTION_NAME.aggregate(AGGREGATE_OPERATION)
```

Query:-

```
db.fees.aggregate({$group:{_id:"name",total:{$sum:"$amount"}}})
```



Command Prompt - mongo

```
MongoDB Enterprise > db.fees.aggregate({$group:{_id:"name",total:{$sum:"$amount"}}})
{ "_id" : "name", "total" : 22000 }
MongoDB Enterprise >
```

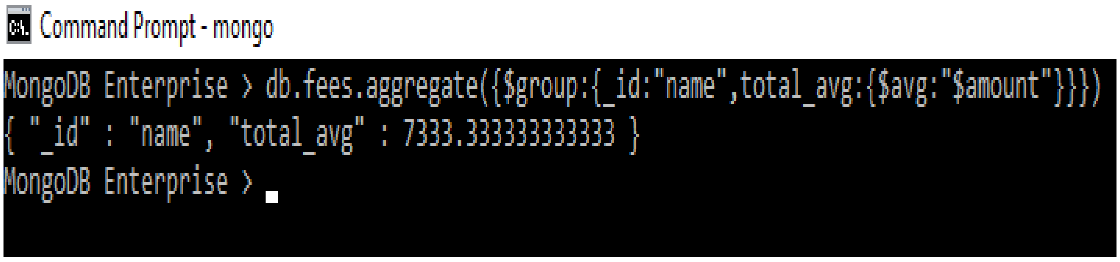
AVG

Syntax:-

```
>db.COLLECTION_NAME.aggregate([{$group : {_id : "$by_user",
num_tutorial : {$avg : "$likes"}}}])
```

Query:-

db.fees.aggregate({$group:{_id:"name",total_avg:{$avg:"$amount"}}})



```
Command Prompt - mongo
MongoDB Enterprise > db.fees.aggregate({$group:{_id:"name",total_avg:{$avg:"$amount"}}})
{ "_id" : "name", "total_avg" : 7333.333333333333 }
MongoDB Enterprise >
```

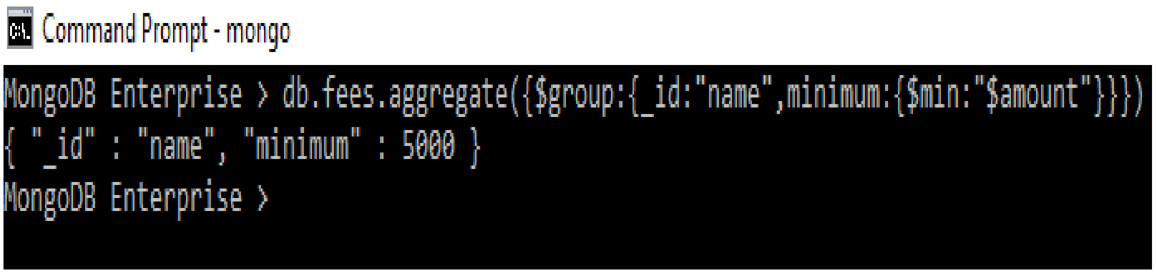
MIN

Syntax:-

```
>db.COLLECTION_NAME.aggregate[{$group : {_id : "$by_user",
num_tutorial : {$min : "$likes"}}}])
```

Query:-

db.fees.aggregate({$group:{_id:"name",minimum:{$min:"$amount"}}})



```
Command Prompt - mongo
MongoDB Enterprise > db.fees.aggregate({$group:{_id:"name",minimum:{$min:"$amount"}}})
{ "_id" : "name", "minimum" : 5000 }
MongoDB Enterprise >
```

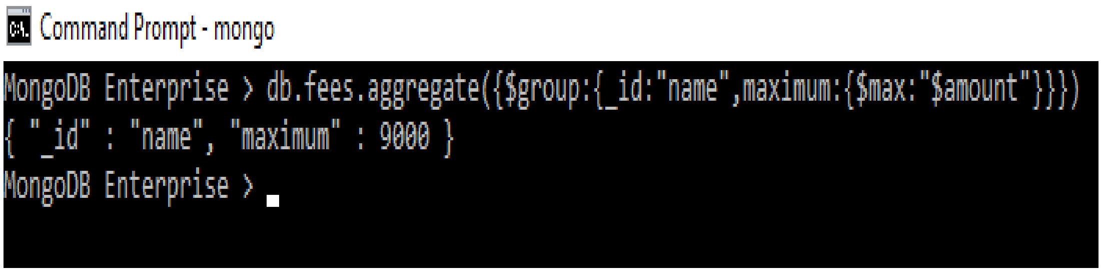
MAX

Syntax:-

```
>db.COLLECTION_NAME.aggregate([{$group : {_id : "$by_user", num_tutorial : {$max : "$likes"}}}])
```

Query:-

```
db.fees.aggregate({$group:{_id:"name",maximum:{$max:"$amount"}}})
```



```
MongoDB Enterprise > db.fees.aggregate({$group:{_id:"name",maximum:{$max:"$amount"}}})
{ "_id" : "name", "maximum" : 9000 }
MongoDB Enterprise >
```

b. **Write a MongoDB query to use push and addToSet expression.**

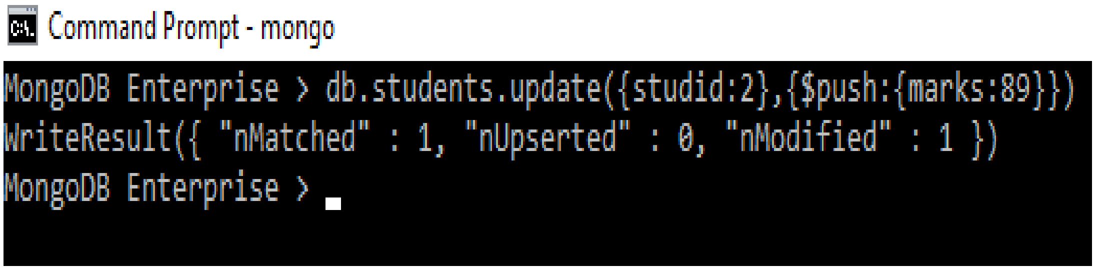
PUSH

Syntax:-

```
>db.COLLECTION_NAME.update({ $push: { <field1>: <value1>, ... } })
```

Query:-

db.students.update({ name: “sid” },{ $push: { marks: 89 } })


Command Prompt - mongo

```
MongoDB Enterprise > db.students.update({studid:2},{$push:{marks:89}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
MongoDB Enterprise > _
```

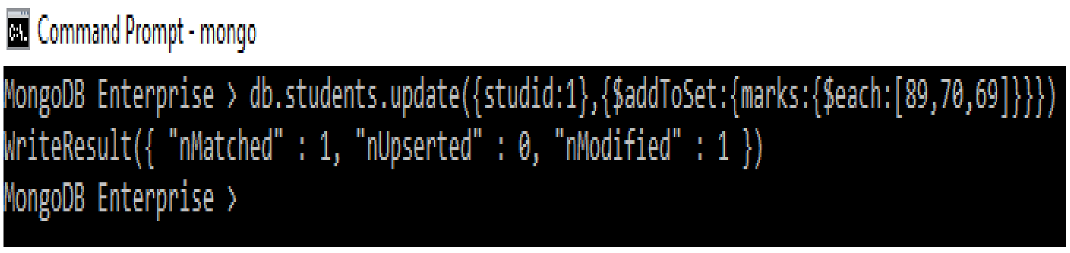
ADDTOSET

Syntax:-

```
>db.COLLECTION_NAME.update({  <field> :  <value>  } ,  { $addtoset:
{ <field1>: <addition> } })
```

Query:-

db.students.update({ name: “sid” },{ $addToSet: { marks:{$each:[89,70,69]} } })


Command Prompt - mongo

```
MongoDB Enterprise > db.students.update({studid:1},{$addToSet:{marks:{$each:[89,70,69]}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
MongoDB Enterprise >
```

c. **Write a MongoDB query to use first and last expression**

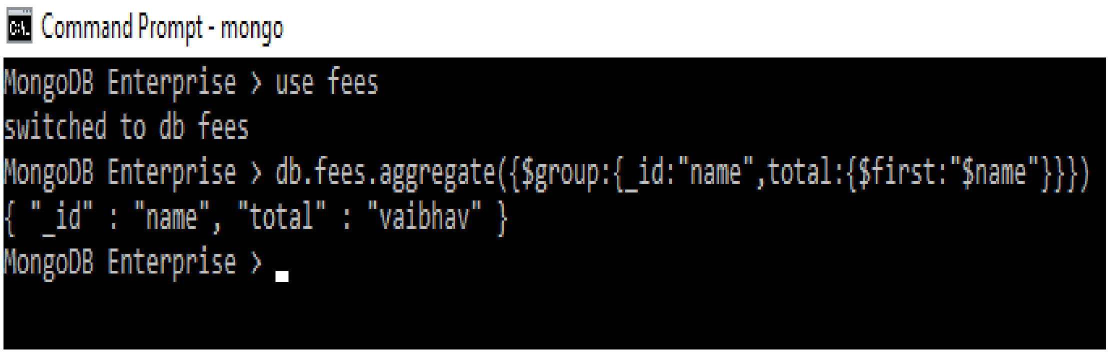
FIRST

Syntax:-

```
>db.COLLECTION_NAME.aggregate([$group : {_id: “$by_user”, first url : {$first : “$url”} } })
```

Query:-

db.fees.aggregate({$group:{_id:"name",total:{$first:"$name"}}}
)



```
MongoDB Enterprise > use fees
switched to db fees
MongoDB Enterprise > db.fees.aggregate({$group:{_id:"name",total:{$first:"$name"}}})
{ "_id" : "name", "total" : "vaibhav" }
MongoDB Enterprise >
```

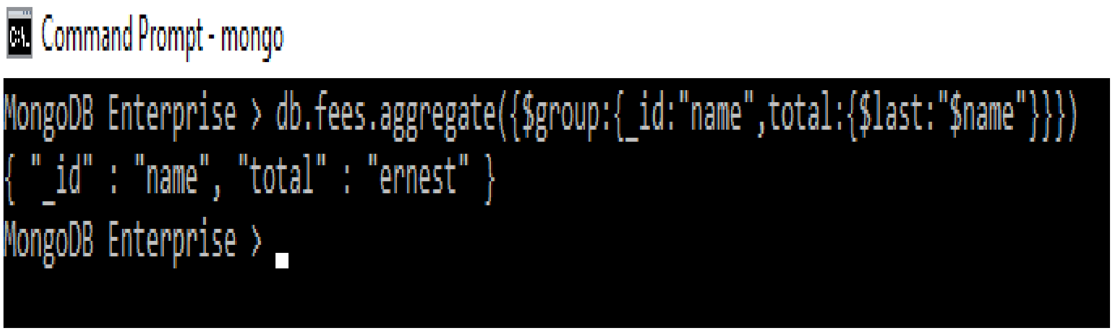
LAST

Syntax:-

> >db.COLLECTION_NAME.aggregate([$group : {_id: “$by_user”, last ur l : {$last : “$url”} } })

Query:-

db.fees.aggregate({$group:{_id:"name",total:{$last:"$name"}}})



```
MongoDB Enterprise > db.fees.aggregate({$group:{_id:"name",total:{$last:"$name"}}})
{ "_id" : "name", "total" : "ernest" }
MongoDB Enterprise >
```

PRACTICAL NO :- 4

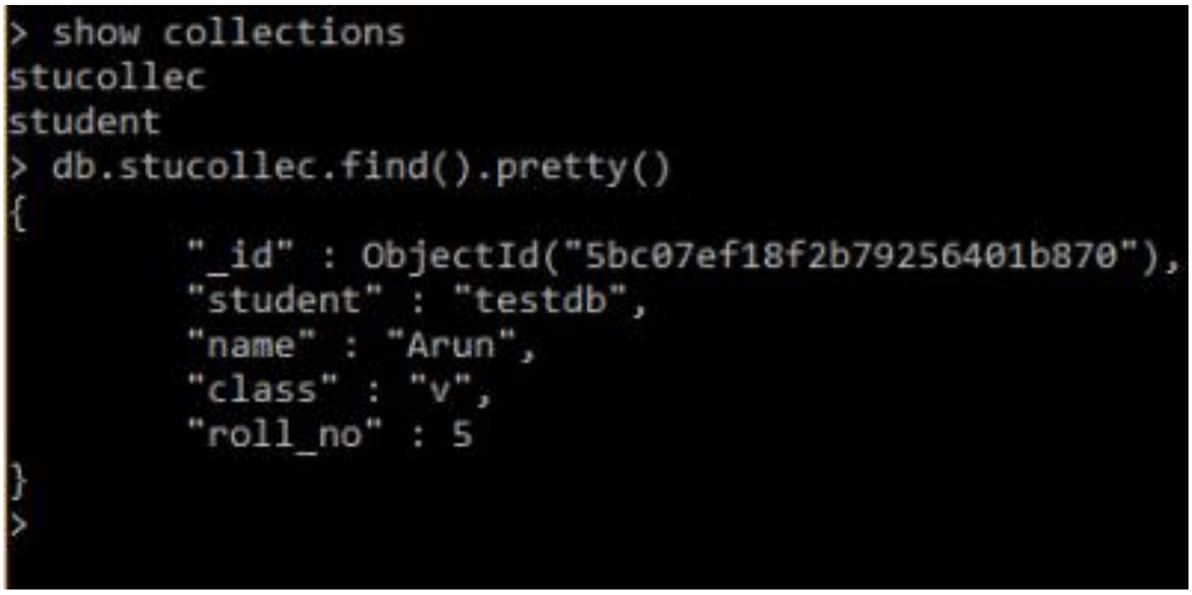
➢JAVA AND MONGODB

a. Connecting Java with MongoDB and inserting, retrieving, updating and deleting

INSERT

Query:-

```
public class Mongodb_connection_insert
{
  public static void main(String args[])
   {
     try{
       MongoClient mongoclient =
               new MongoClient("localhost",27017);
       DB db=mongoClient.getDB("testdb");
       System.out.println("Connection successful");
       DBCollection collec=db.getCollection("stucollec");
       BasicDBObject doc= new
       BasicDBObject("student","testdb").append("name","Arun
       ").apped("class","v").append("roll_no",5);
       collec.insert(doc);
       System.out.println("inserted");}
       catch(Exception e)
         {
               System.err.println(e.getClass().getName()+"."+e.get
               Message())
       }
  }
}
```



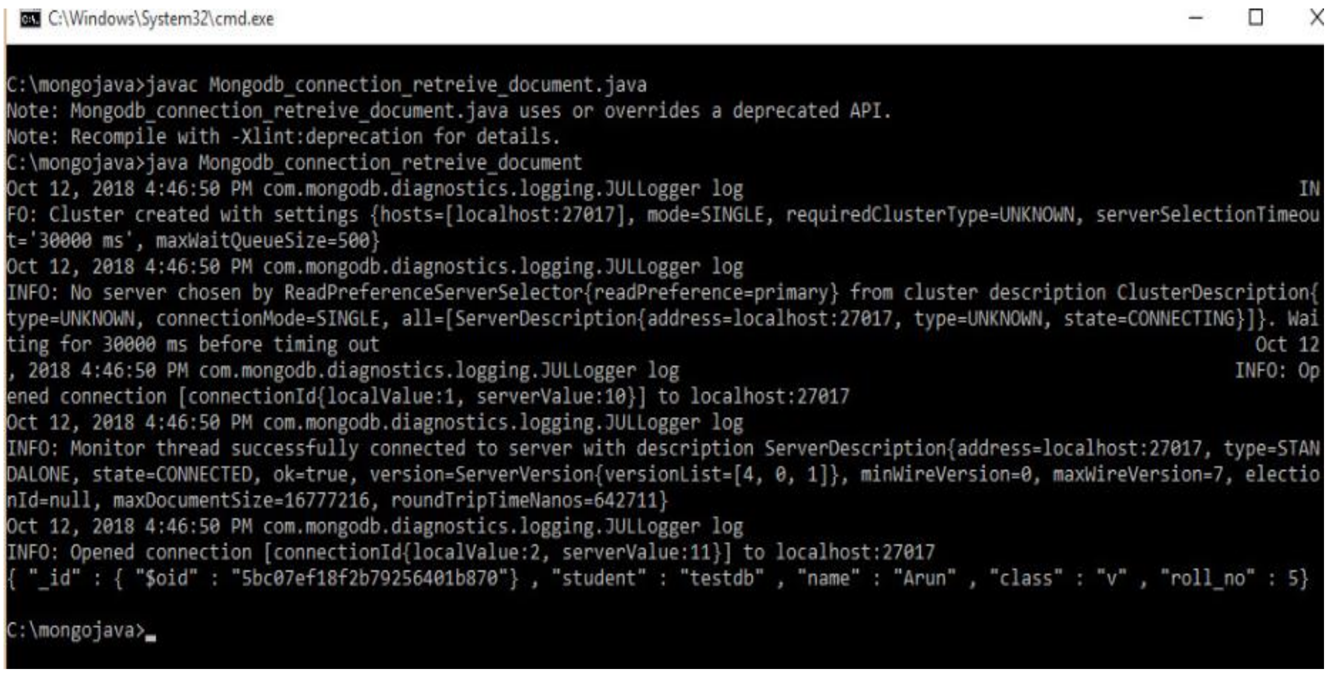
```
> show collections
stucollec
student
> db.stucollec.find().pretty()
{
        "_id" : ObjectId("5bc07ef18f2b79256401b870"),
        "student" : "testdb",
        "name" : "Arun",
        "class" : "v",
        "roll_no" : 5
}
>
```

RETRIVE

Query:-

```
import com.mongodb.MongoClient;
import com.mongodb.MongoException;
import com.mongodb.WriteConcern;
import com.mongodb.DB;
import com.mongodb.DBCollection;
import com.mongodb.BasicDBObject;
import com.mongodb.DBObject;
import com.mongodb.DBCursor;
import com.mongodb.ServerAddress;
```

```
        import java.util.Arrays;
        public class Mongodb_connection_find_all_documents
        {
                public static void main(String args[])
                        {
                        try{
                        MongoClient mongoclient=new
    MongoClient("localhost",27017);
                        DB db=mongoClient.getDB("testdb");
                        DBCollection collec=db.getCollection("stucollec");
                        DBCursor cursor=collec.find();
                                try
                                        {
                                                While(cursor.hasNext())
                                        {
                                                System.out.println(cursor.next());
                                        }
                                                } finally
                                        {
                                                cursor.close();
                                                        }
                                                } catch(Exception e)
                                        {

        System.err.println(e.getClass().getName()+"."+e.getMessage());
                                                }
                                                        }
                                                                }
```



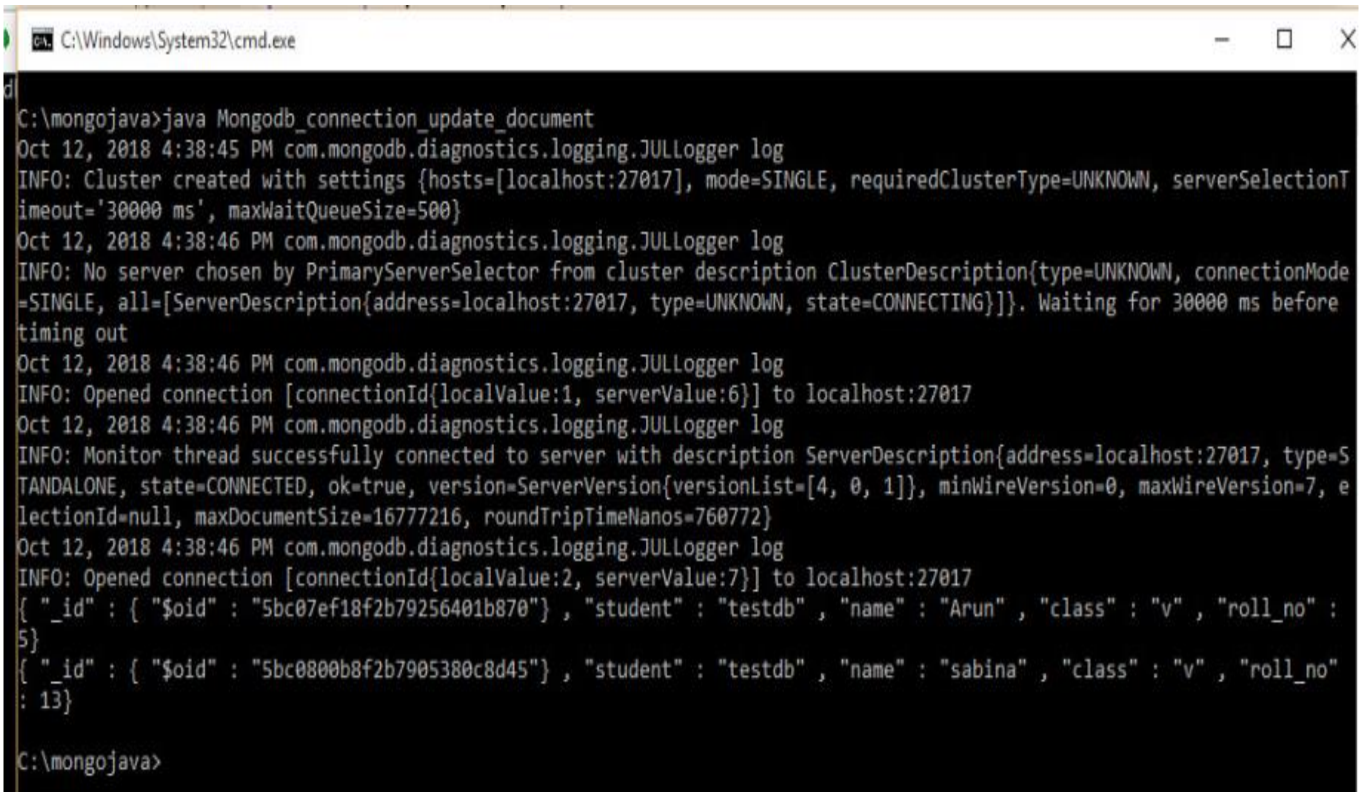
```
C:\mongojava>javac Mongodb_connection_retreive_document.java
Note: Mongodb_connection_retreive_document.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
C:\mongojava>java Mongodb_connection_retreive_document
Oct 12, 2018 4:46:50 PM com.mongodb.diagnostics.logging.JULLogger log                                        IN
FO: Cluster created with settings {hosts=[localhost:27017], mode=SINGLE, requiredClusterType=UNKNOWN, serverSelectionTimeou
t='30000 ms', maxWaitQueueSize=500}
Oct 12, 2018 4:46:50 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: No server chosen by ReadPreferenceServerSelector{readPreference=primary} from cluster description ClusterDescription{
type=UNKNOWN, connectionMode=SINGLE, all=[ServerDescription{address=localhost:27017, type=UNKNOWN, state=CONNECTING}]}. Wai
ting for 30000 ms before timing out                                                                          Oct 12
, 2018 4:46:50 PM com.mongodb.diagnostics.logging.JULLogger log                                              INFO: Op
ened connection [connectionId{localValue:1, serverValue:10}] to localhost:27017
Oct 12, 2018 4:46:50 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Monitor thread successfully connected to server with description ServerDescription{address=localhost:27017, type=STAN
DALONE, state=CONNECTED, ok=true, version=ServerVersion{versionList=[4, 0, 1]}, minWireVersion=0, maxWireVersion=7, electio
nId=null, maxDocumentSize=16777216, roundTripTimeNanos=642711}
Oct 12, 2018 4:46:50 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Opened connection [connectionId{localValue:2, serverValue:11}] to localhost:27017
{ "_id" : { "$oid" : "5bc07ef18f2b79256401b870"} , "student" : "testdb" , "name" : "Arun" , "class" : "v" , "roll_no" : 5}

C:\mongojava>_
```

UPDATE

Query:-

```
public class Mongodb_connection_update_document
{
 public static void main(String args[])
   {
     try{
          MongoClient mongoclient =new
          MongoClient("localhost",27017);
          DB db=mongoClient.getDB("testdb");
```

```
                DBCollection collec=db.getCollection("stucollec");
                DBObject query=new BasicDBObject(“name”,”sabina”);
                DBObject update=new BasicDBObject();
                Update.put(“$set”,new BasicDBObject(“roll_no”,13));
                WriteResult result=collec.update(query,update);
                DBCursor cursor=collec.find();
        try{
                while(cursor.hasNext())
        {
                System.out.println(cursor.next());
        }
}
finally
        {       cursor close();
        }
}
        catch(Exception e)
        {
        System.err.println(e.getClass().getName()+"."+e.getMessage());
        }
    }
}
```



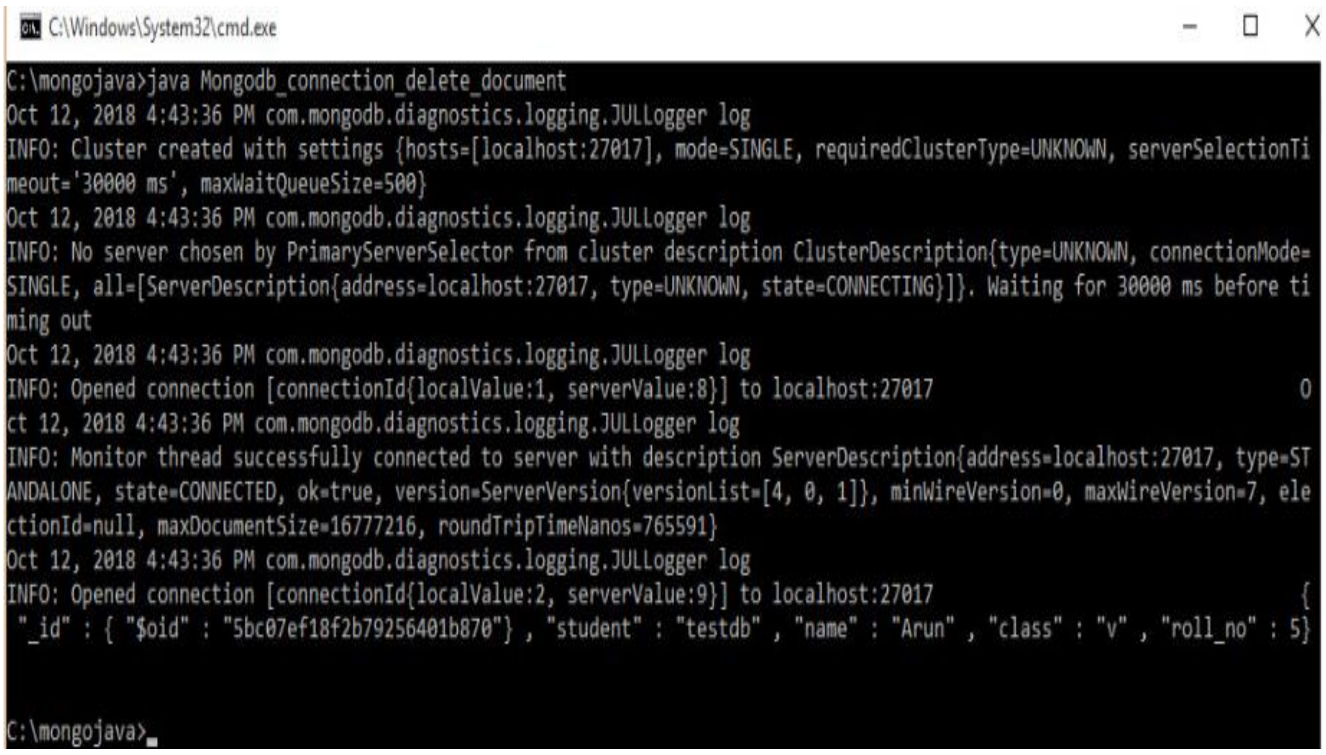
```
C:\mongojava>java Mongodb_connection_update_document
Oct 12, 2018 4:38:45 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Cluster created with settings {hosts=[localhost:27017], mode=SINGLE, requiredClusterType=UNKNOWN, serverSelectionT
imeout='30000 ms', maxWaitQueueSize=500}
Oct 12, 2018 4:38:46 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: No server chosen by PrimaryServerSelector from cluster description ClusterDescription{type=UNKNOWN, connectionMode
=SINGLE, all=[ServerDescription{address=localhost:27017, type=UNKNOWN, state=CONNECTING}]}. Waiting for 30000 ms before
timing out
Oct 12, 2018 4:38:46 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Opened connection [connectionId{localValue:1, serverValue:6}] to localhost:27017
Oct 12, 2018 4:38:46 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Monitor thread successfully connected to server with description ServerDescription{address=localhost:27017, type=S
TANDALONE, state=CONNECTED, ok=true, version=ServerVersion{versionList=[4, 0, 1]}, minWireVersion=0, maxWireVersion=7, e
lectionId=null, maxDocumentSize=16777216, roundTripTimeNanos=760772}
Oct 12, 2018 4:38:46 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Opened connection [connectionId{localValue:2, serverValue:7}] to localhost:27017
{ "_id" : { "$oid" : "5bc07ef18f2b79256401b870"} , "student" : "testdb" , "name" : "Arun" , "class" : "v" , "roll_no" :
5}
{ "_id" : { "$oid" : "5bc0800b8f2b7905380c8d45"} , "student" : "testdb" , "name" : "sabina" , "class" : "v" , "roll_no"
: 13}

C:\mongojava>
```

DELETE

Query:-

```
public class Mongodb_connection_delete_document
{
 public static void main(String args[])
   {
     try{
```

```
                MongoClient mongoclient =new
                MongoClient("localhost",27017);
                DB db=mongoClient.getDB("testdb");
                DBCollection collec=db.getCollection("stucollec");
                DBObject query=new BasicDBObject(“name”,”sabina”);
                DBObject update=new BasicDBObject();
                Update.put(“$set”,new BasicDBObject(“roll_no”,13));
                WriteResult result=collec.remove(query);
                DBCursor cursor=collec.find();
        try{
                while(cursor.hasNext())
        {
                System.out.println(cursor.next());
        }
}
finally
        {       cursor close();
        }
}
        catch(Exception e)
        {
        System.err.println(e.getClass().getName()+"."+e.getMessage());
        }
    }
}
```



```
C:\mongojava>java Mongodb_connection_delete_document
Oct 12, 2018 4:43:36 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Cluster created with settings {hosts=[localhost:27017], mode=SINGLE, requiredClusterType=UNKNOWN, serverSelectionTi
meout='30000 ms', maxWaitQueueSize=500}
Oct 12, 2018 4:43:36 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: No server chosen by PrimaryServerSelector from cluster description ClusterDescription{type=UNKNOWN, connectionMode=
SINGLE, all=[ServerDescription{address=localhost:27017, type=UNKNOWN, state=CONNECTING}]}. Waiting for 30000 ms before ti
ming out
Oct 12, 2018 4:43:36 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Opened connection [connectionId{localValue:1, serverValue:8}] to localhost:27017                                  O
ct 12, 2018 4:43:36 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Monitor thread successfully connected to server with description ServerDescription{address=localhost:27017, type=ST
ANDALONE, state=CONNECTED, ok=true, version=ServerVersion{versionList=[4, 0, 1]}, minWireVersion=0, maxWireVersion=7, ele
ctionId=null, maxDocumentSize=16777216, roundTripTimeNanos=765591}
Oct 12, 2018 4:43:36 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Opened connection [connectionId{localValue:2, serverValue:9}] to localhost:27017                                  {
 "_id" : { "$oid" : "5bc07ef18f2b79256401b870"} , "student" : "testdb" , "name" : "Arun" , "class" : "v" , "roll_no" : 5}


C:\mongojava>_
```

PRACTICAL NO :- 5

➢ PHP AND MONGODB

a. **Connecting PHP with MongoDB and inserting, retrieving, updating and deleting.**

CONNECTION

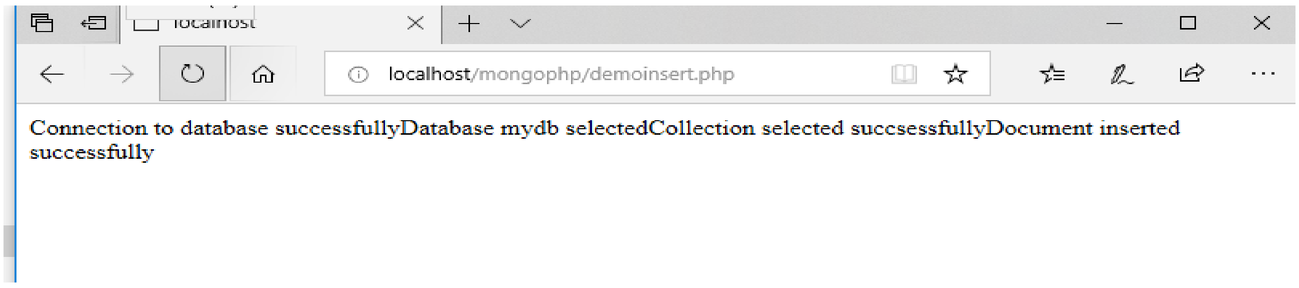
Query:-

```
<?php
  // connect to mongodb
  $m = new MongoClient();

  echo "Connection to database successfully";
  // select a database
  $db = $m->mydb;

  echo "Database mydb selected";
?>
```

INSERTING

Query:-

```
<?php
  // connect to mongodb
  $m = new MongoClient();
  echo "Connection to database successfully";

  // select a database
  $db = $m->mydb;
  echo "Database mydb selected";
  $collection = $db->mycol;
  echo "Collection selected succsessfully";

  $document = array(
    "title" => "MongoDB",
    "description" => "database",
    "likes" => 100,
    "url" => "http://www.tutorialspoint.com/mongodb/",
    "by" => "tutorials point"
  );

  $collection->insert($document);
  echo "Document inserted successfully";
?>
```



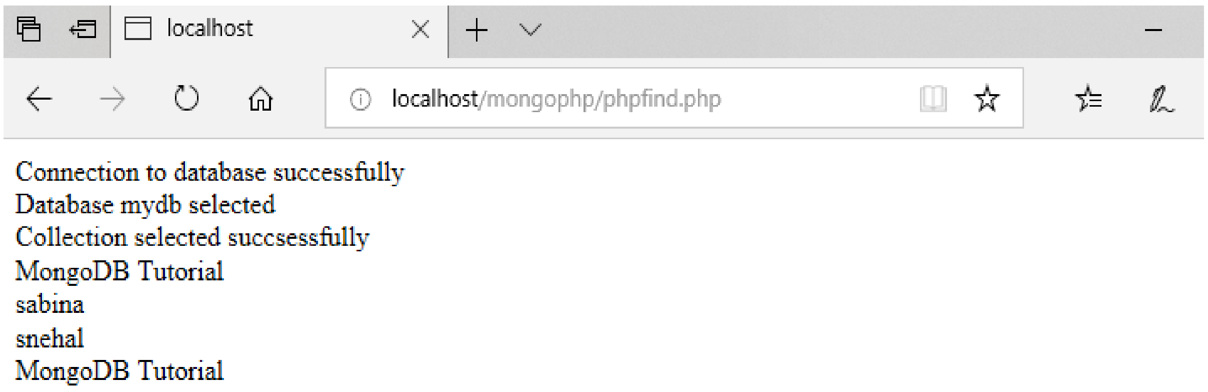
localhost/mongophp/demoinsert.php

```
Connection to database successfullyDatabase mydb selectedCollection selected succsessfullyDocument inserted
successfully
```

RETRIEVING

Query:-

```
<?php
  // connect to mongodb
  $m = new MongoClient();
  echo "Connection to database successfully";

  // select a database
  $db = $m->mydb;
  echo "Database mydb selected";
  $collection = $db->mycol;
  echo "Collection selected succsessfully";
  $cursor = $collection->find();
  // iterate cursor to display title of documents

  foreach ($cursor as $document) {
    echo $document["title"] . "\n";
  }
?>
```

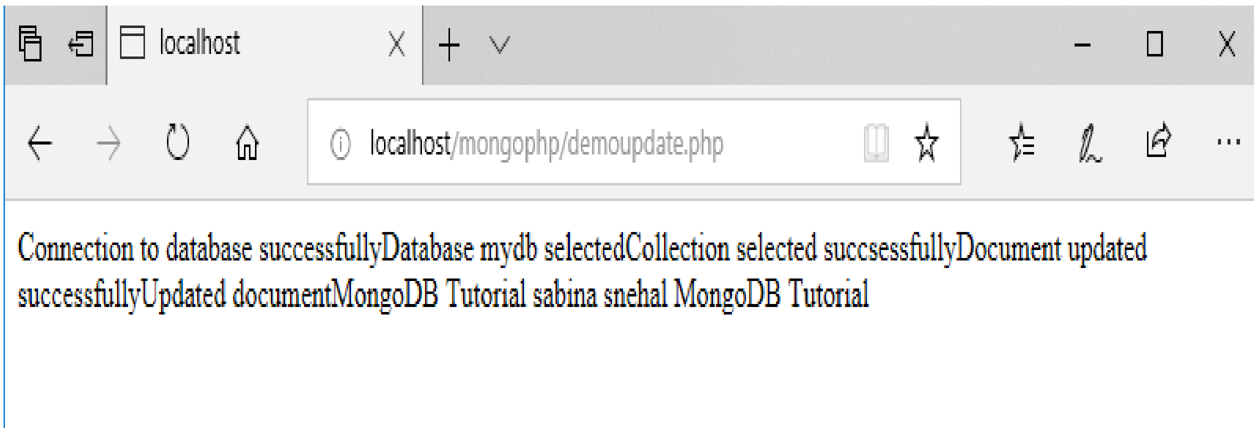


```
Connection to database successfully
Database mydb selected
Collection selected succsessfully
MongoDB Tutorial
sabina
snehal
MongoDB Tutorial
```

UPDATE

Query:-

```
<?php
  // connect to mongodb
  $m = new MongoClient();
  echo "Connection to database successfully";

  // select a database
  $db = $m->mydb;
  echo "Database mydb selected";
  $collection = $db->mycol;
  echo "Collection selected succsessfully";

  // now update the document
  $collection->update(array("title"=>"MongoDB"),
    array('$set'=>array("title"=>"MongoDB Tutorial")));
  echo "Document updated successfully";

  // now display the updated document
  $cursor = $collection->find();

  // iterate cursor to display title of documents
  echo "Updated document";

  foreach ($cursor as $document) {
    echo $document["title"] . "\n";
  }
?>
```

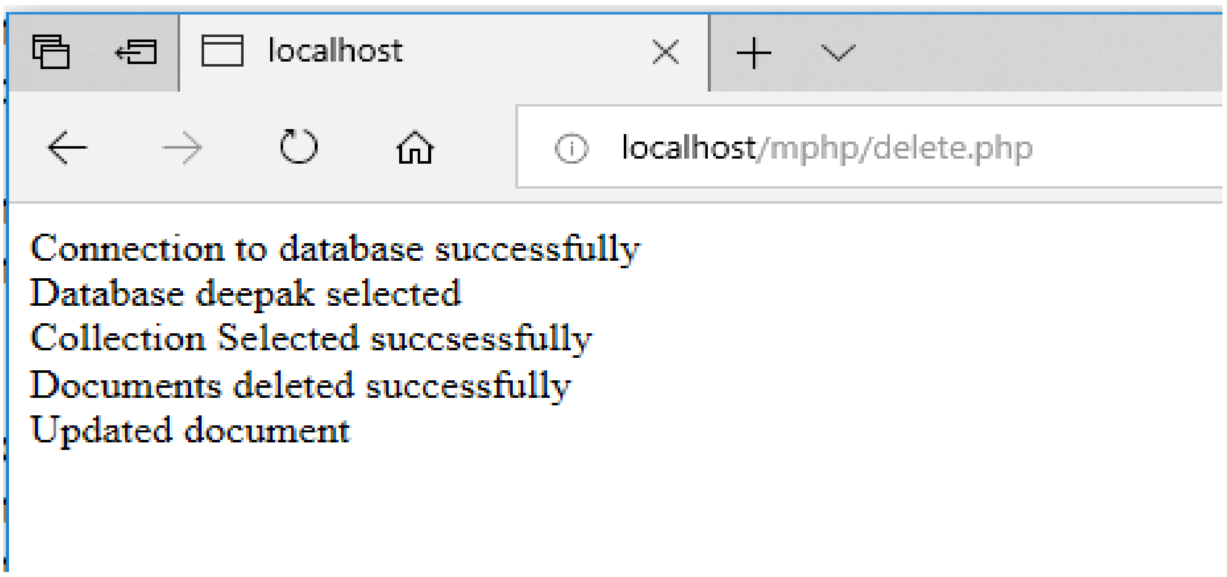


```
Connection to database successfullyDatabase mydb selectedCollection selected succsessfullyDocument updated successfullyUpdated documentMongoDB Tutorial sabina snehal MongoDB Tutorial
```

DELETE

Query:-

```
<?php
  // connect to mongodb
  $m = new MongoClient();
  echo "Connection to database successfully";

  // select a database
  $db = $m->mydb;
  echo "Database mydb selected";
  $collection = $db->mycol;
  echo "Collection selected succsessfully";

  // now remove the document
  $collection->remove(array("title"=>"MongoDB Tutorial"),false);
  echo "Documents deleted successfully";

  // now display the available documents
  $cursor = $collection->find();

  // iterate cursor to display title of documents
  echo "Updated document";

  foreach ($cursor as $document) {
    echo $document["title"] . "\n";
  }
?>
```



```
Connection to database successfully
Database deepak selected
Collection Selected succsessfully
Documents deleted successfully
Updated document
```

PRACTICAL NO :- 6

➢PYTHON AND MONGODB

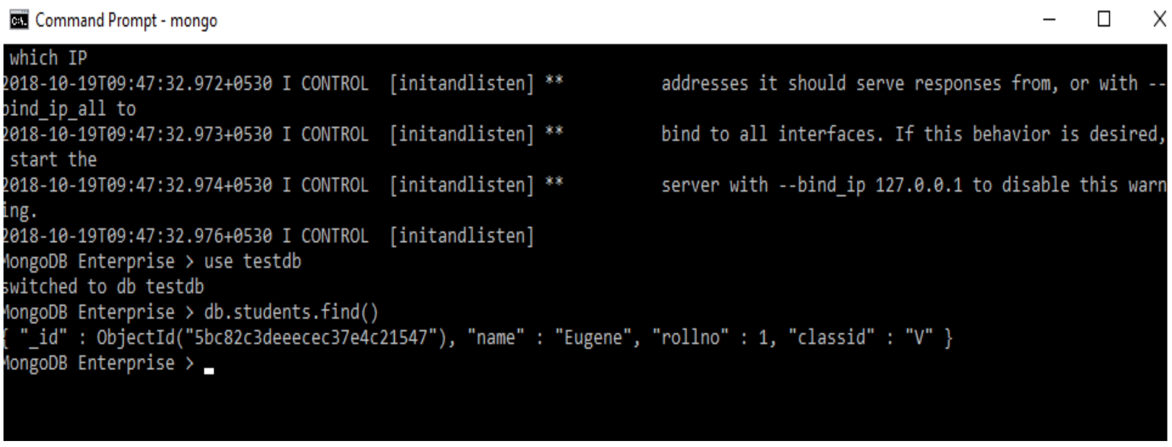
a) Connecting Python with MongoDB and inserting, retrieving, updating and deleting.

```
python -m pip install pymongo
```

INSERT

Query:-

```
import pymongo
from pymongo import MongoClient
client=MongoClient()
db=client.testdb
student1={“name”:”Eugene”,”classid”:’V’,”rollno”:1}
students=db.students
students_id=students.insert(student1)
```

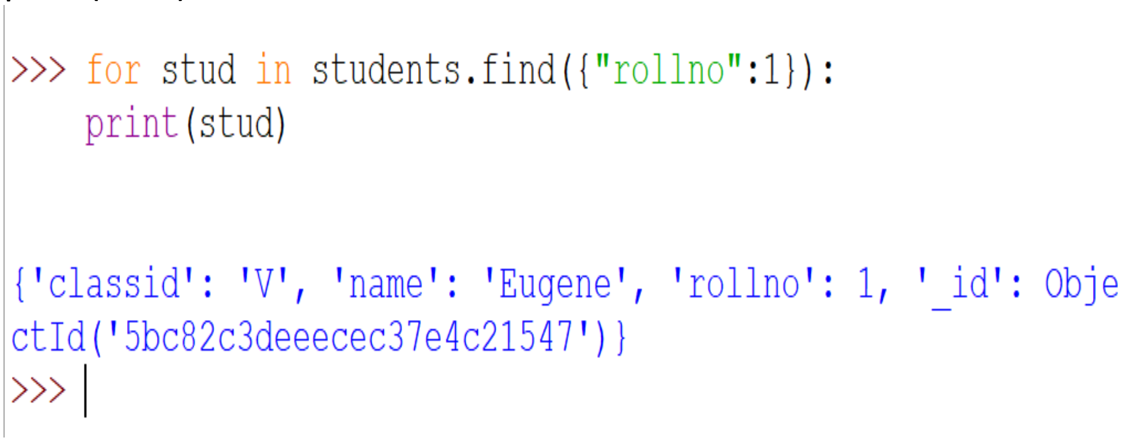


```
which IP
2018-10-19T09:47:32.972+0530 I CONTROL  [initandlisten] **        addresses it should serve responses from, or with --
bind_ip_all to
2018-10-19T09:47:32.973+0530 I CONTROL  [initandlisten] **        bind to all interfaces. If this behavior is desired,
 start the
2018-10-19T09:47:32.974+0530 I CONTROL  [initandlisten] **        server with --bind_ip 127.0.0.1 to disable this warn
ing.
2018-10-19T09:47:32.976+0530 I CONTROL  [initandlisten]
MongoDB Enterprise > use testdb
switched to db testdb
MongoDB Enterprise > db.students.find()
{ "_id" : ObjectId("5bc82c3deeecec37e4c21547"), "name" : "Eugene", "rollno" : 1, "classid" : "V" }
MongoDB Enterprise >
```

RETRIEVE

Query:-

```
import pymongo
from pymongo import MongoClient
client=MongoClient()
db=client.testdb
students=db.students
for stud in students.find({“rollno”:{“gte”:2}}):
print(stud)
```



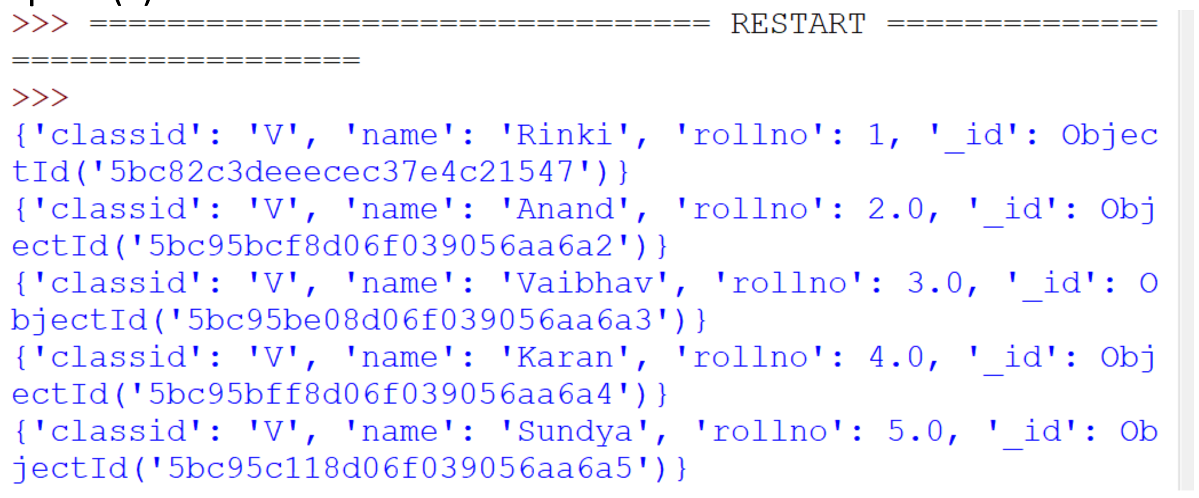
```
>>> for stud in students.find({"rollno":1}):
	print(stud)


{'classid': 'V', 'name': 'Eugene', 'rollno': 1, '_id': ObjectId('5bc82c3deeecec37e4c21547')}
>>> 
```

UPDATE

Query:-

```
import pymongo

myclient =
pymongo.MongoClient("mongodb://localhost:27017/")
mydb = myclient["mydatabase"]
mycol = mydb["customers"]

myquery = { "address": "Valley 345" }
newvalues = { "$set": { "address": "Canyon 123" } }

mycol.update_one(myquery, newvalues)
```

```
#print "customers" after the update:
for x in mycol.find():
  print(x)
```

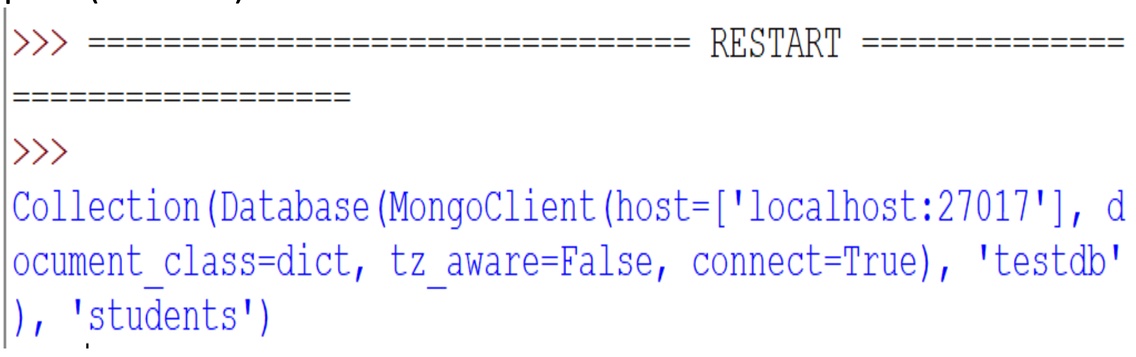


```
>>> ================================ RESTART ==============
==================
>>>
{'classid': 'V', 'name': 'Rinki', 'rollno': 1, '_id': Objec
tId('5bc82c3deeecec37e4c21547')}
{'classid': 'V', 'name': 'Anand', 'rollno': 2.0, '_id': Obj
ectId('5bc95bcf8d06f039056aa6a2')}
{'classid': 'V', 'name': 'Vaibhav', 'rollno': 3.0, '_id': O
bjectId('5bc95be08d06f039056aa6a3')}
{'classid': 'V', 'name': 'Karan', 'rollno': 4.0, '_id': Obj
ectId('5bc95bff8d06f039056aa6a4')}
{'classid': 'V', 'name': 'Sundya', 'rollno': 5.0, '_id': Ob
jectId('5bc95c118d06f039056aa6a5')}
```

DELETE

Query:-

```
import pymongo
from pymongo import MongoClient
client=MongoClient()
db=client.testdb
students=db.students
students.remove({‘rollno’:3})
print(student)
```



```
>>> ================================ RESTART ==============
==================
>>>
Collection(Database(MongoClient(host=['localhost:27017'], d
ocument_class=dict, tz_aware=False, connect=True), 'testdb'
), 'students')
```

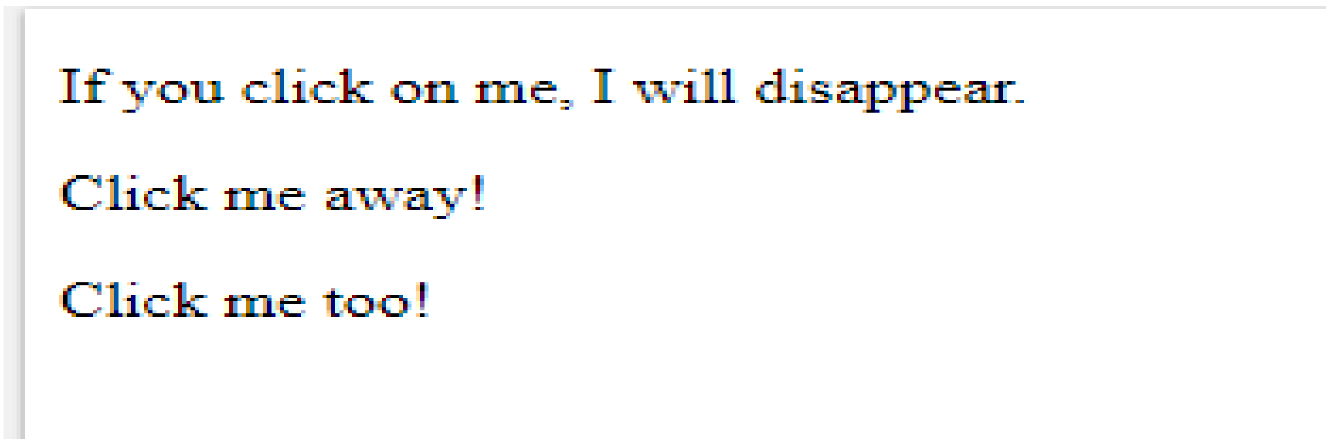
PRACTICAL NO :- 7

➢ PROGRAMS ON BASIC JQUERY

a) jQuery Basic, jQuery Events

click()

```
<!DOCTYPE html>
<html>
<head>
<script src="file:///C:/js/jquery-3.3.1.min.js"></script>
<script>
$(document).ready(function(){
   $("p").click(function(){
      $(this).hide();
   });
});
</script>
</head>
<body>

<p>If you click on me, I will disappear.</p>
<p>Click me away!</p>
<p>Click me too!</p>

</body>
</html>
```


If you click on me, I will disappear.

Click me away!

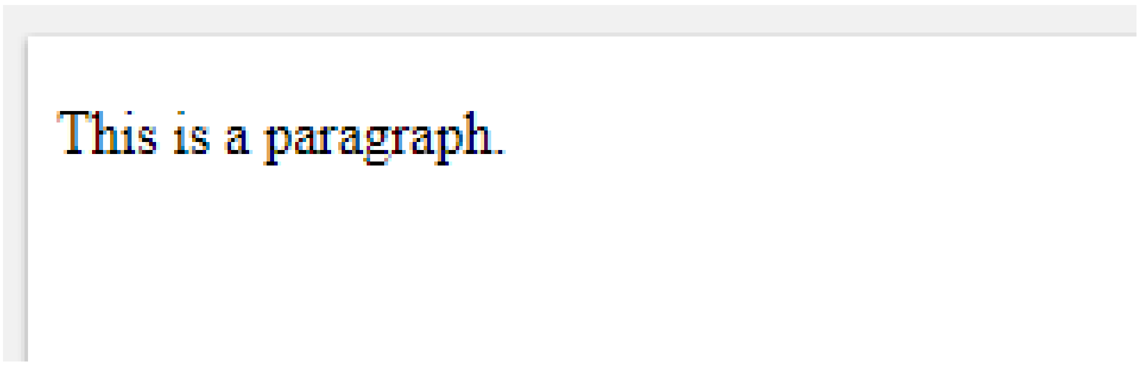
Click me too!

mousedown()

```
<!DOCTYPE html>
<html>
<head>
<script src=" file:///C:/js/jquery-3.3.1.min.js "></script>
<script>
$(document).ready(function(){
    $("#p1").mousedown(function(){
        alert("Mouse down over p1!");
    });
});
</script>
</head>
<body>

<p id="p1">This is a paragraph.</p>

</body>
</html>
```

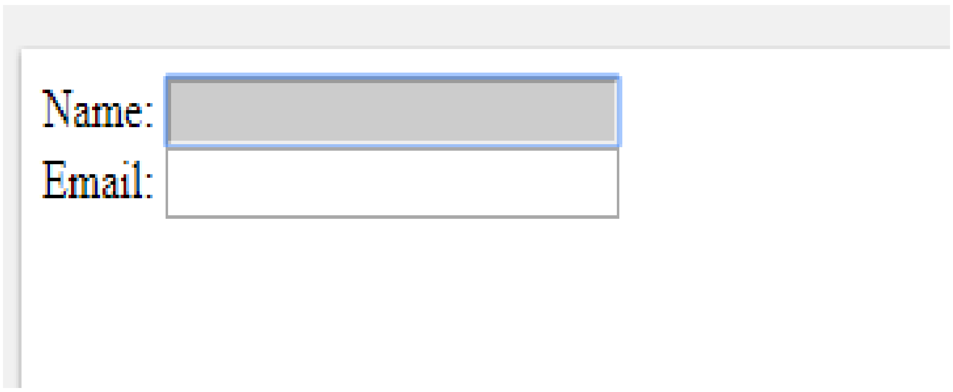
```
<!DOCTYPE html>
<html>
```

```
<head>
```


This is a paragraph.

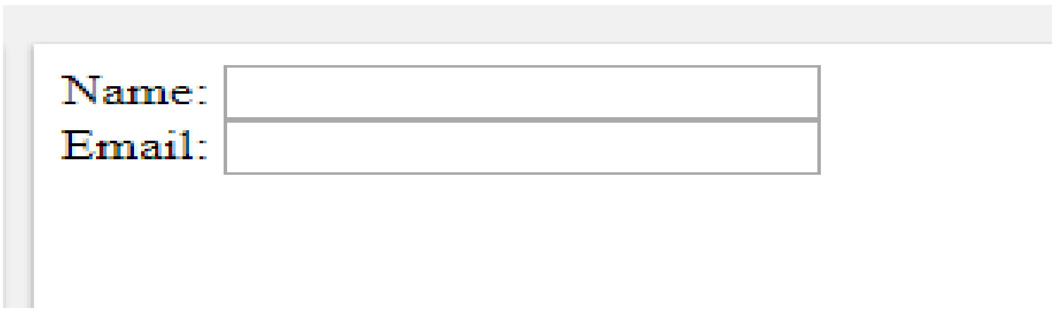
focus()

```
<script src=" file:///C:/js/jquery-3.3.1.min.js "></script>
<script>
$(document).ready(function(){
   $("input").focus(function(){
      $(this).css("background-color", "#cccccc");
   });
   $("input").blur(function(){
      $(this).css("background-color", "#ffffff");
   });
});
</script>
</head>
<body>

Name: <input type="text" name="fullname"><br>
Email: <input type="text" name="email">

</body>
</html>
```



blur()

```
<!DOCTYPE html>
<html>
<head>
<script src=" file:///C:/js/jquery-3.3.1.min.js "></script>
<script>
$(document).ready(function(){
  $("input").focus(function(){
    $(this).css("background-color", "#cccccc");
  });
  $("input").blur(function(){
    $(this).css("background-color", "#ffffff");
  });
});
</script>
</head>
<body>

Name: <input type="text" name="fullname"><br>
Email: <input type="text" name="email">

</body>
</html>
```

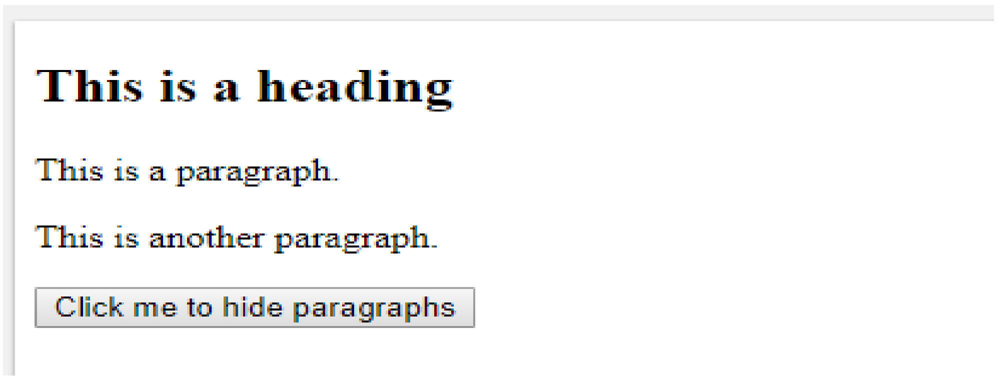


Name:
Email:

b) jQuery Selectors, jQuery Hide and Show effects

selectors()

```
<!DOCTYPE html>
<html>
<head>
<script src=" file:///C:/js/jquery-3.3.1.min.js "></script>
<script>
$(document).ready(function(){
   $("button").click(function(){
      $("p").hide();
   });
});
</script>
</head>
<body>

<h2>This is a heading</h2>

<p>This is a paragraph.</p>
<p>This is another paragraph.</p>

<button>Click me to hide paragraphs</button>

</body>
</html>
```


This is a heading

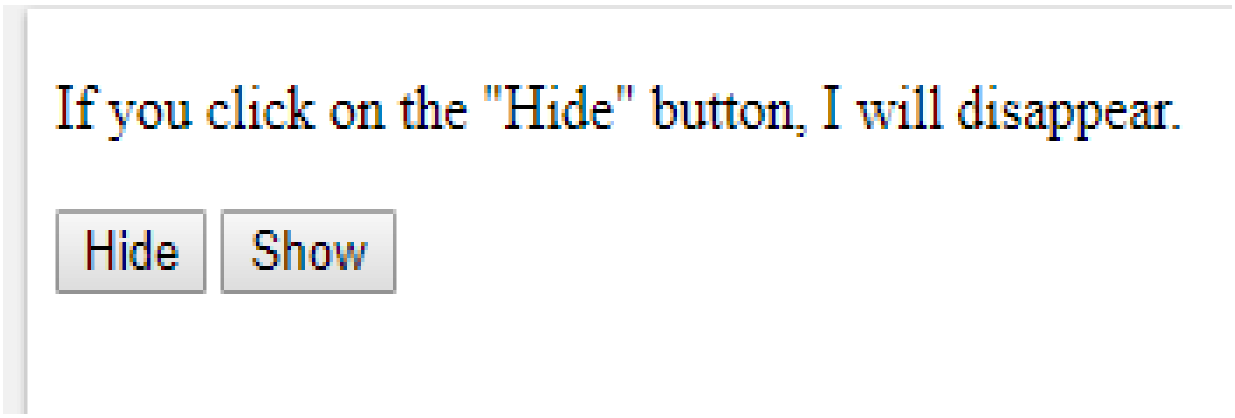
This is a paragraph.

This is another paragraph.

Hide() and show()

```
<!DOCTYPE html>
<html>
<head>
<script src=" file:///C:/js/jquery-3.3.1.min.js "></script>
<script>
$(document).ready(function(){
   $("#hide").click(function(){
      $("p").hide();
   });
   $("#show").click(function(){
      $("p").show();
   });
});
</script>
</head>
<body>

<p>If you click on the "Hide" button, I will disappear.</p>

<button id="hide">Hide</button>
<button id="show">Show</button>
```

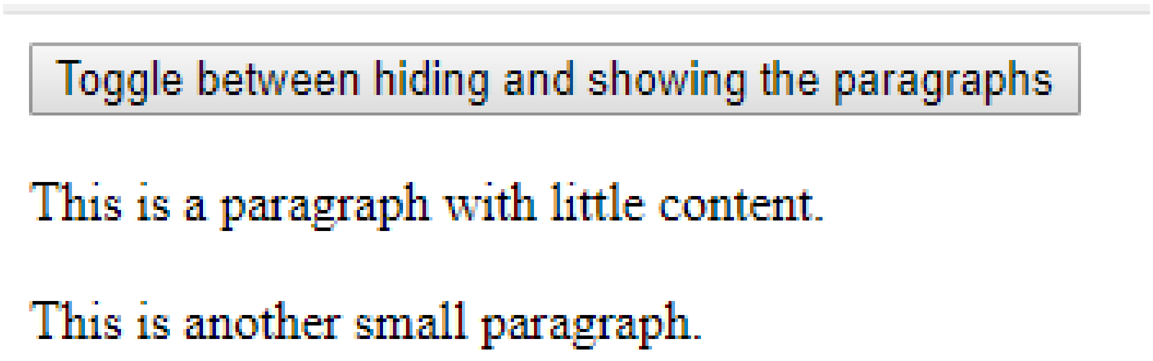
```
</body>
</html>
```


If you click on the "Hide" button, I will disappear.

toggle()

```
<!DOCTYPE html>
<html>
<head>
<script src=" file:///C:/js/jquery-3.3.1.min.js "></script>
<script>
$(document).ready(function(){
   $("button").click(function(){
      $("p").toggle();
   });
});
</script>
</head>
<body>

<button>Toggle between hiding and showing the
paragraphs</button>

<p>This is a paragraph with little content.</p>
<p>This is another small paragraph.</p>

</body>
```

```
</html>
```



This is a paragraph with little content.

This is another small paragraph.

c) jQuery fading effects, jQuery Sliding effects

fadeIn()

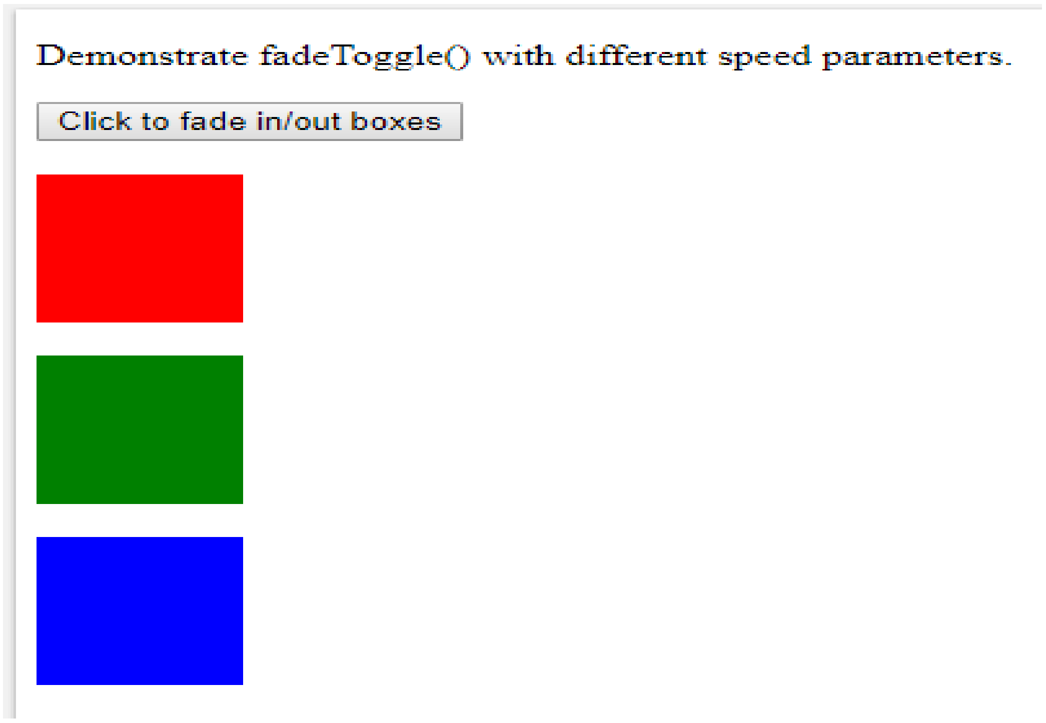
```
<!DOCTYPE html>
<html>
<head>
<script src=" file:///C:/js/jquery-3.3.1.min.js "></script>
<script>
$(document).ready(function(){
   $("button").click(function(){
      $("#div1").fadeIn();
      $("#div2").fadeIn("slow");
      $("#div3").fadeIn(3000);
   });
});
</script>
</head>
<body>

<p>Demonstrate fadeIn() with different parameters.</p>

<button>Click to fade in boxes</button><br><br>

<div id="div1"
style="width:80px;height:80px;display:none;background-
color:red;"></div><br>
<div id="div2"
style="width:80px;height:80px;display:none;background-
color:green;"></div><br>
<div id="div3"
style="width:80px;height:80px;display:none;background-
color:blue;"></div>

</body>
```

```
</html>
```

fadeToggle()

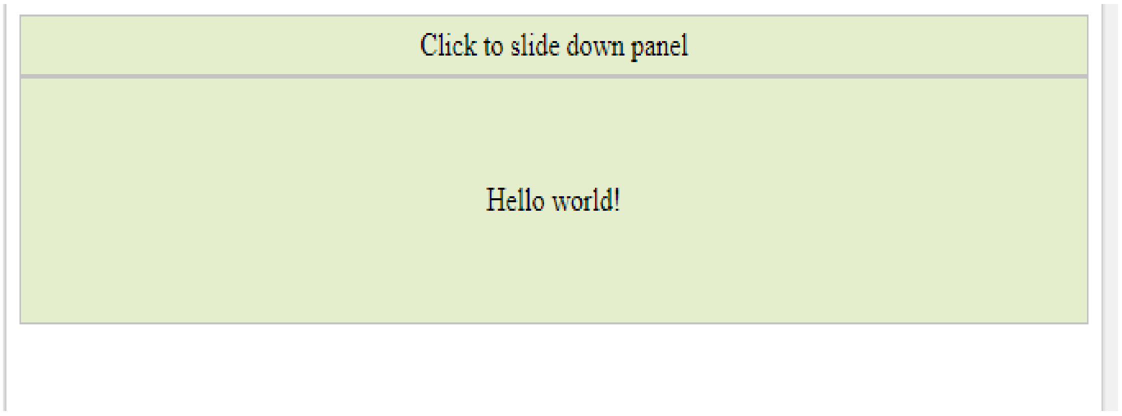
```
<!DOCTYPE html>
<html>
<head>
<script src=" file:///C:/js/jquery-3.3.1.min.js "></script>
<script>
$(document).ready(function(){
   $("button").click(function(){
      $("#div1").fadeToggle();
      $("#div2").fadeToggle("slow");
      $("#div3").fadeToggle(3000);
   });
});
</script>
</head>
<body>

<p>Demonstrate fadeToggle() with different speed
parameters.</p>

<button>Click to fade in/out boxes</button><br><br>

<div id="div1" style="width:80px;height:80px;background-
color:red;"></div>
<br>
<div id="div2" style="width:80px;height:80px;background-
color:green;"></div>
<br>
<div id="div3" style="width:80px;height:80px;background-
color:blue;"></div>

</body>
```

```
</html>
```


Demonstrate fadeToggle() with different speed parameters.

slideDown()

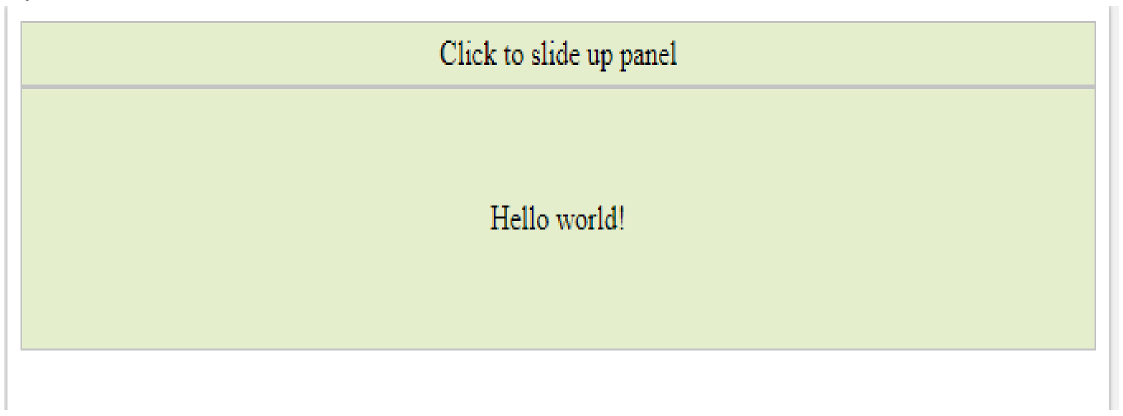
```
<!DOCTYPE html>
<html>
<head>
<script src=" file:///C:/js/jquery-3.3.1.min.js "></script>
<script>
$(document).ready(function(){
    $("#flip").click(function(){
        $("#panel").slideDown("slow");
    });
});
</script>

<style>
#panel, #flip {
    padding: 5px;
    text-align: center;
    background-color: #e5eecc;
    border: solid 1px #c3c3c3;
}

#panel {
    padding: 50px;
    display: none;
}
</style>
</head>
<body>

<div id="flip">Click to slide down panel</div>
<div id="panel">Hello world!</div>

</body>
</html>
```


Click to slide down panel

Hello world!

slideUp()

```
<!DOCTYPE html>
<html>
<head>
<script src=" file:///C:/js/jquery-3.3.1.min.js "></script>
<script>
$(document).ready(function(){
    $("#flip").click(function(){
        $("#panel").slideUp("slow");
    });
});
</script>

<style>
#panel, #flip {
    padding: 5px;
    text-align: center;
    background-color: #e5eecc;
    border: solid 1px #c3c3c3;
}

#panel {
    padding: 50px;
}
</style>
</head>
<body>

<div id="flip">Click to slide up panel</div>
<div id="panel">Hello world!</div>

</body>
</html>
```



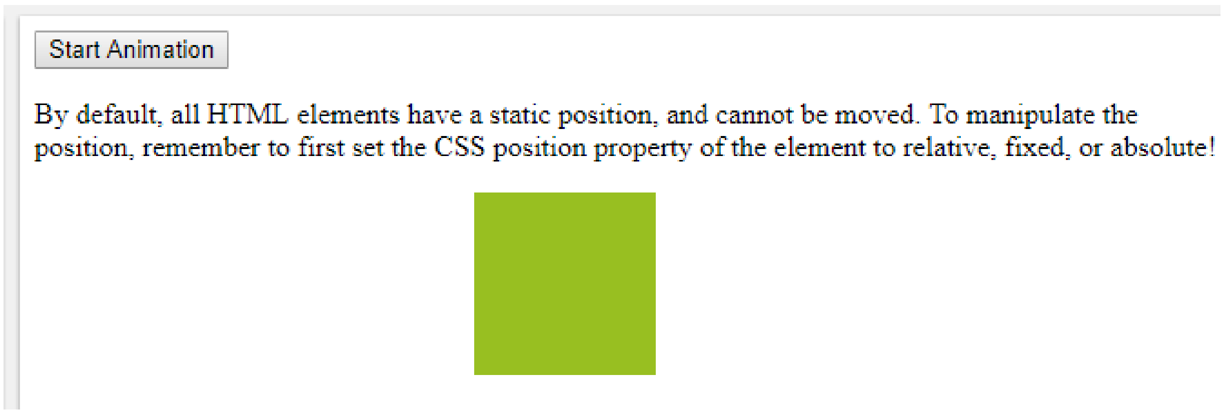
PRACTICAL NO :- 8

jQuery Advanced

a) jQuery Animation effects, jQuery Chaining

ANIMATION

```
<!DOCTYPE html>
<html>
<head>
<script src=" file:///C:/js/jquery-3.3.1.min.js "></script>
<script>
$(document).ready(function(){
   $("button").click(function(){
      $("div").animate({left: '250px'});
   });
});
</script>
</head>
<body>

<button>Start Animation</button>

<p>By default, all HTML elements have a static position, and cannot
be moved. To manipulate the position, remember to first set the CSS
position property of the element to relative, fixed, or absolute!</p>

<div
style="background:#98bf21;height:100px;width:100px;position:absol
ute;"></div>

</body>
</html>
```

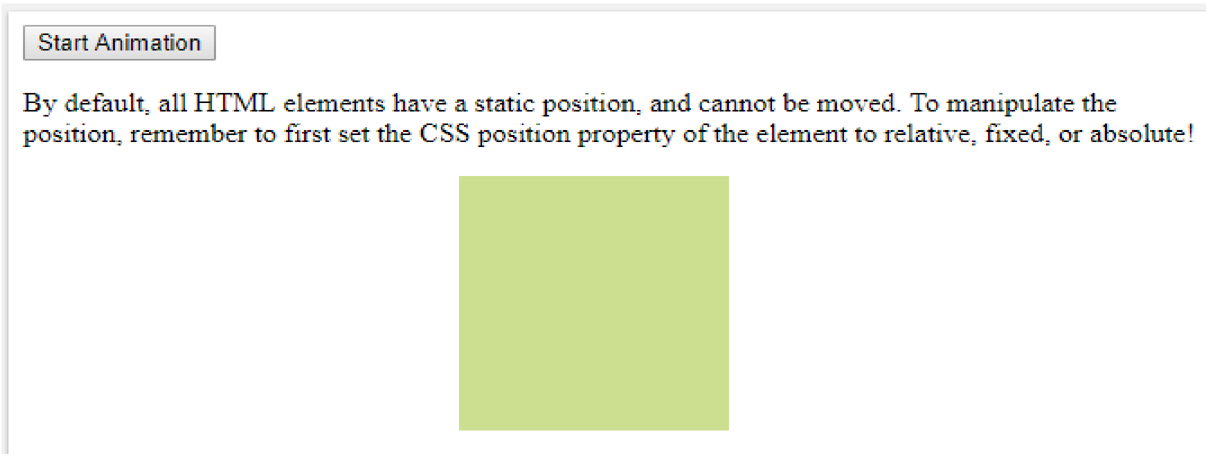


By default, all HTML elements have a static position, and cannot be moved. To manipulate the position, remember to first set the CSS position property of the element to relative, fixed, or absolute!

jQuery animate() - Manipulate Multiple Properties

```
<!DOCTYPE html>
<html>
<head>
<script src=" file:///C:/js/jquery-3.3.1.min.js "></script>
<script>
$(document).ready(function(){
   $("button").click(function(){
      $("div").animate({
         left: '250px',
         opacity: '0.5',
         height: '150px',
         width: '150px'
      });
   });
});
</script>
</head>
<body>

<button>Start Animation</button>
```

```
<p>By default, all HTML elements have a static position, and cannot
be moved. To manipulate the position, remember to first set the CSS
position property of the element to relative, fixed, or absolute!</p>

<div
style="background:#98bf21;height:100px;width:100px;position:absol
ute;"></div>

</body>
</html>
```

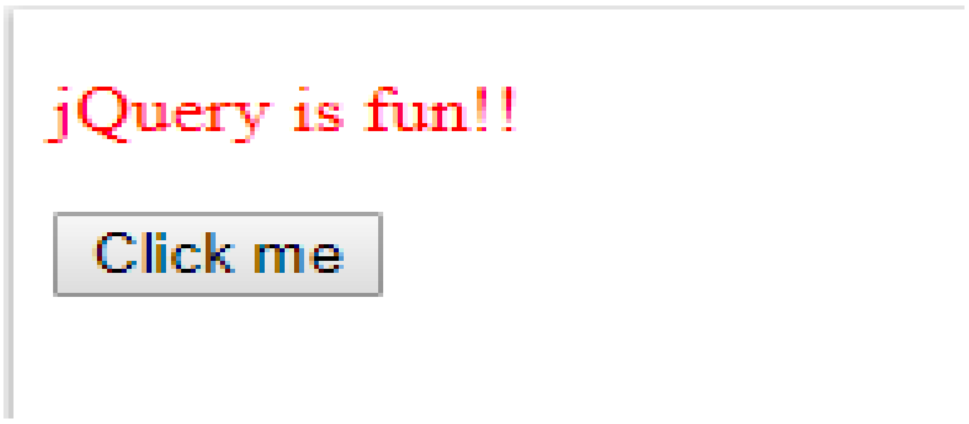


By default, all HTML elements have a static position, and cannot be moved. To manipulate the position, remember to first set the CSS position property of the element to relative, fixed, or absolute!

jQuery Method Chaining

```
<!DOCTYPE html>
<html>
<head>
<script src=" file:///C:/js/jquery-3.3.1.min.js "></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("#p1").css("color", "red").slideUp(2000).slideDown(2000);
  });
});
</script>
</head>
<body>
```

```
<p id="p1">jQuery is fun!!</p>

<button>Click me</button>

</body>
</html>
```

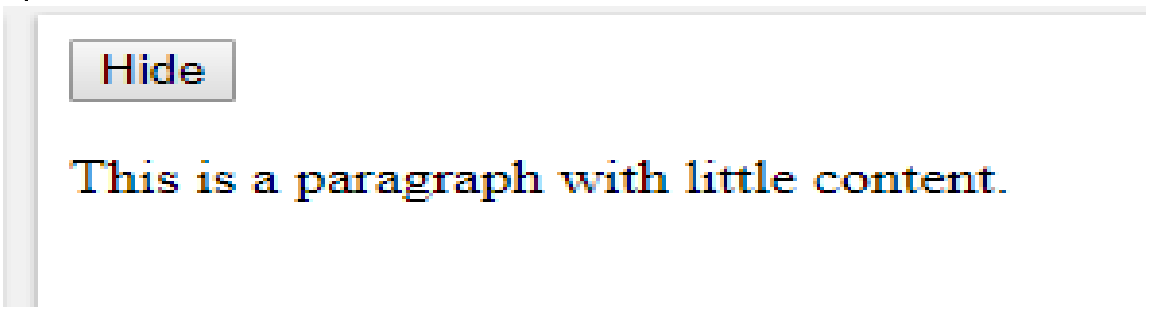


jQuery is fun!!

b) jQuery Callback, jQuery Get and Set Contents

jQuery Callback Functions

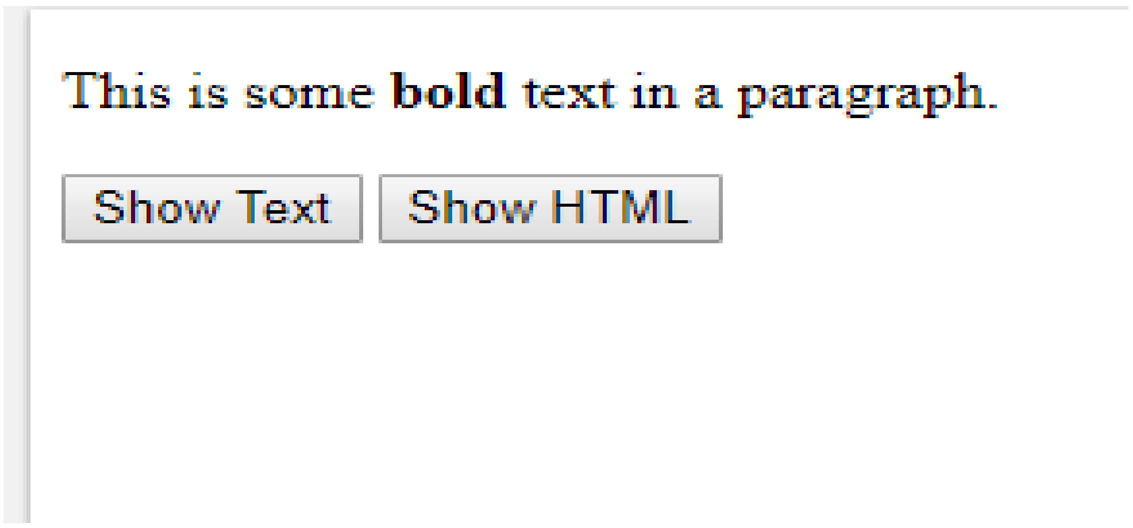
```
<!DOCTYPE html>
<html>
<head>
<script src=" file:///C:/js/jquery-3.3.1.min.js "></script>
<script>
$(document).ready(function(){
    $("button").click(function(){
        $("p").hide("slow", function(){
            alert("The paragraph is now hidden");
        });
    });
});
</script>
</head>
<body>

<button>Hide</button>

<p>This is a paragraph with little content.</p>

</body>
</html>
```



This is a paragraph with little content.

Get Content - text()

```
<!DOCTYPE html>
<html>
<head>
<script src=" file:///C:/js/jquery-3.3.1.min.js "></script>
<script>
$(document).ready(function(){
   $("#btn1").click(function(){
      alert("Text: " + $("#test").text());
   });
   $("#btn2").click(function(){
      alert("HTML: " + $("#test").html());
   });
});
</script>
</head>
<body>
<p id="test">This is some <b>bold</b> text in a paragraph.</p>
<button id="btn1">Show Text</button>
<button id="btn2">Show HTML</button>
</body>
</html>
```


This is some **bold** text in a paragraph.

PRACTICAL NO:-9

➢ JSON

a) creating JSON

- JSON stands for **J**ava**S**cript **O**bject **N**otation
- JSON is a lightweight data interchange format
- JSON is language independent *****
- JSON is "self-describing" and easy to understand

Example:-

```
   {
"employees":[
   {"firstName":"John", "lastName":"Doe"},
   {"firstName":"Anna", "lastName":"Smith"},
   {"firstName":"Peter", "lastName":"Jones"}
]
}
```

b)Parsing JSON

file:-

```
<!DOCTYPE html>
<html>
<body>
```

```
<h2>Use the XMLHttpRequest to get the content of a
file.</h2>
<p>The content is written in JSON format, and can easily
be converted into a JavaScript object.</p>

<p id="demo"></p>

<script>

var xmlhttp = new XMLHttpRequest();
xmlhttp.onreadystatechange = function() {
   if (this.readyState == 4 && this.status == 200) {
       var myObj = JSON.parse(this.responseText);
       document.getElementById("demo").innerHTML =
myObj.name;
   }
};
xmlhttp.open("GET", "json_demo.txt", true);
xmlhttp.send();

</script>

<p>Take a look at <a href="json_demo.txt"
target="_blank">json_demo.txt</a></p>

</body>
</html>
```

example:-

```
  var obj = JSON.parse(text);
```

PRACTICAL NO:-10

➢ Create a JSON file and import it to MongoDB

Import MongoDB to JSON:-

```
mongoimport --db dbName --collection collectionName --file
fileName.json --jsonArray
```

Export MongoDB to JSON:-

```
mongoexport --db sales --collection contacts --out contacts.js
on
```