GROUP C
ASSIGNMENT 4

NAME – Kartik Pingale
CLASS – TE-IT
BATCH - T2
ROLL NO – 7048

4. Implement Map reduce example with suitable example

MAP REDUCE

```
> var students = [
... {name : "Dale Cooper", class: "Calculus", tests: [30, 28,
45]},
... {name : "Harry Truman", class: "Geometry", tests: [28, 26,
44]},
... {name : "Shelly Johnson", class: "Calculus", tests: [27, 26,
43]},
... {name : "Bobby Briggs", class: "College Algebra, tests: [20,
18, 35]"},
... {name : "Donna Heyward", class: "Geometry", tests: [28, 28,
44]},
... {name : "Audrey Horne", class: "College Algebra", tests:
[22, 26, 44]},
... {name : "James Hurley", class: "Calculus", tests: [20, 20,
38]},
... {name : "Lucy Moran", class: "College Algebra", tests: [26,
24, 40]},
... {name : "Tommy Hill", class: "College Algebra", tests: [30,
29, 46]},
... {name : "Andy Brennan", class: "Geometry", tests: [20, 21,
38]}
... ];

> var studNames = [];

>  for( i=0; i < students.length; i++){
... studNames.push(students[i].name)
... }
10

> studNames
[
     "Dale Cooper",
```

```
     "Harry Truman",
     "Shelly Johnson",
     "Bobby Briggs",
     "Donna Heyward",
     "Audrey Horne",
     "James Hurley",
     "Lucy Moran",
     "Tommy Hill",
     "Andy Brennan"
]

> var studInfo = students.map(function(x) {
... return x.name + ' is in ' + x.class;
... });

> studInfo
[
     "Dale Cooper is in Calculus",
     "Harry Truman is in Geometry",
     "Shelly Johnson is in Calculus",
     "Bobby Briggs is in College Algebra, tests: [20, 18, 35]",
     "Donna Heyward is in Geometry",
     "Audrey Horne is in College Algebra",
     "James Hurley is in Calculus",
     "Lucy Moran is in College Algebra",
     "Tommy Hill is in College Algebra",
     "Andy Brennan is in Geometry"
]

> var tests = [
... {score: 30},
... {score: 28},
... {score: 45}
... ]

> var testSum = tests.reduce(function(sum, tests){
... return sum + tests.score;
... }, 0)

> testSum
103
```

PROBLEM STATEMENT

Implement Map Reduce operation with following example using MongoDB. Using Map Reduce function find the number of books

having pages less than 250 pages and greater than or equal to 250 pages

```
> show dbs
admin   0.000GB
config  0.000GB
emp_db  0.000GB
local   0.000GB

> use book_db
switched to db book_db

> db.createCollection("books")
{ "ok" : 1 }

> db.books.insert({Name: "Python", Pages: 100})
WriteResult({ "nInserted" : 1 })

> db.books.insert({Name: "Java", Pages: 500})
WriteResult({ "nInserted" : 1 })

> db.books.insert({Name: "PHP", Pages: 300})
WriteResult({ "nInserted" : 1 })

> db.books.insert({Name: "C++", Pages: 250})
WriteResult({ "nInserted" : 1 })

> db.books.insert({Name: "DBMS", Pages: 800})
WriteResult({ "nInserted" : 1 })

> var map = function() {
... emit(this.Pages >= 250 ? "More than 250" : "Less than 250",
1);
... }

> var reduce = function(key, values) {
... return Array.sum(values);
... }

> db.books.mapReduce(map, reduce, {out: "mymapreduce"})
{
     "result" : "mymapreduce",
     "timeMillis" : 383,
     "counts" : {
          "input" : 5,
```

```
          "emit" : 5,
          "reduce" : 1,
          "output" : 2
     },
     "ok" : 1
}

> db.mymapreduce.find()
{ "_id" : "Less than 250", "value" : 1 }
{ "_id" : "More than 250", "value" : 4 }

> db.books.insert({Name: "Machine Learning", Pages: 200})
WriteResult({ "nInserted" : 1 })

> db.mymapreduce.find()
{ "_id" : "Less than 250", "value" : 1 }
{ "_id" : "More than 250", "value" : 4 }

> db.books.mapReduce(map, reduce, {out: "mymapreduce"})
{
     "result" : "mymapreduce",
     "timeMillis" : 245,
     "counts" : {
          "input" : 6,
          "emit" : 6,
          "reduce" : 2,
          "output" : 2
     },
     "ok" : 1
}

> db.mymapreduce.find()
{ "_id" : "Less than 250", "value" : 2 }
{ "_id" : "More than 250", "value" : 4 }
```