Problem Statement 13 (Map Reduce)
Create collection for Student{roll_no, name, class, dept, aggregate_marks}. Write Map Reduce Functions for
following requirements.
1. Finding the total marks of students of “TE” class department-wise.
2. Finding the highest marks of students of “SE” class department-wise.
3. Find Average marks of students of “BE” class department-wise.

Create collection for Student{roll_no, name, class, dept, aggregate_marks}.
json

```
db.createCollection("Student");
{
    "roll_no": 1,
    "name": "Alice",
    "class": "TE",
    "dept": "Computer",
    "aggregate_marks": 75
},
{
    "roll_no": 2,
    "name": "Bob",
    "class": "TE",
    "dept": "Electrical",
    "aggregate_marks": 85
},
{
    "roll_no": 3,
    "name": "Charlie",
    "class": "SE",
    "dept": "Computer",
    "aggregate_marks": 90
},
{
    "roll_no": 4,
    "name": "David",
    "class": "SE",
    "dept": "Electrical",
    "aggregate_marks": 95
},
{
    "roll_no": 5,
    "name": "Eve",
    "class": "BE",
    "dept": "Computer",
    "aggregate_marks": 88
},
{
    "roll_no": 6,
    "name": "Frank",
    "class": "BE",
    "dept": "Electrical",
    "aggregate_marks": 92
}
```

Write Map Reduce Functions for following requirements.
1. Finding the total marks of students of “TE” class department-wise.

```
var mapTotalMarks = function() {
    if (this.class === "TE") {
        emit(this.dept, this.aggregate_marks);
    }
};

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


db.students.mapReduce(
    mapTotalMarks,
    reduceTotalMarks,
    { out: "total_marks_TE_department_wise" }
);
```

2. Finding the highest marks of students of “SE” class department-wise.

```
var mapHighestMarks = function() {
    if (this.class === "SE") {
        emit(this.dept, this.aggregate_marks);
    }
};


var reduceHighestMarks = function(key, values) {
    return Math.max.apply(null, values);
};


db.students.mapReduce(
    mapHighestMarks,
    reduceHighestMarks,
    { out: "highest_marks_SE_department_wise" }
);
```

3. Find Average marks of students of “BE” class department-wise.

```
var mapAverageMarks = function() {
    if (this.class === "BE") {
        emit(this.dept, { sum: this.aggregate_marks, count: 1 });
    }
};
var reduceAverageMarks = function(key, values) {
    var result = { sum: 0, count: 0 };
    values.forEach(function(value) {
        result.sum += value.sum;
        result.count += value.count;
    });
    return result;
};
var finalizeAverageMarks = function(key, reducedValue) {
    if (reducedValue.count > 0) {
        reducedValue.avg = reducedValue.sum / reducedValue.count;
```

```
    }
    return reducedValue.avg;
};
db.students.mapReduce(
    mapAverageMarks,
    reduceAverageMarks,
    {
        out: "average_marks_BE_department_wise",
        finalize: finalizeAverageMarks
    }
);
```