```
class LinkedList {

constructor(arr){
    if(arr){
      arr.forEach(elem => this.add(elem));
    }
}

  add(element) {
    if (this.value === undefined) {
      this.value = element;
      this.next = null;
    } else {
      let current = this;
      while (current.next) {
        current = current.next;
      }
      current.next = { value: element, next: null };
    }
  }

  remove(element) {
    var current = this;
    var prev = null;

    while (current) {
      if (current.value === element) {
        if (prev == null) {
          this.value = current.next.value;
          this.next = current.next.next;
        } else {
          prev.next = current.next;
        }
        return true;
      }
      prev = current;
      current = current.next;
    }
    return false;
  }

  printHelper(list, result) {

    if (list.next == null) {
      result += list.value;
```

```
      return result;
    }
    result += list.value + ',';
    return this.printHelper(list.next, result);
  }

  print() {
    let result = 'LinkedList{';
    result = this.printHelper(this, result);
    result += '}';
    console.log(result);
  }

}

let linkedlist = new LinkedList();
linkedlist.add(1);
linkedlist.add(2);
linkedlist.add(3);
linkedlist.print(); //in the console, you should see: LinkedList{1,2,3}
linkedlist.remove(3);
linkedlist.print(); //in the console, you should see: LinkedList{1,3}
```

This is a quiz system which allows students to take quiz, get each student's quiz score and compute average score of students.

You need to use constructor functions or class syntax to implement Student, Question and Quiz

constructor function/class Student:

properties:

studentId: a unique student id

answers: holds an array that records the student's answers for the questions.

method: addAnswer(question) - add student's question(id, answer) to answers array.

constructor function/class Question:

properties:

qid: unique quesiton id

answer: hold quiz correct answer or student's answer

method: checkAnswer(answer) - used to check if student's answer is correct

constructor function/class Quiz:

properties:

questions: It's a Map which holds question id and correct answers. The key is question id, the value is the correct answer for this question

students: an array holds all students

methods:

- scoreStudent(sid), computes the quiz score for this student
- getAverageScore(), computes the average score over all students

After you complete the Question, Student and Quiz constructor functions, we may use the system as below:

Your system should return the correct result for getAverageScore() and scoreStudent(sid) as the expected result.

```
const student1 = new Student(10);
student1.addAnswer(new Question(2, 'a'));
student1.addAnswer(new Question(3, 'b'));
student1.addAnswer(new Question(1, 'b'));
const student2 = new Student(11);
student2.addAnswer(new Question(3, 'b'));
student2.addAnswer(new Question(2, 'a'));
student2.addAnswer(new Question(1, 'd'));
const students = [student1, student2];
const questions =[new Question(1, 'b'), new Question(2, 'a'), new
Question(3, 'b')];
const quiz = new Quiz(questions, students);
let scoreforStudent10 = quiz.scoreStudentBySid(10);
console.log(scoreforStudent10); //Expected Result: 3
let scoreforStudent11 = quiz.scoreStudentBySid(11);
console.log(scoreforStudent11); //Expected Result: 2
let average = quiz.getAverageScore();
console.log(average); //Expected Reuslt: 2.5
```

Solution 1:

```
function Question(qid, answer) {
    this.qid = qid;
    this.answer = answer;
}

Question.prototype.checkAnswer = function(answer) {
    return this.answer === answer;
}

function Student(studentId) {
    this.studentId = studentId;
    this.answers = [];
}
```

```
Student.prototype.addAnswer = function(question) {
    this.answers.push(question);
}

function Quiz(questions, students) {
    this.questions = new Map();
    questions.forEach(q => this.questions.set(q.qid, q.answer));
    this.students = students;
}

Quiz.prototype.scoreStudentBySid = function(studentId) {
    const student = this.students.filter(s => s.studentId === studentId)[0];
    return student.answers.reduce((sum, currentQuestion) => {
        if (currentQuestion.checkAnswer(this.questions.get(currentQuestion.qid)))
 {
            sum = sum + 1;
        }
        return sum;
    }, 0);
}

Quiz.prototype.getAverageScore = function() {
    return this.students.reduce((accmulator, student, index, array) => {
        return accmulator + this.scoreStudentBySid(student.studentId) / array.len
gth;
    }, 0);
}


const questions = [new Question(1, 'b'), new Question(2, 'a'), new Question(3, 'b
')];

const student1 = new Student(10);
student1.addAnswer(new Question(2, 'a'));
student1.addAnswer(new Question(3, 'b'));
student1.addAnswer(new Question(1, 'b'));

const student2 = new Student(11);
student2.addAnswer(new Question(3, 'b'));
student2.addAnswer(new Question(2, 'a'));
student2.addAnswer(new Question(1, 'd'));

const students = [student1, student2];

const quiz = new Quiz(questions, students);
```

```
let scoreforStudent10 = quiz.scoreStudentBySid(10);
console.log(scoreforStudent10);

let scoreforStudent11 = quiz.scoreStudentBySid(11);
console.log(scoreforStudent11);

let average = quiz.getAverageScore();
console.log(average);
```

Solution 2:

```
class Question {
    constructor(questionId, answer) {
        this.questionId = questionId;
        this.answer = answer;
    }

    checkAnswer(correctAnswer) {
        return this.answer === correctAnswer;
    }

}

class Student {

    constructor(studentId, answers = []) {
        this.studentId = studentId;
        this.answers = answers;
    }

    addAnswer(question) {
        this.answers.push(question);
    }

}

class Quiz {
    constructor(questionsArray = [], students = []) {
        this.questions = new Map();
        //TODO: add line to convert questionArray to Map questions
```

```
        questionsArray.forEach(question => this.questions.set(question.questionId
, question.answer));
        this.students = students;
    }

    scoreStudent(studentId) {
        //TODO: compute student score based on answers
        let student = this.students.filter(student => student.studentId === stude
ntId)[0];
        return student.answers.reduce((sum, currentQuestion) => {
            let questionId = currentQuestion.questionId; //find quesiton id
            let correctAnswer = this.questions.get(questionId); //get correctAnsw
er from Map
            let result = currentQuestion.checkAnswer(correctAnswer); //compare cu
rrentQuestion answer with correctAnswer
            if (result) {
                sum = sum + 1;
            }

            // if(currentQuestion.checkAnswer(this.questions.get(currentQuestion.
questionId))){
            //   sum = sum + 1;
            // }


            return sum;
        }, 0);
    }

    getAverageScore() {
        return this.students.reduce((average, currentStudent, index, array) => av
erage + this.scoreStudent(currentStudent.studentId) / array.length, 0);
    }

}

const questionsArraywithCorrectAnswers = [new Question(1, 'a'), new Question(2, '
b'), new Question(3, 'd')];

let student1 = new Student(1001, [new Question(1, 'b'), new Question(2, 'b'), new
 Question(3, 'b')]);

let student2 = new Student(1002);
student2.addAnswer(new Question(1, 'a'));
student2.addAnswer(new Question(2, 'b'));
```

```
student2.addAnswer(new Question(3, 'd'));
const students = [student1, student2];

let quizObj = new Quiz(questionsArraywithCorrectAnswers, students);

console.log(quizObj.scoreStudent(1001));
console.log(quizObj.scoreStudent(1002));
console.log(quizObj.getAverageScore());
```