```
                    Solution to CSE143X Section #18 Problems


1.     Method Call        Value Returned
       -----------------------------------------------------------------------
       mystery1(1)        [[0]]
       mystery1(2)        [[0, 1, 2], [1, 2, 3]]
       mystery1(3)        [[0, 1, 2, 3, 4], [1, 2, 3, 4, 5], [2, 3, 4, 5, 6]]
       mystery1(4)        [[0, 1, 2, 3, 4, 5, 6], [1, 2, 3, 4, 5, 6, 7],
                          [2, 3, 4, 5, 6, 7, 8], [3, 4, 5, 6, 7, 8, 9]]


2.     Method Call               Output Produced
       ----------------------------------------------
       mystery2(grid, 0, 2, 2);    10 11
       mystery2(grid, 2, 3, 2);    12 9 12
       mystery2(grid, 1, 4, 1);     4 8 2 3


3.     Two-Dimensional Array               Contents of Set Returned
       ----------------------------------------------------------------
       [[1, 2], [3, 4]]                    [1, 2, 13, 14]
       [[7], [], [8, 8, 9, 10]]            [7, 28, 29, 30]
       [[3, 14], [5, 13, 4], [4, 3, 1]]    [3, 14, 15, 21, 23, 24]


4.     Method Call              Contents of Set Returned
       ----------------------------------------------
       mystery4(grid, 2, 2)     [6, 7]
       mystery4(grid, 0, 2)     [1, 2, 5, 8]
       mystery4(grid, 3, 3)     [1, 2, 3, 7, 9]


5. One possible solution appears below.
    public void recordGrade(Map<String, Map<String, Double>> grades,
                            String id, double grade, String course) {
        if (!grades.containsKey(id)) {
            grades.put(id, new TreeMap<>());
        }
        Map<String, Double> next = grades.get(id);
        if (next.containsKey(course)) {
            grade = Math.max(grade, next.get(course));
        }
        next.put(course, grade);
    }


6. One possible solution appears below.
    public Set<Point> removePoints(Map<Integer, List<Point>> points, int n) {
        Set<Point> removed = new HashSet<>();
        if (points.containsKey(n)) {
            Iterator<Point> itr = points.get(n).iterator();
            while (itr.hasNext()) {
                Point p = itr.next();
                if (p.getX() < p.getY()) {
                    itr.remove();
                    removed.add(p);
                }
            }
        }
        return removed;
    }


The complete Grid class and resources for the Sudoku program can be found at:

  https://courses.cs.washington.edu/courses/cse143x/23au/lectures/sudoku.zip
```