Solutions to Practice Midterm

Problem 1: Karel Prepares for the Olympics

```java
public class StrikerKarel extends SuperKarel {
    public void run() {
        // Karel stops upon reaching the goal, which is the only situation
        // when Karel can be blocked to the north, south, and east.
        while (leftIsClear() || rightIsClear() || frontIsClear()) {
            if (frontIsBlocked()) {
                findWallOpening();
            }
            dribble();
        }
    }

    /**
     * Pre: Karel is facing east in some avenue and their front is blocked.
     * Post: Karel is facing east in the same avenue and their front is clear.
     */
    private void findWallOpening() {
        turnLeft();
        // Go as far north as possible
        while (frontIsClear()) {
            dribble();
        }
        turnAround();
        // Go south until there is an opening in the wall
        while (leftIsBlocked()) {
            dribble();
        }
        turnLeft();
    }

    // Less elegant but marginally more efficient implementation of the above
    private void findWallOpeningFast() {
        // Look for the opening to the north.
        turnLeft();
        while (frontIsClear() && rightIsBlocked()) {
            dribble();
        }
        // Did we finish that loop because we found an opening to the north?
        if (rightIsClear()) {
            turnRight();
        } else {
            // If not, look for the opening that must be to the south.
            turnAround();
            while (leftIsBlocked()) {
                dribble();
            }
        }
        turnLeft();
    }
}
```

/**
 * Pre: Karel is on top of a beeper, and Karel's front is clear.
 * Post: Karel is on top of a beeper, having moved once in the direction
private void dribble() {
    pickBeeper();
    move();
    putBeeper();
}

Problem 2: You Prepare to be a Section Leader

(2a)

(double)(16 / 5) * 10  
'D' - 'A' == '3'
2 + 5 + "M" + 2 * 5 + 2
200 + 19 % 10 - (42 / 10.0 / 2) * 100
!(false || 3 != 4) && !(7 / 2.0 < 3.5)

(2b) What is printed out?

i. 9
ii. 18
iii. 92.0
iv. 66
v. NSP
Problem 3: Crossword

```java
public class MyCrosswordTrainer extends ConsoleProgram {
    private static final String FILENAME = "res/crossword.txt";

    // Part B
    public void run() {
        double revealChance = readDouble("Probability of revealing a letter? ");
        while (revealChance < 0 || revealChance > 1) {
            revealChance = readDouble("Enter a valid probability between 0 and 1: ");
        }
        try {
            Scanner input = new Scanner(new File(FILENAME));
            // input.nextInt() technically does not work but we'll allow it
            int numClues = Integer.parseInt(input.nextLine());
            int numCorrect = 0;
            for (int i = 0; i < numClues; i++) {
                String line = input.nextLine();
                int delimIndex = -1;
                for (int j = line.length() - 1; j >= 0; j--) {
                    if (line.charAt(j) == '#') {
                        delimIndex = j;
                        break;
                    }
                }
                String clue = line.substring(0, delimIndex);
                String answer = line.substring(delimIndex + 1);
                boolean success = testOneClue(clue, answer, revealChance);
                if (success) {
                    println("Correct!");
                    numCorrect++;
                } else {
                    println("The answer was: " + answer);
                }
            }
            println("Percent correct: " + 100.0 * numCorrect / numClues + ");
        } catch (FileNotFoundException e) {
            println(e);
        }
    }
}
```
private boolean testOneClue(String clue, String answer, double revealChance) {
    println(clue);
    String answerLetters = "";
    for (int i = 0; i < answer.length(); i++) {
        if (RandomGenerator.getInstance().nextBoolean(revealChance)) {
            answerLetters += answer.charAt(i);
        } else {
            answerLetters += "-";
        }
    }
    println(answerLetters);
    String guess = readLine("Guess: ");
    return guess.equalsIgnoreCase(answer);
}
}
Problem 4: Elections

```java
public class PollsterPanic extends GraphicsProgram {

    private static final int N_ROWS = 3;
    private static final int N_COLS = 5;
    private static final int ZONE_WIDTH = 120;
    private static final int ZONE_HEIGHT = 150;

    private GLabel label;

    // Part A
    public void run() {
        for (int row = 0; row < N_ROWS; row++) {
            for (int col = 0; col < N_COLS; col++) {
                double x = col * ZONE_WIDTH;
                double y = row * ZONE_HEIGHT;
                GRect zone = new GRect(x, y, ZONE_WIDTH, ZONE_HEIGHT);
                if (RandomGenerator.getInstance().nextBoolean()) {
                    zone.setColor(Color.BLUE);
                    zone.setFilled(true);
                    zone.setFillColor(Color.CYAN);
                    add(zone);
                } else {
                    zone.setColor(Color.RED);
                    zone.setFilled(true);
                    zone.setFillColor(Color.PINK);
                    add(zone);
                }
            }
        }
        label = new GLabel("Map created!");
        double mapEdgeX = ZONE_WIDTH * N_COLS;
        double openSpaceCenterX = mapEdgeX + (getWidth() - mapEdgeX) / 2;
        double labelX = openSpaceCenterX - (label.getWidth()) / 2;
        double labelY = label.getAscent();
        add(label, labelX, labelY);
    }

    // Part B
    public void mouseClicked(MouseEvent e) {
        int x = e.getX();
        int y = e.getY();
        GObject obj = getElementAt(x, y);
        String message = "";
        if (obj == null || obj == label) {
            message = "No zone selected";
        } else if (obj.getColor().equals(Color.BLUE)) {
            message = "Democrat zone";
        }
    }
}
```
} else if (obj.getColor().equals(Color.RED)) {
    message = "Republican zone";
}

label.setText(message);