Problem 1: Karel

```java
public class DayAtTheBeach extends SuperKarel {
    public void run() {
        collectAllShells();
        depositShells();
    }

    /* Pre: Karel is at (2, 1), facing east
    * Post: Karel is at upper left corner of the world, facing west */
    private void collectAllShells() {
        while(leftIsClear()) {
            collectRow();
            moveToNextRow();
        }
        collectRow();
    }

    /* Pre: Karel is in the leftmost spot of a row, facing east
    * Post: Karel has collected valid beeper piles in that row,
    * and is back in the leftmost spot of that row, facing west */
    private void collectRow() {
        while (frontIsClear()) {
            checkPile();
            move();
        }
        checkPile();
        turnAround();
        moveToEnd();
    }

    /* Pre: Karel is in the leftmost spot of a row, facing west
    * Post: Karel is one row up, facing east */
    private void moveToNextRow() {
        turnRight();
        move();
        turnRight();
    }

    /* Pre: Karel is standing on any corner
    * Post: Karel is standing in the same spot as before, but has
    * picked up the pile if it contained exactly 2 beepers */
    private void checkPile() {
        if (beepersPresent()) {
            pickBeeper();
            if (beepersPresent()) {
                pickBeeper();
            } else {
                putBeeper();
            }
        }
    }
}
```
private void moveToEnd() {
    while (frontIsClear()) {
        move();
    }
}

private void depositShells() {
    // move to bucket
    turnLeft(); // Karel is now facing south
    moveToEnd(); // this moves Karel down to (1, 1)

    // place all collected beepers into bucket
    while (beepersInBag()) {
        putBeeper();
    }
}
Problem 2: Expressions & Tracing

(2a)

\[ 5 + 7 \times 3 / 2 + (1 - 1) \times 2 \]
\[ \text{________15___________} \]

\[ 'a' + "pp" + (double)1 + 'e' \]
\[ \text{______)"appl.0e"___________} \]

\[ (!true || 9 \% 2 > 0) && (44 / 10 == 4.4) \]
\[ \text{______)false___________} \]

(2b)  What are the values of each variable?

```
a: 'b'
num: 15
circle: width 100, height 100, BLUE
dukesPwd: "iloveCS106A"
```
Problem 3: Console Program

```java
public class GrandOpening extends ConsoleProgram {
    public void run() {
        println("Welcome to the Grand Opening of the new Stanford Dog Park! ");

        int totNumTreats = 0;

        String name = readLine("What is your dog's name? ");
        while (!name.isEmpty()) {
            int size = readInt("How big is your dog?
                Enter a number: 1 - small, 2 - medium, 3 - large ");

            // calculate num treats
            int numTreats = size * 2;
            totNumTreats += numTreats;
            println("Oh boy! " + name + " gets " + numTreats + " dog treats!");

            // calculate if dog gets squeaky toy
            boolean getsToy = RandomGenerator.getInstance().nextBoolean(0.25);
            if (getsToy) {
                println(name + " also won a squeaky toy!");
            }

            // prompt for new name
            name = readLine("What is your dog's name? ");
        }

        // at the end of the program, print total number of treats
        println("At the grand opening, there will be ");
        println(" + totNumTreats + " dog treats!");
    }
}
```
public class MatchingGame extends GraphicsProgram {
    private static final double NUM_ROWS = 5;
    private static final double NUM_COLS = 5;

    // tracks the first of two cards that a user clicks
    private GRect card1;

    public void run() {
        double cardWidth = getWidth() / NUM_COLS;
        double cardHeight = getHeight() / NUM_ROWS;

        for (int r = 0; r < NUM_ROWS; r++) {
            for (int c = 0; c < NUM_COLS; c++) {
                GRect square = new GRect(c * cardWidth, r * cardHeight, cardWidth, cardHeight);

                Color fillColor = getRandomColor();
                Color borderColor = getRandomColor();
                square.setFilled(true);
                square.setFillColor(fillColor);
                square.setColor(borderColor);
                add(square);
            }
        }
    }

    public void mouseClicked(MouseEvent e) {
        double x = e.getX();
        double y = e.getY();
        GRect elem = getElementAt(x, y);

        if (elem != null) {
            if (card1 == null) {
                // if card1 is null, this is the first of two cards the user
                // has clicked: store that card in our instance variable
                card1 = elem;
            } else if (elem != card1) {
                // if we get here, the user has clicked a second card
                if (elem.getColor() == card1.getColor() && elem.getFillColor() == card1.getFillColor()) {
                    remove(card1); // remove first card
                    remove(elem); // remove second card
                }

                // clear user selection: reset card1 to null
                card1 = null;
            }
        }
    }
}
Problem 5: Strings

```java
private String frontCoding(String str1, String str2) {
    // get the index of the first character where the two strings differ
    int endIdx = 0;
    for (int i = 0; i < str1.length(); i++) {
        if (str1.charAt(i) != str2.charAt(i)) {
            break;
        } else {
            endIdx++;
        }
    }

    // the two strings have a common prefix up until (but excluding) endIdx
    String prefix = str1.substring(0, endIdx);

    // extract suffixes
    String suffix1 = str1.substring(endIdx);
    String suffix2 = str2.substring(endIdx);

    return prefix.length() + prefix + "*" + suffix1.length() + suffix1 + suffix2.length() + suffix2;
}
```