Question 1

(a) Part

```
public int findFreeBlock (int period, int duration)  {
    int blockLength = 0;

    for (int minute = 0; minute < 60; minute++) {
        if(isMinuteFree(period, minute)) {
            blockLength++;
            if (blockLength == duration){
                return minute - blockLength + 1;
            }
        } else {
            blockLength = 0;
        }
    }

    return -1;
}
```

(b) Part

```
public boolean makeAppointment(int startPeriod, int endPeriod, int duration) {
    // reserveBlock method: reserveBlock(period, startMinute, duration)
    for (int period = startPeriod; period <= endPeriod; period++) {
        int freeBlockMinute = findFreeBlock(period, duration);

        if (freeBlockMinute != -1) {
            reserveBlock(period, freeBlockMinute, duration);
            return true;
        }
    }

    return false;
}
```

Question 2

```
public class Sign {
    private String signName;
    private int widthOfSign;

    public Sign (String n, int w) {
        signName = n;
        widthOfSign = w;
    }

    public int numberOfLines() {
        int signNameLength = signName.length();

        if (signNameLength % widthOfSign != 0) {
            return signNameLength / widthOfSign + 1;
        } else {
            return signNameLength / widthOfSign;
        }
    }
```

```
    public String getLines() {
        String signNameModulated = new String();
        int signNameLength = signName.length();

        int counter = 0;

        for(int i = 0; i < signNameLength; i++) {
            signNameModulated = signNameModulated + signName.substring(i, i+1);
            counter++;
            if (counter % widthOfSign == 0) {
                signNameModulated = signNameModulated + ";";
                counter = 0;
            }
        }

        return signNameModulated;
    }
}
```

Question 3

(a) Part

```
public void cleanData (double lower, double upper) {
    for (int index = 0; index < temperatures.size(); index++) {
        if (temperatures.get(index) < lower || temperatures.get(index) > upper) {
            temperatures.remove(index);
        }
    }
}
```

(b) Part

```
public int longestHeatWave(int threshold) {
    int counter = 0;
    int longestCounter = 0;
    for (int index = 0; index < temperatures.size(); index++){
        if (temperatures.get(index) > threshold) {
            counter++;
        } else {
            if (counter > longestCounter) {
                longestCounter = counter;
                counter = 0;
            } else {
                counter = 0;
            }
        }
    }

    return longestCounter;
}
```

Question 4

(a) Part

```
public boolean moveCandyToFirstRow(int col) {
    if (box[0][col] != null) {
```

```
        return true;
    }

    for (int row = 1; row < box.length; row++) {
        if (box[row][col] != null) {
            box[0][col] = box[row][col];
            box[row][col] = null;
            return true;
        }
    }

    return false;
}
```

(b) Part

```
public Candy removeNextByFlavor(String flavor) {
    for (int row = box.length - 1; row >= 0; row--) {
        for (int col = 0; col < box[row].length; col++) {
            if (box[row][col] != null) {
                if (box[row][col].getFlavor().equals(flavor)) {
                    Candy removedCandy = box[row][col];
                    box[row][col] = null;
                    return removedCandy;
                }
            }
        }
    }

    return null;
}
```