Ones.java

```
import java.util.Scanner;
public class Ones {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        while (sc.hasNextInt()) {
            int n = sc.nextInt();
            int num = 1, length = 1;
            while (num % n != 0) {
                num = (num * 10 + 1) % n;
                length++;
            }
            System.out.println(length);
        }
    }
}
```

PrimaryArithmetic.java

```
import java.util.Scanner;
public class PrimaryArithmetic {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        while (true) {
            int a = sc.nextInt(), b = sc.nextInt();
            if (a == 0 && b == 0) break;
            int carry = 0, count = 0;
            while (a > 0 || b > 0) {
                if ((a % 10 + b % 10 + carry) >= 10) {
                    carry = 1;
                    count++;
                } else {
                    carry = 0;
                }
                a /= 10;
                b /= 10;
            }
            if (count == 0) System.out.println("No carry operation.");
            else if (count == 1) System.out.println("1 carry operation.");
            else System.out.println(count + " carry operations.");
        }
    }
}
```

LongestNap.java

```
import java.util.*;
public class LongestNap {
    static int toMin(String t) {
        String[] parts = t.split(":");
        return Integer.parseInt(parts[0]) * 60 + Integer.parseInt(parts[1]);
    }
    static String toTime(int min) {
        return String.format("%02d:%02d", min / 60, min % 60);
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int day = 1;
        while (sc.hasNext()) {
            int n = sc.nextInt();
            sc.nextLine();
            List<int[]> intervals = new ArrayList<>();
            for (int i = 0; i < n; i++) {
                String[] parts = sc.nextLine().split(" ", 3);
                int start = toMin(parts[0]);
                int end = toMin(parts[1]);
                intervals.add(new int[]{start, end});
            }
            intervals.add(new int[]{600, 600}); // 10:00
            intervals.add(new int[]{1080, 1080}); // 18:00
            intervals.sort(Comparator.comparingInt(a -> a[0]));
            int maxNap = 0, napStart = 600;
            for (int i = 1; i < intervals.size(); i++) {
                int nap = intervals.get(i)[0] - intervals.get(i - 1)[1];
                if (nap > maxNap) {
                    maxNap = nap;
                    napStart = intervals.get(i - 1)[1];
                }
            }
              System.out.printf("Day #%d: the longest nap starts at %s and will last for
%d minutes.\n",
                    day++, toTime(napStart), maxNap);
        }
    }
}
```

VitosFamily.java

```
import java.util.Arrays;
import java.util.Scanner;
public class VitosFamily {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int t = sc.nextInt();
        while (t-- > 0) {
            int n = sc.nextInt();
            int[] arr = new int[n];
            for (int i = 0; i < n; i++) arr[i] = sc.nextInt();
            Arrays.sort(arr);
            int median = arr[n / 2];
            int sum = 0;
            for (int x : arr) sum += Math.abs(x - median);
            System.out.println(sum);
        }
    }
}
```

WheresWaldorf.java

```
import java.util.*;
public class WheresWaldorf {
    static int[] dx = {-1,-1,-1,0,0,1,1,1};
    static int[] dy = {-1,0,1,-1,1,-1,0,1};
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int T = sc.nextInt();
        while (T-- > 0) {
            int m = sc.nextInt(), n = sc.nextInt();
            sc.nextLine();
            char[][] grid = new char[m][n];
                                for (int i = 0; i < m; i++) grid[i] =
sc.nextLine().toLowerCase().toCharArray();
            int w = sc.nextInt();
            sc.nextLine();
            for (int k = 0; k < w; k++) {
                String word = sc.nextLine().toLowerCase();
                boolean found = false;
                for (int i = 0; i < m && !found; i++) {
                    for (int j = 0; j < n && !found; j++) {
                        for (int d = 0; d < 8 && !found; d++) {
                            int x = i, y = j, l = 0;
                             while (l < word.length() && x >= 0 && y >= 0 && x < m && y <
n && grid[x][y] == word.charAt(l)) {
                                x += dx[d];
                                y += dy[d];
                                l++;
                            }
                            if (l == word.length()) {
                                System.out.println((i + 1) + " " + (j + 1));
                                found = true;
                            }
                        }
                    }
                }
            }
            if (T > 0) System.out.println();
        }
    }
}
```

WERTYU.java

```
import java.util.Scanner;
public class WERTYU {
    public static void main(String[] args) {
        String keys = "`1234567890-=QWERTYUIOP[]\\ASDFGHJKL;'ZXCVBNM,./";
        Scanner sc = new Scanner(System.in);
        while (sc.hasNextLine()) {
            String line = sc.nextLine();
            StringBuilder result = new StringBuilder();
            for (char c : line.toCharArray()) {
                int index = keys.indexOf(c);
                result.append(index > 0 ? keys.charAt(index - 1) : c);
            }
            System.out.println(result);
        }
    }
}
```

TheTrip.java

```
import java.util.Scanner;
public class TheTrip {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        while (true) {
            int n = sc.nextInt();
            if (n == 0) break;
            double[] expenses = new double[n];
            double total = 0;
            for (int i = 0; i < n; i++) {
                expenses[i] = sc.nextDouble();
                total += expenses[i];
            }
            double avg = total / n, give = 0;
            for (double exp : expenses) {
                if (exp > avg) give += exp - avg;
            }
            System.out.printf("$%.2f\n", give);
        }
    }
}
```

JollyJumpers.java

```
import java.util.Scanner;
import java.util.HashSet;
public class JollyJumpers {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        while (sc.hasNext()) {
            int n = sc.nextInt();
            int[] arr = new int[n];
            for (int i = 0; i < n; i++) arr[i] = sc.nextInt();
            HashSet<Integer> diffs = new HashSet<>();
            for (int i = 1; i < n; i++) {
                diffs.add(Math.abs(arr[i] - arr[i - 1]));
            }
            boolean jolly = true;
            for (int i = 1; i < n; i++) {
                if (!diffs.contains(i)) {
                    jolly = false;
                    break;
                }
            }
            System.out.println(jolly ? "Jolly" : "Not jolly");
        }
    }
}
```

Hartals.java

```
import java.util.Scanner;
public class Hartals {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int T = sc.nextInt();
        while (T-- > 0) {
            int N = sc.nextInt();
            int P = sc.nextInt();
            boolean[] days = new boolean[N + 1];
            for (int i = 0; i < P; i++) {
                int h = sc.nextInt();
                for (int j = h; j <= N; j += h) {
                    if (j % 7 != 6 && j % 7 != 0) days[j] = true;
                }
            }
            int count = 0;
            for (int i = 1; i <= N; i++) if (days[i]) count++;
            System.out.println(count);
        }
    }
}
```

ThreeNPlusOne.java

```
import java.util.Scanner;
public class ThreeNPlusOne {
    static int cycleLength(int n) {
        int count = 1;
        while (n != 1) {
            if (n % 2 == 0)
                n /= 2;
            else
                n = 3 * n + 1;
            count++;
        }
        return count;
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        while (sc.hasNext()) {
            int i = sc.nextInt(), j = sc.nextInt();
            int maxCycle = 0;
            for (int k = Math.min(i, j); k <= Math.max(i, j); k++) {
                maxCycle = Math.max(maxCycle, cycleLength(k));
            }
            System.out.println(i + " " + j + " " + maxCycle);
        }
    }
}
```