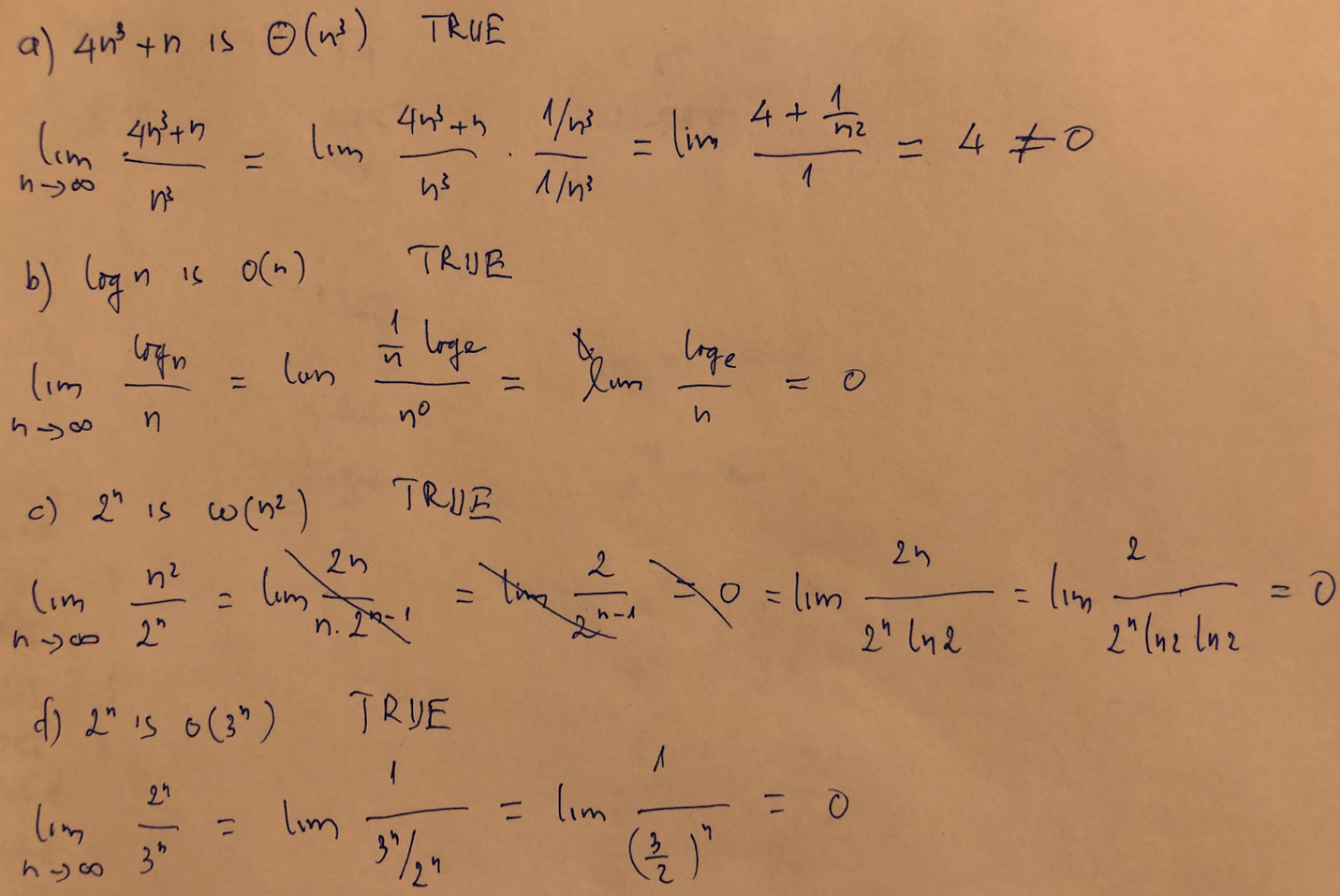
# Problem1:

1. f(x) = -x2 is eventually nondecreasing
2. f(x) = x2 + 2x + 1 is eventually nondecreasing
3. f(x) = x3 + x is increasing

# Problem2:



# Problem3:

n > 4, ϕ(n): 2n < n!

**Basic step:**

with n=5: 25 < 1\*2\*3\*3\*5 is true

**Induction step:**

Assume ϕ(n) is true

2n+1 = 2\*2n < 2 \* n!

< (n+1) \* n! (with n>1)

= (n+1)!

So we have 2n+1<(n+1)! => ϕ(n+1) is true

# Problem4:

package lab01;

public class prob4 {

public static void main(String[] agrs) {

// System.out.println("skdjflsjdf");

System.out.println(gcd(12, 42));

System.out.println(gcd(7, 9));

}

static int gcd(int m, int n) {

int min = Math.min(m, n);

for (int i = min; i >= 1; i--) {

if (m % i == 0 && n % i == 0)

return i;

}

return 1;

}

}

# Problem5:

public static int secondSmallest(int[] a) {

if (a == null || a.length < 2) {

throw new IllegalArgumentException("Input array too small");

}

// implement

int min = Integer.MAX\_VALUE;

int min2 = min;

for (int i = 0; i < a.length; i++) {

if (a[i] <= min) {

min2 = min;

min = a[i];

} else {

if (a[i] < min2)

min2 = a[i];

}

}

return min2;

}

# Problem6:

public static List<Integer> subset(int[] A, int sum) {

int n = A.length;

// T[i][j] stores true if subset with sum j can be attained

// with using items up to first i items

boolean[][] T = new boolean[n + 1][sum + 1];

// if sum is zero

for (int i = 0; i <= n; i++) {

T[i][0] = true;

}

// do for ith item

for (int i = 1; i <= n; i++) {

// consider all sum from 1 to sum

for (int j = 1; j <= sum; j++) {

// don't include ith element if j-A[i-1] is negative

if (A[i - 1] > j) {

T[i][j] = T[i - 1][j];

} else {

// find subset with sum j by excluding or including

// the ith item

T[i][j] = T[i - 1][j] || T[i - 1][j - A[i - 1]];

}

}

}

//back track

if (T[n][sum]) {

List<Integer> l = new ArrayList<>();

int i = n, j = sum;

boolean movedLeft = false;

while (i > 0 && j > 0) {

if (T[i - 1][j]) {

i--;

if (movedLeft) {

movedLeft = false;

l.add(A[i]);

}

continue;

}

j--;

movedLeft = true;

}

return l;

}

return null;

}