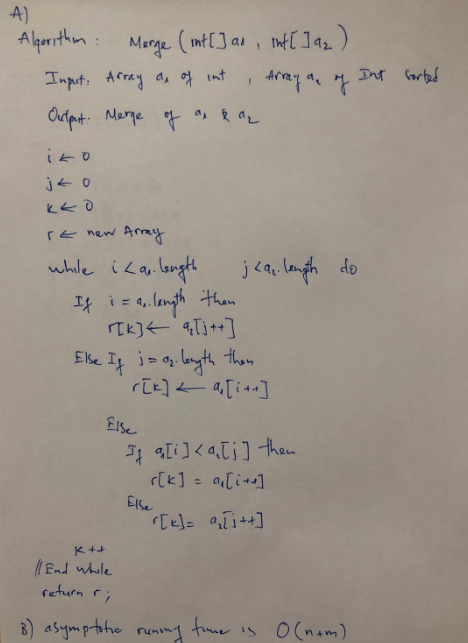
# Prob1:

Asymptotic running time is O(n2)

# Prob2:



C)

static int[] merge(int[] a1, int[] a2) {

int[] r = new int[a1.length + a2.length];

int i = 0, j = 0, k = 0;

while (i < a1.length || j < a2.length) {

if (i == a1.length) {

r[k] = a2[j++];

} else if (j == a2.length) {

r[k] = a1[i++];

} else {

if (a1[i] < a2[j])

r[k] = a1[i++];

else

r[k] = a2[j++];

}

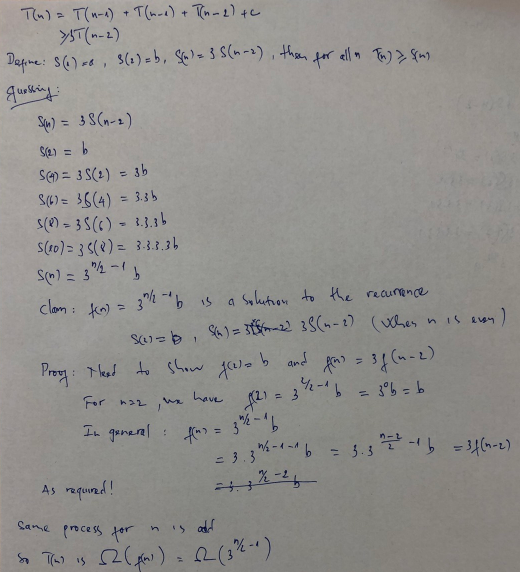
k++;

}

return r;

}

# Prob3:



# Prob4:

static List<HashSet<Integer>> powerSet(List<Integer> x) {

List<HashSet<Integer>> p = new ArrayList<HashSet<Integer>>();

p.add((new HashSet<Integer>()));

while (!x.isEmpty()) {

int f = x.remove(0);

List<HashSet<Integer>> pp = new ArrayList<HashSet<Integer>>();

for (HashSet<Integer> i : p) {

HashSet<Integer> j = (HashSet<Integer>) i.clone();

j.add(f);

pp.add(j);

}

p.addAll(pp);

}

return p;

}

# Prob5:

static int fib(int n) {

if (n == 0 || n == 1)

return n;

int n1 = 1, n2 = 0;

for (int i = 2; i <= n; i++) {

int t = n2 + n1;

n2 = n1;

n1 = t;

}

return n1;

}

# Prob6:

T(n) = T(n/2) + n; T(1) = 1

a=1, b=2, c=1, k=1

we have a(=1)<bk(=21)

=>Running time: Θ(n1) = Θ(n)