**Problem1:**

Text, letter

Description automatically generated

A piece of paper with writing on it

Description automatically generated

Text, letter

Description automatically generated

Text, letter

Description automatically generated

**Problem 2:** Show that for all n > 4, 2n < n!. Hint: Use induction.

Here we have n>4

Let’s increase the expression 2n < n! by 1.

2(n+1) < (n+1)!

2n \* 2 < n! \* (n+1)

As we can see from the above expression, left side is increased by 2 and right side is increased by (n+1).

2 < (n+1) , where n > 4

So, from the above result we can conclude that 2n < n! for all the value of n is greater than 4.

**Problem 3:**

**public** **class** prob3\_gcd {

**public** **static** **void** main(String[] args) {

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

}

**public** **static** **int** gcd(**int** m, **int** n) {

**int** remainder;

**while**(**true**) {

remainder = m % n;

**if**(remainder == 0) {

**return** n;

}**else** {

m = n;

n = remainder;

}

}

}

}

Problem4:

**public** **class** prob3\_gcd {

**public** **static** **void** main(String[] args) {

**int**[] arr = {81,3,4,6,5,8,97,4,92,9,63,86,98};

System.***out***.println(*secondSmallest*(arr));

}

**public** **static** **int** secondSmallest(**int**[] arr) {

**if**(arr==**null** || arr.length < 2) {

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

}

//implement

**int** smallest = arr[0];

**for**(**int** i = 1 ; i < arr.length ; i++) {

**if**(smallest > arr[i])

{

smallest = arr[i];

}

}

**int** secondSmallest = arr[0];

**for**(**int** i = 1 ; i < arr.length ; i++) {

**if**(secondSmallest > arr[i] && smallest != arr[i]) {

secondSmallest = arr[i];

}

}

**return** secondSmallest;

}

}

**Problem5:**

**private** **static** Set<Integer> sum(List<Integer> S, **int** k){

**if**(k ==0) **return** **new** HashSet<>();

List<Set<Integer>> powerset = *powerSetGenerator*(S);

**for**(Set<Integer> subset: powerset){

**int** sum = 0 ;

**for**(Integer x : subset){

sum += x;

}

**if**(sum == k)

**return** subset;

}

**return** **null**;

}

**public** **static** Set<Integer> subsetSum(List<Integer> S, **int** k) {

**if** (S.isEmpty())

**return** **null**;

**return** *sum*(S, k);

}

**private** **static** List<Set<Integer>>powerSetGenerator(List<Integer> X) {

List<Set<Integer>> P = **new** ArrayList<>();

P.add(**new** HashSet<>());

Iterator<Integer> iterator = X.iterator();

**while** (iterator.hasNext()) {

Integer f = iterator.next();

**int** size\_p = P.size();

**for** (**int** i = 0; i < size\_p; i++) {

Set<Integer> T = **new** HashSet<>();

T.addAll(P.get(i));

T.add(f);

P.add(T);

}

}

**return** P;

}