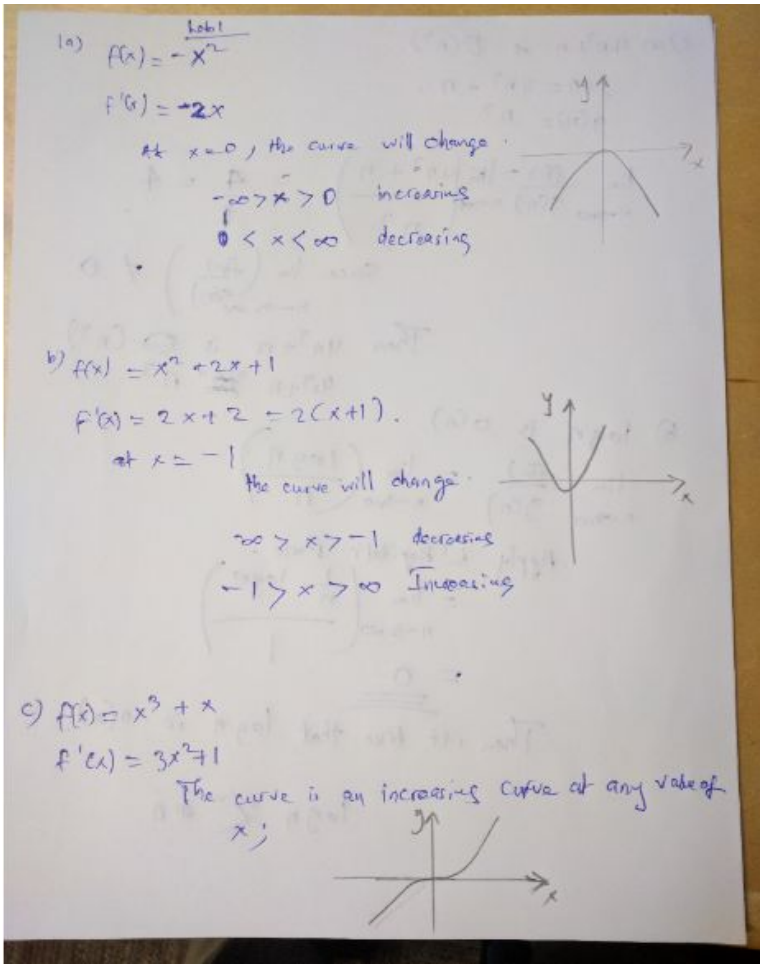
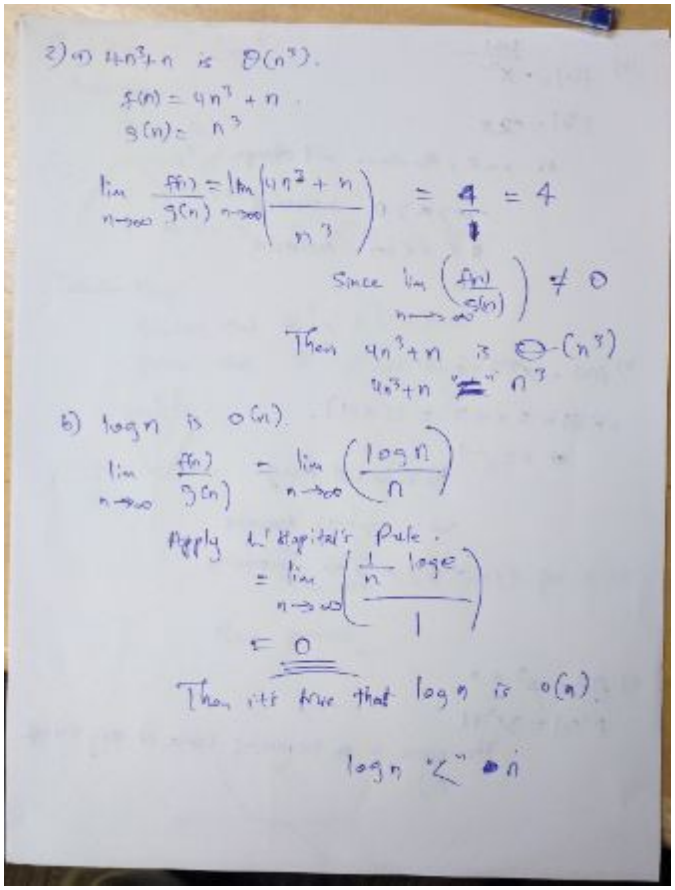
Problem 1


1a)
$f(x) = -x^2$
$f'(x) = -2x$
At $x = 0$, the curve will change
$-\infty > x > 0$ increasing
$0 < x < \infty$ decreasing
y
x
b) $f(x) = x^2 + 2x + 1$
$f'(x) = 2x + 2 = 2(x+1)$.
at $x = -1$
The curve will change
$\infty > x > -1$ decreasing
$-1 > x > \infty$ Increasing
y
x
c) $f(x) = x^3 + x$
$f'(x) = 3x^2 + 1$
The curve is an increasing curve at any value of x;
y
x

Problem2


a) $4n^3+n$ is $\Theta(n^3)$.
$f(n) = 4n^3 + n$
$g(n) = n^3$
$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty}\left(\frac{4n^3+n}{n^3}\right) = \frac{4}{1} = 4$
Since $\lim_{n\to\infty}\left(\frac{f(n)}{g(n)}\right) \neq 0$
Then $4n^3+n$ is $\Theta(n^3)$
$4n^3+n$ "=" n^3
b) $\log n$ is $o(n)$.
$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty}\left(\frac{\log n}{n}\right)$
Apply L'Hopital's Rule.
$= \lim_{n\to\infty}\left(\frac{\frac{1}{n}\log e}{1}\right)$
$= 0$
Then it's true that $\log n$ is $o(n)$.
$\log n$ "<" n

20 2^n is $\omega(n^2)$.

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty}\left(\frac{2^n}{n^2}\right)$$

Apply L'hopital's Rule

$$= \lim_{n\to\infty}\left(\frac{2^n \ln 2}{2n}\right) = \lim_{n\to\infty}(c\cdot 2^n)$$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$$

$$\lim_{n\to\infty} \frac{g(n)}{f(n)} = 0$$

2^n " $>$ " n^2

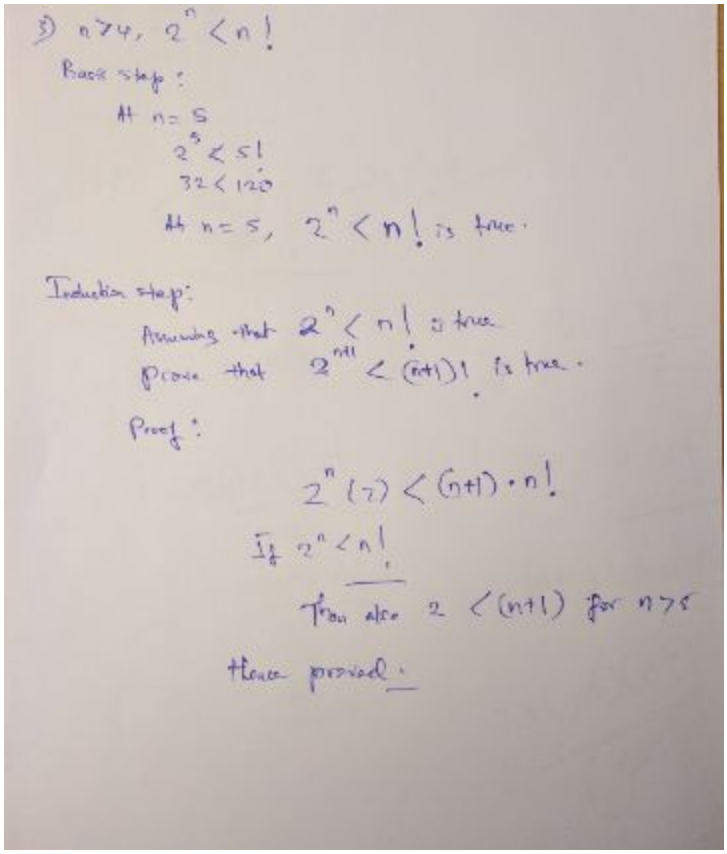
d) 2^n is $o(3^n)$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty}\left(\frac{2^n}{3^n}\right) = \lim_{n\to\infty}\left(\frac{2}{3}\right)^n$$

$$= 0$$

2^n " $<$ " 3^n

Problem 3



Problem 4

```
public int gcd(int m, int n){
      int d = m < n ? m : n;
      for(int i = d; i >= 1; i--){
            if((m%i == 0) && (n%i == 0)){
                  return i;
            }
      }
      return 1;
}
```

Problem 5

```
public int secondSmallest(int[] any){
      if(any==null || any.length < 2) {
            throw new IllegalArgumentException("Input array too small");
      }
      int smallest = Integer.MAX_VALUE;
      int secsmallest = Integer.MAX_VALUE;
      for(int i = 0; i < any.length; i++){
            if(any[i] < smallest){
                  secsmallest = smallest;
```

```
            smallest = any[i];
      }else if(any[i] < secsmallest && any[i] >= smallest){
            secsmallest = any[i];
            }
      }
      return secsmallest;
}
```

Problem 6

```
public Set<Integer> subsetSum(LinkedList<Integer> list, int target){
      if(list == null) return null;
      if(list.isEmpty()) return new HashSet<>();
      int sum = 0;
      for(Set<Integer> set : powerSet(list)){
            sum = set.parallelStream().reduce(0, (a,b)->a+b);
            if(sum == target) return set;
      }
      return null;
}


private LinkedList<Set<Integer>> powerSet(LinkedList<Integer> inputList){
      LinkedList<Set<Integer>> resultSet = new LinkedList<>();
      Set<Integer> emptySet = new HashSet<>();
      resultSet.add(emptySet);
      while(!inputList.isEmpty()){
      Integer f = inputList.removeFirst();
      int size = resultSet.size();
      Set<Integer> currentSet;
      for(int i = 0; i < size; i++){
            Set<Integer> union = new HashSet<>();
            currentSet = resultSet.get(i);
            union.add(f);
            union.addAll(currentSet);
            resultSet.add(union);
            }
      }
      return resultSet;
}
```