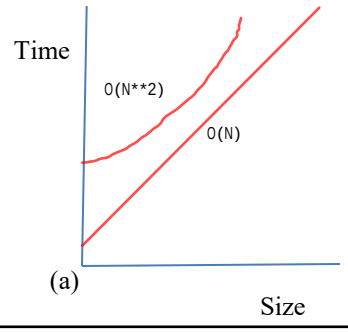
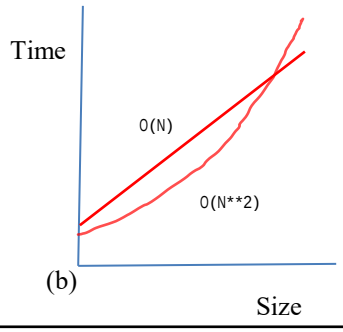
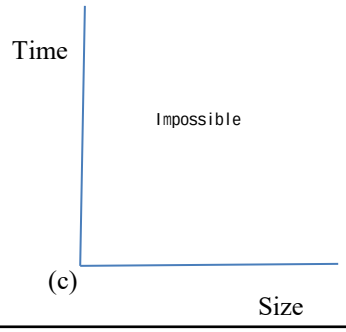
1.







2. (a) Because log is a really slowly growing function.

(b) When N is big enough, the f_a with lower complexity class will be faster than f_b.

(c) When the problem size is too small, the complexity class analysis cannot help to understand the behavior of different algorithms accurately.

3. (a1) 3.00x10**(-5)

(a2) 2.00 x10**(-7)+6.00 x10**(-4)

((b1) 3.00x10**(-1)

(b2) 2.00 x10**(-5)+6.00 x10**(-2)

(c1) 2

(c2) 0

4(a).

```
def sumsto_1 (alist,asum):
  for f in alist:                    N
      for s in alist:                N**2
          if f+s == asum:            N**2
              return (f,s)           N**2
  return None                        1
```

(b) 3N**2+N+1

(c) N**2

(d)

```
def sumsto_2 (alist,asum):
    aset = set(alist)                1
    for v in alist:                  N
        if asum-v in asset           N
            return(v,asum-v)         N
    return None                      1
```

(b) 3N+2

(c) N

5. (a) T(N)~ cN(LogN)**2
T(1000)~c1000(Log1000)**2~0.003
c*1000*100~0.003
c*100000~0.003
c~3*(10)**(-7)
formula: T(N)~3*(10)**(-7)*N*(LogN)**2

(b) 1000000/1000=1000
time=3*(10)**(-7)*1000+1000*0.003
time=3.0003

6.

N = Problem Size	Complexity Class	Time to Solve on Old Machine (secs)	M Solvable in the same Time on a New Machine 10x as Fast
10^6	$O(\text{Log}_2 N)$	1	$2^{**}(10*\text{Log}_2(ki10^{**}(6)))$
10^6	$O(N)$	1	10×10^6
10^6	$O(N \text{ Log}_2 N)$	1	$\sqrt{}(2^{**}((10^{**}7)*\log_2(10^{**}6)))$
10^6	$O(N^2)$	1	$\sqrt{}(10^{**}(13))$ square root