Show that if an algorithm makes at most a constant number of calls to polynomial-time subroutines and performs an additional amount of work that also takes polynomial time, then it runs in polynomial time. Also show that a polynomial number of calls to polynomial-time subroutines may result in an exponential-time algorithm.

1. 證明呼叫常數數量 run time in polynomial time 的 subroutines ，Runtime 也會在 polynomial time

定義一連串的 subroutine of S {S1,S2,S3,S4…..,Si}，run time is O(n^k)for every subroutine in S，所有的 subroutine Si(n)= n^k，定義一個演算法會依序呼叫 S 中的 subroutine， Si 的 polynomial runtime 定義為 pi(n)，且 pi(n)≦p(n)= n^k

用數學歸納法證明即使在很大的 return value 跟 worst case runtime 經過 i 次呼叫仍然都會在 O(p(n))

當 i=1

p1(n) =$n^k = O(n^k)$為 polynomial runtime

當第 i 次呼叫

pi(n) = p(p(. . .(p(n)). . .) =$n^{k^i} = O(n^k)$為 polynomial runtime

當第 i+1 次呼叫

Pi+1(n) = O($(\mathrm{pi}(\mathrm{n}))^k$) = $O(n^k)$ 還是 polynomial runtime

設 m 為 i<m **總 total runtime** 為

O($\sum_{i=1}^{m} pi(n)$)=O(mpm(n)) = O(mn^{k^m})為 polynomial runtime for any k and m

根據數學歸納法得證

2. 證明呼叫 polynomial time 後時間複雜度為 exponential-time

假設 S output 為 2 倍的 input，第 i 次呼叫 S 的 return value and runtime 會是 O(n2^i)

總 runtime 為

O($\sum_{i=1}^{n} pi(n)$) = O(n $\sum_{i=1}^{n} 2^i$) = O(n2^n)為 exponential-time