【1】
```
import math
    def inp(x):
        return complex(math.cos(x), math.sin(x))
    def is_power(n):
        return False if n<=0 else (n==1 or is_power(n>>1))
    def dft(xs):
        n=len(xs)
        return [sum(xs[k]*inp(2*math.pi*i*k/n) for k in range(n))
                for i in range(n)]
    def dft_inv(xs):
        n=len(xs)
        return [sum(xs[k]*exp(2*math.pi*i*k/n) for k in range(n))/n
                for i in range(n)]
```

(a) [(1+0j), (1+0j), (1+0j), (1+0j), (1+0j), (1+0j), (1+0j)]

(b) [(8+0j), (-5.55j), (-4,+8j), (-3,+3g), (-4.89+0j), (-2,+09j), (-2.973j), (3.55j)]

(c) [((c)(1)(1-16.1102-16j), (0.556-19.1102-16j), (0.500-16.184+15j), (1.133-15j), (213.4+-15j), (2+454-157)f5.298-15j), (-2,665-16j)]

(d) [(9+0j), (0.9994-16j), (1-4.289-16j), (1-6.6614-4j), 9+1-934+-15j], (0.999-2.332-15j),
      (1-1.469-15j), (0.999-1.076-15j)]

(f) 都不是

【2】
(b)   a = np.array([1,1,1,1])
      v = np.array([1,1])
      np.convolve(a, v, mode='full')

  => array([1,2,2,2,1])

【3】
(a) Convolution Theorem 其主要應用有幫助到 laplace 積分反轉換之運作。用比例花帽
    精的運算量
```