Runtime Practice

For each of the following pseudocode blocks, how many steps are needed to execute the code? Answer in big-O notation in terms of n. (Note: some are from CS106 but I know people had different versions.)
Credit: Jeff Knerr

1.
```
n = int(input("n: "))
for i in range(n):
  print(i)
```

2.
```
n = int(input("n: "))
for i in range(100):
  print(i*n)
```

3.
```
n = int(input("n: "))
for i in range(n):
  print(i)
for j in range(n):
  print(j)
```

4.
```
n = int(input("n: "))
for i in range(n):
  for j in range(n):
    print(i, j)
```

5.
```
n = int(input("n: "))
for i in range(n):
  for j in range(i,n):
    print(i, j)
```

6.
```
n = int(input("n: "))
for i in range(n):
  for j in range(10):
    print(i, j)
```

7.
```
n = int(input("n: "))
while n > 1:
  print(n)
  n = n/2
```

8.
```
arraylst = {1,2,5,7,13,21,24,25,26,33,34,38,50,57,58,63}
n = arraylst.length
mid = int(n/2)
print lst[mid]
```

9.
   ```
   n = int(input("n: "))
   for i in range(n):
     k = n
     while k > 1:
       print(i, k)
       k = k/2
   ```

10.
    ```
    n = int(input("n: "))
    T = 1
    for i in range(n):
      for t in range(T):
        print(i,t)
      T *= 2
    ```

11. Our analytic solution to linear regression was:

    $$\vec{w} = (X^T X)^{-1} X^T \vec{y}$$

    where the shape of X is $(n, p+1)$ and the shape of y is $(n, 1)$. What is the runtime (in big-O notation) of computing the analytic solution in terms of n and p?