Hsuan-You(Shaun) Lin (Aug. 25th, 2022)

1. My Answer: b) Procedure A is NOT an algorithm.
My reason: Because there is no break point in the program, and we don't know that is halts and when will finally satisfy the condition.

---

2. My Answer: a) Procedure A in an algorithm.
My reason: Because value i increase 1 every time in the while loop, when value i bigger than 10000000000, it will break out of the while loop, which means this program will be halts, and it will eventually satisfy the condition.

---

3. My Answer: c) It is not possible to decide of procedure A is an algorithm.
My reason: There's a function call "external_procedure()" didn't defined in this program, so this algorithm won't be work or we don't know that is halts.

---

4. My Answer: $O(N^2)$
My reason:

```
for i in range(N): → i=0 to N-1
    V[i] = 0
    for j in range(N):          → j=0 to N-1
        V[i] += A[i,j] * b[j]
```

∴ Complexity will be: $O(N) \times O(N) = \underline{O(N^2)}$ #

---

5. My Answer: $O(N^2)$
My reason:

```
for i in range(N): → i=0 to N-1
    V[i] = 0
    for j in range(i:N): →  j=0 to N-1
        V[i] += A[i,j]*b[j]   ↳j=1 to N-1
                              ↳j=2 to N-1
                              :
                              ↳j=N-1 to N-1
```

$\sum_{i=1}^{N} i = \frac{N \times N-1}{2} = \frac{1}{2}N^2 - \frac{1}{2}$

∴ Complexity will be: $\underline{O(N^2)}$ #

---

6. My Answer: $O(N \cdot M)$
My reason:

```
for i in range(N): → i=0 to N-1
    V[i] = 0
    for j in range(M):  → j=0 to M-1
        V[i] += A[i,j] * b[j]
```

∴ Complexity will be: $O(N) \times O(M) = \underline{O(N \cdot M)}$ #

7.

```
def pow(x, m):
    if m == 0:
        return 1
    if m % 2 == 0:
        return (pow(x, m//2)
        return tmp * tmp
    tmp = pow(x, m//2)
    return x * tmp * tmp
```

∴ the complexity of the algorithm is $O(\log(m))$ #

8.

```
def Bin(A,s,e,V):
    if e<s:
        return -1
    if s==e:
        if A[s]==V:
            return s
        else:
            return -1
    m=(s+e)//2
    if A[m]==V:
        return m
    if A[m]<V:
        return Bin(A,m+1,e,V)
    return Bin(A,s,m-1,V)
}
idx= Bin(A,0,len(A)-1,V)
```

} O(logn)

The work before recursive call is $O(1)$.

So recursion equation $T(N) = O(1) + T(\frac{N}{2})$

∴ the complexity will be $O(\log(N))$ #