William Friend

Hands-on Exercise #2: Run-time Analysis

For each of the following program fragments,

1. count the total number of operations. What is the big O value?
2. What is the return value if A = {1, 2, 3} and B = {1, 3, 5}?

1) Algorithm Ex1(A):

Input: An array A storing n ≥ 1 integers.
Output: The sum of the elements in A.
s ← A[0]
for i ← 1 to n - 1 **do**
    s ← s + A[i]
return s

Action	# of operations	Current total	Simplified total	Big O
s ← A[0]	1	1	1	
for i ← 1 to n - 1 do	(n-1)	1+(n-1)	n	
s ← s + A[i]	(n-1)	1+(n-1) + (n-1)	n + (n-1)	
return s	1	1+(n-1) + (n-1) + 1	2n	n

A = {1, 2, 3}	s	i	n
1, 2, 3	1	1	3
	1+2=3	2	
	3+3=6		

2) Algorithm Ex2(A):

Input: An array A storing n ≥ 1 integers.
Output: The sum of the elements at even cells in A.
s ← A[0]
for i ← 2 to n - 1 by increments of 2 **do**
    s ← s + A[i]
return s

Action	# of operations	Current total	Simplified total	Big O
s ← A[0]	1	1	1	
for i ← 2 to n - 1 by increments of 2 do	(n-1)	1+(n-1)	n	
s ← s + A[i]	(n-1)	1+(n-1) + (n-	n + (n-1)	

		1)		
return s	1	1+(n-1) + (n-1) + 1	2n	n

A = {1, 2, 3}	s	i	n
1, 2, 3	1	2	3
	1+3=4	2	

3) Algorithm Ex3(A):

Input: An array A storing $n \geq 1$ integers.
Output: The sum of the prefix sums in A.

```
s ← 0
for i ← 0 to n – 1 do
    s ← s + A[0]
    for j ← 1 to i do
        s ← s + A[j]
return s
```

A = {1, 2, 3}	s	i	j	n
1, 2, 3	0	0	?	3
s ← s + A[0]	0+1=1	0	?	
s ← s + A[j]	1+2=3	0	1	
s ← s + A[0]	3+1 = 4	1	1	
s ← s + A[j]	4+1 = 5	1	1	
s ← s + A[0]	5+1=6	2	1	
s ← s + A[j]	6+1=7	2	1	
s ← s + A[j]	7+2=9	2	2	
s ← s + A[0]	9+1=10	3	2	
s ← s + A[j]	10+1=11	3	1	
s ← s + A[j]	11+2=13	3	2	
s ← s + A[j]	11+3=16	3	3	

Action	# of operations	Current total	Simplified total	Big O
s ← 0	1	1	1	
for i ← 0 to n – 1 do	(n-1)	1+(n-1)	n	
s ← s + A[0]	(n-1)	1+(n-1) + (n-1)	n + (n-1)	
for j ← 1 to i do	(n-1)	[1+(n-1) + (n-1)] * *[(n-1)]		

s ← s + A[j]		[1+(n-1) + (n-1)] * *[(n-1) +(n-1)]		
return s	1	[1+(n-1) + (n-1)] * *[(n-1) +(n-1)] + 1	n * n	N^2

4) Algorithm Ex4(A):

Input: An array A storing n ≥ 1 integers.
Output: The sum of the prefix sums in A.

```
s ← A[0]
t ← s
for i ← 1 to n - 1 do
    s ← s + A[i]
    t ← t + s
return t
```

A = {1, 2, 3}	s	i	t	n
s ← A[0] t ← s	1	0	1	3
s ← s + A[i]	1+2=3	1	1	
t ← t + s	3	1	1+3=4	
s ← s + A[i]	3+3=6	2	4	
t ← t + s	6	2	4+6=10	

Action	# of operations	Current total	Simplified total	Big O
s ← A[0]	1	1	1	
t ← s	1	1+1+(n-1)	n	
for i ← 1 to n - 1 do	(n-1)	1+1+(n-1)	n + (n-1)	
s ← s + A[i]	(n-1)	1+1+(n-1) + (n-1)		
t ← t + s	(n-1)	1+1+(n-1) + (n-1) +(n-1)		
return t	1	1+1+(n-1) + (n-1) +(n-1) +1	n	O(n)

5) **Algorithm Ex5(A):**
   Input: Arrays A and B each storing $n \geq 1$ integers.
   Output: The number of elements in B equal to the sum of the prefix sums in A.

```
c ← 0
for i ← 0 to n - 1 do
    s ← 0
    for j ← 0 to i do
        s ← s + A[0]
        for k ← 1 to j do
            s ← s + A[k]
    if B[i] = s then
        c ← c + 1
return c
```

A={1,2,3}	B={1,3,5}	c	i	j	k	s
c ← 0 for i ← 0 to n - 1 do s ← 0		**0**	**0**	**?**	**?**	**0**
for j ← 0 to i do	**0**	**0**	**0**	**0**	**?**	**0**
s ← s + A[0]		**0**	**0**	**0**	**?**	**0+1=1**
for k ← 1 to j do		**0**	**0**	**0**	**1**	
s ← s + A[k]						
if B[i] = s then	**B[0]=1**					
c ← c + 1						
for i ← 0 to n - 1 do	**0**	**0**	**1**	**0**	**?**	**0**
s ← 0			**1**	**0**		**0**
for j ← 0 to i do			**1**	**0**		**0**
s ← s + A[0]			**1**	**0**		**0+1=1**
Skip to						
s ← s + A[0]			**1**	**1**		**1+1=2**
for k ← 1 to j do			**1**	**1**	**1**	
s ← s + A[k]			**1**	**1**	**1**	**2+2=4**
if B[i] = s then	**B[1]=3**					
c ← c + 1						
for i ← 0 to n - 1 do			**2**	**1**	**1**	
s ← 0			**2**	**1**	**1**	**0**

for j ← 0 to i do			**2**	**0**	**1**	
s ← s + A[0]						
Skip to						
s ← s + A[0]			**2**	**1**	**1**	
for k ← 1 to j do			**2**	**1**	**1**	
s ← s + A[k]			**2**	**1**	**1**	**0+3=3**
if B[i] = s then	**B[3]=n/a**					
c ← c + 1						
for j ← 0 to i do			**2**	**2**	**1**	
s ← s + A[0]			**2**	**2**	**1**	**3+1=4**
for k ← 1 to j do			**2**	**2**	**1**	
s ← s + A[k]			**2**	**2**	**1**	**4+2=6**
if B[i] = s then	**B[2]=5**		**2**	**2**	**1**	**6**
c ← c + 1						
for k ← 1 to j do			**2**	**2**	**2**	
s ← s + A[k]			**2**	**2**	**2**	**6+3=9**
if B[i] = s then						
c ← c + 1						
		0				

Action	# of operations	Current total	Simplified total	Big O
c ← 0	1	1	1	
for i ← 0 to n - 1 do	(n-1)	1+(n-1)	n	
s ← 0	(n-1)	1+(n-1) +(n-1)	n + (n-1)	
for j ← 0 to i do	(n-1)	1+(n-1) + (n-1) +(n-1) *(n-1)		
s ← s + A[0]	(n-1)	1+(n-1) + (n-1) +(n-1) *(n-1) +(n-1)		

for k ← 1 to j do	1	1+(n-1) + (n-1) +(n-1) *(n-1) +(n-1) *(n-1)	n	O(n)
s ← s + A[k]		1+(n-1) + (n-1) +(n-1) *(n-1) +(n-1) *(n-1) +(n-1)		
if B[i] = s then	1	1+(n-1) + (n-1) +(n-1) *(n-1) +(n-1) *(n-1) +(n-1) +1		
c ← c + 1	1	1+(n-1) + (n-1) +(n-1) *(n-1) +(n-1) *(n-1) +(n-1) +1 +1		
return c	1	1+(n-1) + (n-1) +(n-1) *(n-1) +(n-1) *(n-1) +(n-1) +1 +1 +1	n*n*n	O(n^3)