Find the asymptotic run time for each of the algorithms below. Do as many problems as you can before class on 9/6.

```
procedure SUM(A[1..n])
    total ← 0
    for j ← 1..n do
        total ← total + A[i]
    end for
    return total
end procedure
```

```
procedure COMBINATIONS(n)
    n ← 0
    for i ← 1..n do
        for j ← i + 1..n do
            n ← n + 1
        end for
    end for
end procedure
```

```
procedure EDITDISTANCE(A[1..n], B[1..n])
    Let C be an (n + 1) × (n + 1) matrix of all zeros
    for i ← 1..n do
        C[i + 1][0] ← i
    end for
    for j ← 1..n do
        C[0][j + 1] ← j
    end for
    for i ← 1..n do
        for j ← 1..n do
            if A[i] = B[j] then
                C[i + 1][j + 1] ← min{A[i + 1][j] + 1, A[i][j + 1] + 1, A[i][j]}
            else
                C[i + 1][j + 1] ← min{A[i + 1][j] + 1, A[i][j + 1] + 1, A[i][j] + 1}
            end if
        end for
    end for
    return C[n + 1][n + 1]
end procedure
```

```
procedure EUCLID(a, b)
    r ← a mod b
    while r ≠ 0 do
        a ← b
        b ← r
        r ← a mod b
    end while
    return b
end procedure
```

```
procedure INSERTIONSORT(A[1..n])
    for j ← 2..n do
        key ← A[j]
        i ← j − 1
        while i > 0 and A[i] > key do
            A[i + 1] ← A[i]
            i ← i − 1
        end while
        A[i + 1] ← key
    end for
end procedure
```

```
procedure FIBONACCI(n)
    if  thenn ≤ 2
        return 1
    else
        return FIBONACCI(n − 1) + FIBONACCI(n − 2)
    end if
end procedure
```

```
procedure MERGE(A[1..n], B[1..m])
    Let C be an array of length n + m
    i ← 1
    j ← 1
    while i < n and j < m do
        if A[i] < B[j] then
            C[i + j] ← A[i]
            i ← i + 1
        else
            C[i + j] ← B[j]
            j ← j + 1
        end if
    end while
    while i < n do
        C[i + j] ← A[i]
        i ← i + 1
    end while
    while j < m do
        C[i + j] ← B[j]
        j ← j + 1
    end while
    return C
end procedure


procedure MERGESORT(A[1..n])
    m = ⌊n/2⌋
    C ← MERGESORT(A[1..m])
    D ← MERGESORT(A[m + 1..n])
    return Merge(C, D)
end procedure
```