Introduction
CSCI 532

Problem: Given a sorted array of 0’s followed by 1’s (e.g., [0,0,0,1,1,1,1]), return the number of 1’s in the array.

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

How can we solve this problem?

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

```
countOnes_1(array a):
    count = 0
    for x in a:
        if x == 1:
            count += 1
    return count
```

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

```
countOnes_1(array a):
    count = 0
    for x in a:
        if x == 1:
            count += 1
    return count
```

0	0	0	1	1	1	1

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

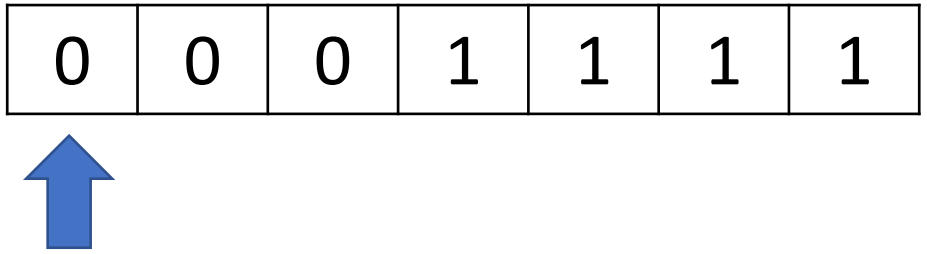
```
countOnes_1(array a):
    count = 0
    for x in a:
        if x == 1:
            count += 1
    return count
```

0	0	0	1	1	1	1

```
count = 0
```

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

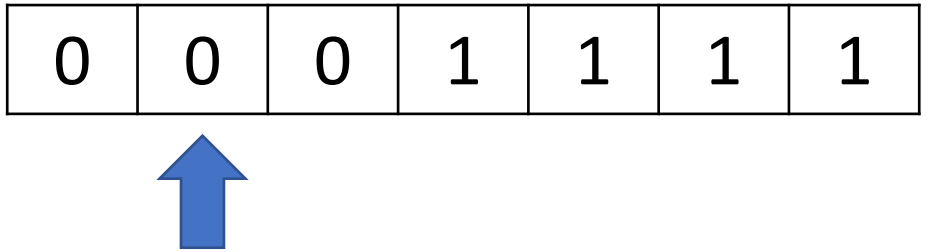
```
countOnes_1(array a):
    count = 0
    for x in a:
        if x == 1:
            count += 1
    return count
```



count = 0

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

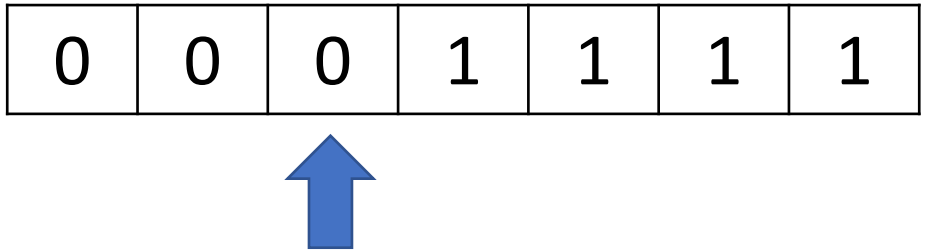
```
countOnes_1(array a):
    count = 0
    for x in a:
        if x == 1:
            count += 1
    return count
```



count = 0

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

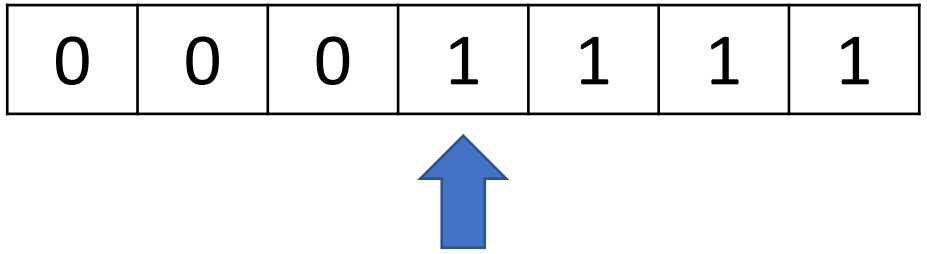
```
countOnes_1(array a):
    count = 0
    for x in a:
        if x == 1:
            count += 1
    return count
```



count = 0

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

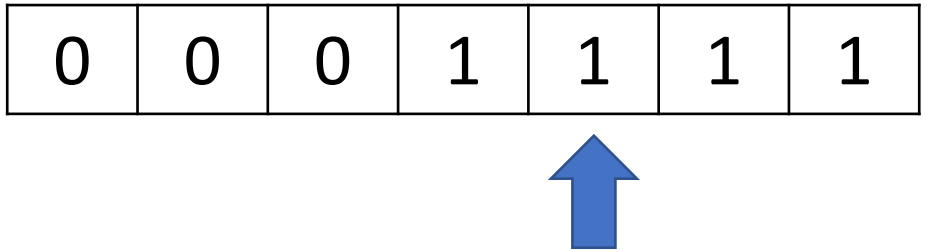
```
countOnes_1(array a):
    count = 0
    for x in a:
        if x == 1:
            count += 1
    return count
```



count = 1

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

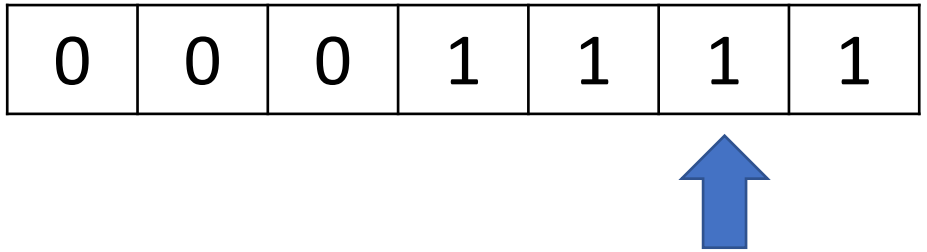
```
countOnes_1(array a):
    count = 0
    for x in a:
        if x == 1:
            count += 1
    return count
```



count = 2

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

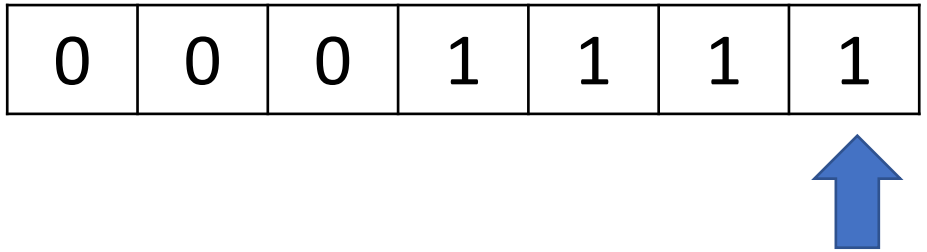
```
countOnes_1(array a):
    count = 0
    for x in a:
        if x == 1:
            count += 1
    return count
```



count = 3

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

```
countOnes_1(array a):
    count = 0
    for x in a:
        if x == 1:
            count += 1
    return count
```



count = 4

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

```
countOnes_1(array a):
   count = 0
   for x in a:
      if x == 1:
         count += 1
   return count
```

```
countOnes_2(array a):
   left = 0, right = a.length - 1
   first_one = -1
   while left <= right:
      mid = (left + right) / 2
      if a[mid] == 1:
         first_one = mid
         right = mid - 1
      else:
         left = mid + 1
   if first_one == -1:
      return 0
   else:
      return a.length - first_one
```

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

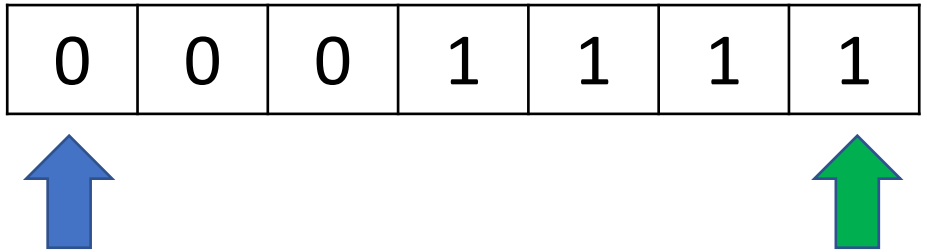
```
countOnes_1(array a):
   count = 0
   for x in a:
      if x == 1:
         count += 1
   return count
```

0	0	0	1	1	1	1

```
countOnes_2(array a):
   left = 0, right = a.length - 1
   first_one = -1
   while left <= right:
      mid = (left + right) / 2
      if a[mid] == 1:
         first_one = mid
         right = mid - 1
      else:
         left = mid + 1
   if first_one == -1:
      return 0
   else:
      return a.length - first_one
```

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

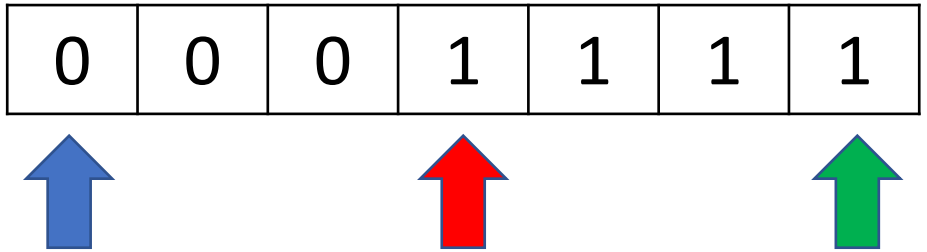
```
countOnes_1(array a):
    count = 0
    for x in a:
        if x == 1:
            count += 1
    return count
```




```
countOnes_2(array a):
    left = 0, right = a.length - 1
    first_one = -1
    while left <= right:
        mid = (left + right) / 2
        if a[mid] == 1:
            first_one = mid
            right = mid - 1
        else:
            left = mid + 1
    if first_one == -1:
        return 0
    else:
        return a.length - first_one
```

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

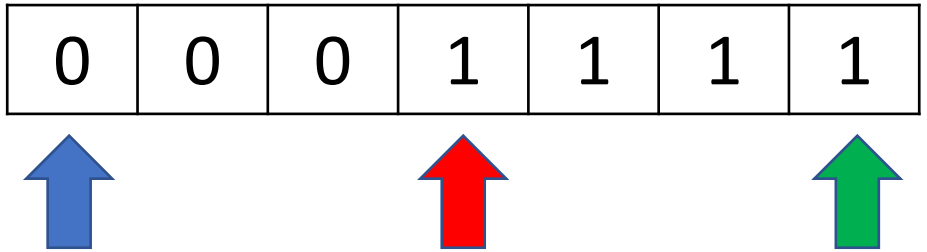
```
countOnes_1(array a):
    count = 0
    for x in a:
        if x == 1:
            count += 1
    return count
```




```
countOnes_2(array a):
    left = 0, right = a.length - 1
    first_one = -1
    while left <= right:
        mid = (left + right) / 2
        if a[mid] == 1:
            first_one = mid
            right = mid - 1
        else:
            left = mid + 1
    if first_one == -1:
        return 0
    else:
        return a.length - first_one
```

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

```
countOnes_1(array a):
    count = 0
    for x in a:
        if x == 1:
            count += 1
    return count
```

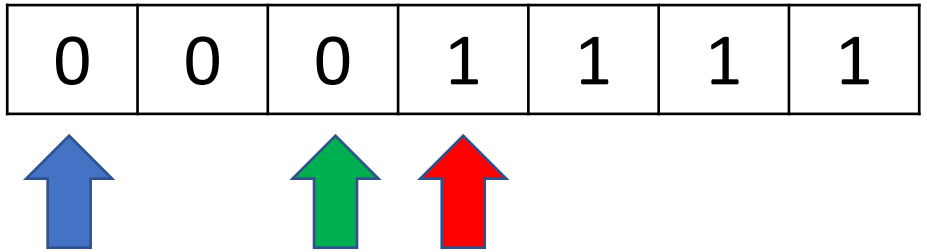


first_one = 3

```
countOnes_2(array a):
    left = 0, right = a.length - 1
    first_one = -1
    while left <= right:
        mid = (left + right) / 2
        if a[mid] == 1:
            first_one = mid
            right = mid - 1
        else:
            left = mid + 1
    if first_one == -1:
        return 0
    else:
        return a.length - first_one
```

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

```
countOnes_1(array a):
    count = 0
    for x in a:
        if x == 1:
            count += 1
    return count
```

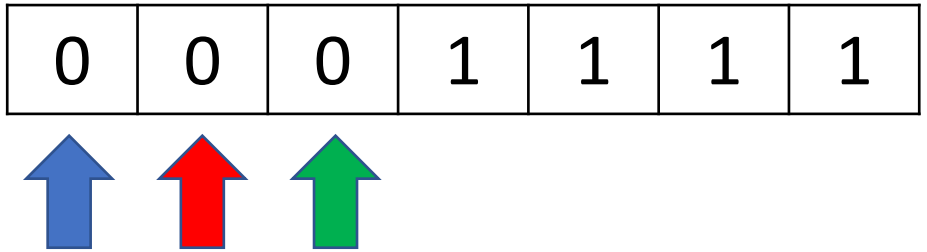


first_one = 3

```
countOnes_2(array a):
    left = 0, right = a.length - 1
    first_one = -1
    while left <= right:
        mid = (left + right) / 2
        if a[mid] == 1:
            first_one = mid
            right = mid - 1
        else:
            left = mid + 1
    if first_one == -1:
        return 0
    else:
        return a.length - first_one
```

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

```
countOnes_1(array a):
    count = 0
    for x in a:
        if x == 1:
            count += 1
    return count
```

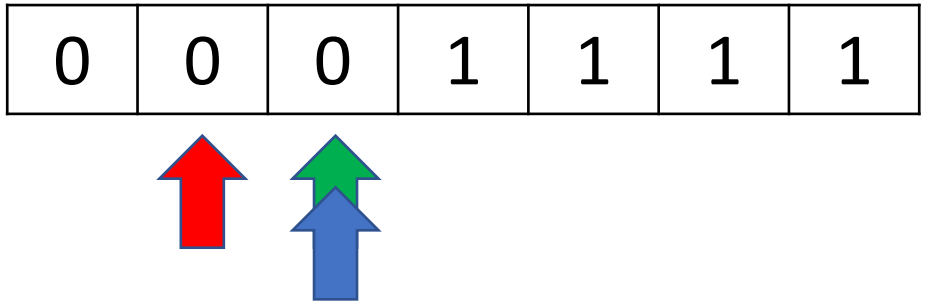


first_one = 3

```
countOnes_2(array a):
    left = 0, right = a.length - 1
    first_one = -1
    while left <= right:
        mid = (left + right) / 2
        if a[mid] == 1:
            first_one = mid
            right = mid - 1
        else:
            left = mid + 1
    if first_one == -1:
        return 0
    else:
        return a.length - first_one
```

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

```
countOnes_1(array a):
    count = 0
    for x in a:
        if x == 1:
            count += 1
    return count
```



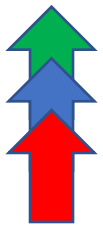
first_one = 3

```
countOnes_2(array a):
    left = 0, right = a.length - 1
    first_one = -1
    while left <= right:
        mid = (left + right) / 2
        if a[mid] == 1:
            first_one = mid
            right = mid - 1
        else:
            left = mid + 1
    if first_one == -1:
        return 0
    else:
        return a.length - first_one
```

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

```
countOnes_1(array a):
    count = 0
    for x in a:
        if x == 1:
            count += 1
    return count
```

0	0	0	1	1	1	1

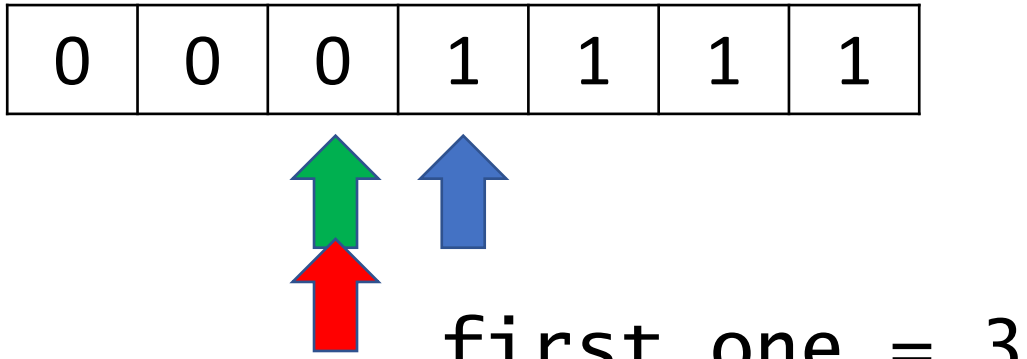


first_one = 3

```
countOnes_2(array a):
    left = 0, right = a.length - 1
    first_one = -1
    while left <= right:
        mid = (left + right) / 2
        if a[mid] == 1:
            first_one = mid
            right = mid - 1
        else:
            left = mid + 1
    if first_one == -1:
        return 0
    else:
        return a.length - first_one
```

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

```
countOnes_1(array a):
    count = 0
    for x in a:
        if x == 1:
            count += 1
    return count
```




```
countOnes_2(array a):
    left = 0, right = a.length - 1
    first_one = -1
    while left <= right:
        mid = (left + right) / 2
        if a[mid] == 1:
            first_one = mid
            right = mid - 1
        else:
            left = mid + 1
    if first_one == -1:
        return 0
    else:
        return a.length - first_one
```

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

```
countOnes_1(array a):
    count = 0
    for x in a:
        if x == 1:
            count += 1
    return count
```

0	0	0	1	1	1	1

first_one = 3

```
countOnes_2(array a):
    left = 0, right = a.length - 1
    first_one = -1
    while left <= right:
        mid = (left + right) / 2
        if a[mid] == 1:
            first_one = mid
            right = mid - 1
        else:
            left = mid + 1
    if first_one == -1:
        return 0
    else:
        return a.length - first_one
```

7 – 3 = 4

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

```
countOnes_1(array a):
   count = 0
   for x in a:
      if x == 1:
         count += 1
   return count
```

Which algorithm is best?

```
countOnes_2(array a):
   left = 0, right = a.length - 1
   first_one = -1
   while left <= right:
      mid = (left + right) / 2
      if a[mid] == 1:
         first_one = mid
         right = mid - 1
      else:
         left = mid + 1
   if first_one == -1:
      return 0
   else:
      return a.length - first_one
```

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

```
countOnes_1(array a):
   count = 0
   for x in a:
      if x == 1:
         count += 1
   return count
```

Which algorithm is best?

How can we compare algorithms?

```
countOnes_2(array a):
   left = 0, right = a.length - 1
   first_one = -1
   while left <= right:
      mid = (left + right) / 2
      if a[mid] == 1:
         first_one = mid
         right = mid - 1
      else:
         left = mid + 1
   if first_one == -1:
      return 0
   else:
      return a.length - first_one
```

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

```
countOnes_1(array a):
   count = 0
   for x in a:
      if x == 1:
         count += 1
   return count
```

Which algorithm is best?

How can we compare algorithms?

Running time, accuracy, simplicity, memory requirements, communication requirements,…

```
countOnes_2(array a):
   left = 0, right = a.length - 1
   first_one = -1
   while left <= right:
      mid = (left + right) / 2
      if a[mid] == 1:
         first_one = mid
         right = mid - 1
      else:
         left = mid + 1
   if first_one == -1:
      return 0
   else:
      return a.length - first_one
```

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

```
countOnes_1(array a):
   count = 0
   for x in a:
      if x == 1:
         count += 1
   return count
```

Running Time – The rough worst-case number of operations needed to run an algorithm.

Allows us to ignore hardware performance.

```
countOnes_2(array a):
   left = 0, right = a.length - 1
   first_one = -1
   while left <= right:
      mid = (left + right) / 2
      if a[mid] == 1:
         first_one = mid
         right = mid - 1
      else:
         left = mid + 1
   if first_one == -1:
      return 0
   else:
      return a.length - first_one
```

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

```
countOnes_1(array a):
   count = 0
   for x in a:
      if x == 1:
         count += 1
   return count
```

Running Time – The rough worst-case number of operations needed to run an algorithm.

items in a

$$1 + (1 + 1)n + 1$$

```
countOnes_2(array a):
   left = 0, right = a.length - 1
   first_one = -1
   while left <= right:
      mid = (left + right) / 2
      if a[mid] == 1:
         first_one = mid
         right = mid - 1
      else:
         left = mid + 1
   if first_one == -1:
      return 0
   else:
      return a.length - first_one
```

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

```
countOnes_1(array a):
    count = 0
    for x in a:
        if x == 1:
            count += 1
    return count
```

Running Time – The rough worst-case number of operations needed to run an algorithm.

items in a

$$1 + (1 + 1)n + 1 = 2n + 2 \in O(n)$$

Big-O Notation

```
countOnes_2(array a):
    left = 0, right = a.length - 1
    first_one = -1
    while left <= right:
        mid = (left + right) / 2
        if a[mid] == 1:
            first_one = mid
            right = mid - 1
        else:
            left = mid + 1
    if first_one == -1:
        return 0
    else:
        return a.length - first_one
```

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

```
countOnes_1(array a):
   count = 0
   for x in a:
      if x == 1:
         count += 1
   return count
```

Running Time:

```
countOnes_2(array a):
   left = 0, right = a.length - 1
   first_one = -1
   while left <= right:
      mid = (left + right) / 2
      if a[mid] == 1:
         first_one = mid
         right = mid - 1
      else:
         left = mid + 1
   if first_one == -1:
      return 0
   else:
      return a.length - first_one
```

Problem: Given a sorted array of 0's followed by 1's (e.g., [0,0,0,1,1,1,1]), return the number of 1's in the array.

```
countOnes_1(array a):
   count = 0
   for x in a:
      if x == 1:
         count += 1
   return count
```

Running Time:

$O(\log n)$ – Binary Search

```
countOnes_2(array a):
   left = 0, right = a.length - 1
   first_one = -1
   while left <= right:
      mid = (left + right) / 2
      if a[mid] == 1:
         first_one = mid
         right = mid - 1
      else:
         left = mid + 1
   if first_one == -1:
      return 0
   else:
      return a.length - first_one
```

CSCI 532

Goals:

- Learn algorithmic tools/techniques.
- Learn to evaluate algorithms.

What it is:

- Algorithmic thinking
- Mathematical proofs
- Application of data structures

What it is not:

- Programming
- An Introductory Course

Skills Check

- Not graded.
- Allows me to see where we are starting.