Algorithm performance

Analysis of selection sort

By the end of this video you will be able to…

- Analyze the performance of selection sort

Recall Selection Sort

4	7	2	10	1	8

Recall Selection Sort

4	7	2	10	1	8

1	7	2	10	4	8

Recall Selection Sort

4	7	2	10	1	8

1	7	2	10	4	8

1	2	7	10	4	8

Recall Selection Sort

4	7	2	10	1	8

1	7	2	10	4	8

1	2	7	10	4	8

1	2	4	10	7	8

```java
public static void selectionSort( int[] vals )     {

   int indexMin;

   for ( int i=0; i < vals.length-1 ; i++ ) {

         indexMin = i ;
         for ( int j=i+1; j < vals.length; j++ ) {
           if ( vals[j] < vals[indexMin] ) {
                 indexMin = j ;
           }
         }

         swap ( vals, indexMin , i );
   }
}
```

```java
public static void selectionSort( int[] vals )     {

   int indexMin;

   for ( int i=0; i < vals.length-1 ; i++ ) {

      indexMin = i ;
      for ( int j=i+1; j < vals.length; j++ ) {
        if ( vals[j] < vals[indexMin] ) {
            indexMin = j ;
        }
      }

      swap ( vals, indexMin , i );
   }
}
```

```
public static void selectionSort( int[] vals )    {

  int indexMin;

  for ( int i=0; i < vals.length-1 ; i++ ) {

      indexMin = i ;
      for ( int j=i+1; j < vals.length; j++ ) {
        if ( vals[j] < vals[indexMin] ) {
            indexMin = j ;
        }
      }

      swap ( vals, indexMin , i );
  }
}
```

```java
public static void selectionSort( int[] vals )    {

  int indexMin;

  for ( int i=0; i < vals.length; i++ ) {

      indexMin = i ;
      for ( int j=i+1; j < vals.length; j++ ) {
        if ( vals[j] < vals[indexMin] ) {
            indexMin = j ;
        }
      }

      swap ( vals, indexMin , i );
  }
}
```

```
public static void selectionSort( int[] vals )    {

  int indexMin;

  for ( int i=0; i < vals.length; i++ ) {

      indexMin = i ;
      for ( int j=i+1; j < vals.length; j++ ) {
        if ( vals[j] < vals[indexMin] ) {
            indexMin = j ;
        }
      }

      swap ( vals, indexMin , i );
  }
}
```

OUTER LOOP
n times

```
public static void selectionSort( int[] vals )    {

  int indexMin;

  for ( int i=0; i < vals.length; i++ ) {

      indexMin = i ;  O(1)
      for ( int j=i+1; j < vals.length; j++ ) {
        if ( vals[j] < vals[indexMin] ) {
            indexMin = j ;
        }
      }                          INNER LOOP

      swap ( vals, indexMin , i );  O(1)
  }
}
```

```
for ( int j=i+1; j < vals.length; j++ ) {
  if ( vals[j] < vals[indexMin] ) {
      indexMin = j ;
  }
}
```

O(1)

```
for ( int j=i+1; j < vals.length; j++ ) {

    O(1)

}
```

```
for ( int j=i+1; j < vals.length; j++ ) {

        O(1)

}
```

INNER FOR LOOP: O (n-(i+1))

```
for ( int i=0; i < vals.length; i++ ) {

    indexMin = i ;
    for ( int j=i+1; j < vals.length
      if (
    }

    swap ( vals, indexMin , i );
}
```

INNER FOR LOOP: O (n-(i+1))

```
for ( int i=0; i < vals.length; i++ ) {

    O(1)

    INNER FOR LOOP: O ( n-(i+1) )

    O(1)

}
```

```
for ( int i=0; i < vals.length; i++ ) {

}
```

$O(1) + O\ (\ n\text{-}i\text{-}1\ ) + O(1)$

```
for ( int i=0; i < vals.length; i++ ) {

        O ( n-i )

}
```

```
for ( int i=0; i < vals.length; i++ ) {

}
```

O (n-i)

Outer * inner is
O(n) * O(n-i)

```
for ( int i=0; i < vals.length; i++ ) {

}
```

O (n-i)

Outer * inner is
O(n) * O(n-i)

But what is O(n-i)?

```
for ( int i=0; i < vals.length; i++ ) {

}
```

O (n-i)

Outer * inner is
O(n) * O(n-i)

But what is O(n-i)?

As i increases, this
term decreases….

```
for ( int i=0; i < vals.length; i++ ) {
```

$O(n-i)$

Let's try to capture both loops in one series…

```
}
```

n +

```
for ( int i=0; i < vals.length; i++ ) {
```

O (n-i)

Let's try to capture both loops in one series…

```
}
```

n + (n-1) +

```
for ( int i=0; i < vals.length; i++ ) {
```

O (n-i)

Let's try to capture both loops in one series…

```
}
```

n + (n-1) + (n-2)

```
for ( int i=0; i < vals.length; i++ ) {
```

O (n-i)

Let's try to capture both loops in one series…

```
}
```

n + (n-1) + (n-2) + … + 1

```
for ( int i=0; i < vals.length; i++ ) {

                O ( n-i )

}
```

Captures the entire runtime!

$$(n-0) + (n-1) + (n-2) + \ldots + (n - (n-1))$$

```
for ( int i=0; i < vals.length; i++ ) {

        O ( n-i )

}
```

$$(n-0) + (n-1) + (n-2) + \ldots + (n-(n-1))$$

$$1 + 2 + 3 + \ldots + (n-1) + n$$

Reverse it!

How do we solve?

$$1 + 2 + 3 + \ldots + (n-2) + (n-1) + n$$

How do we solve?

$$1 + 2 + 3 + \ldots + (n-2) + (n-1) + n$$

These sum to n

How do we solve?

$$1 + 2 + 3 + \ldots + (n-2) + (n-1) + n$$

These sum to n

How do we solve?

$$\underbrace{1 + 2 + 3 + \ldots + (n-2) + (n-1)}_{\text{How many total pairs (in the range above) sum to n?}} + n$$

How do we solve?

$$\underbrace{1 + 2 + 3 + \ldots + (n-2) + (n-1)}_{\frac{(n-1)}{2} \text{ pairs sum to n}} + n$$

How do we solve?

$$\underbrace{1 + 2 + 3 + \ldots + (n-2) + (n-1)}_{\frac{(n-1)}{2}\text{ pairs sum to n}} + n$$

$\frac{(n-1)}{2}$ pairs sum to n → $\frac{n(n-1)}{2}$

How do we solve?

$$\underbrace{1 + 2 + 3 + \ldots + (n-2) + (n-1)}_{\frac{(n-1)}{2}\text{ pairs sum to } n \;\rightarrow\; \frac{n(n-1)}{2}} + n$$

$$= \frac{n(n-1)}{2} + n$$

Fun Fact (Gauss Summation Trick)

$$1 + 2 + 3 + \cdots + (n-1) = \frac{n(n-1)}{2}$$

Fun Fact (Gauss Summation Trick)

$$1 + 2 + 3 + \cdots + (n-1) = \frac{n(n-1)}{2}$$

Busy work: add all the numbers from 1 to 100

Fun Fact (Gauss Summation Trick)

$$1 + 2 + 3 + \cdots + (n-1) = \frac{n(n-1)}{2}$$

Busy work: add all the numbers from 1 to 100

$$= \frac{\mathbf{101(101-1)}}{\mathbf{2}} = \frac{\mathbf{10{,}100}}{\mathbf{2}} = \mathbf{5{,}050}$$

Wrapping up…

$$= \frac{n(n-1)}{2} + n$$

To finish the analysis, what is the tightest correct classification for the equation above?

IVQ

- What is the tightest correct classification for
- $f(n) = n(n-1)/2 + n$?
  - $O(n)$
  - $O(n-1)$
  - $O(n^2)$
  - $O(n(n-1))$

How do we solve?

$$= \frac{n(n-1)}{2} + n$$

$$= \frac{\boxed{n^2} - n}{2} + n$$

$$= \frac{n^2 - n}{2} + \frac{2n}{2}$$

$$= \frac{n^2 + n}{2} = O(n^2)$$

How do we solve?

$$= \frac{n(n-1)}{2} + n$$

$$= \frac{\boxed{n^2} - n}{2} + n$$

$$= \frac{n^2 - n}{2} + \frac{2n}{2}$$

$$= \frac{n^2 + n}{2} = O(n^2)$$

Selection sort's runtime is $O(n^2)$