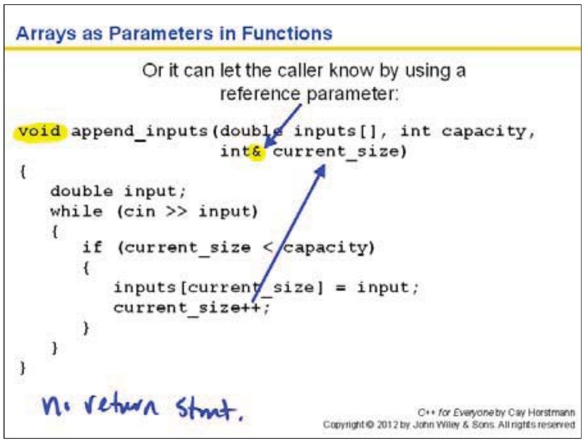


Arrays as Parameters in Functions

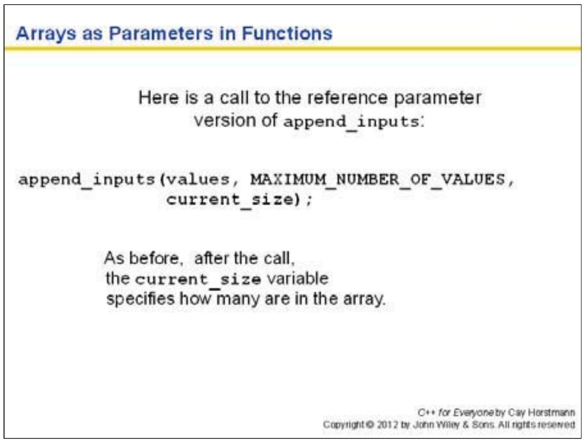
Or it can let the caller know by using a reference parameter:

```
void append_inputs(double inputs[], int capacity,
                   int& current_size)
{
   double input;
   while (cin >> input)
   {
      if (current_size < capacity)
      {
         inputs[current_size] = input;
         current_size++;
      }
   }
}
```

no return stmt.

C++ for Everyone by Cay Horstmann

Copyright © 2012 by John Wiley & Sons. All rights reserved



Arrays as Parameters in Functions

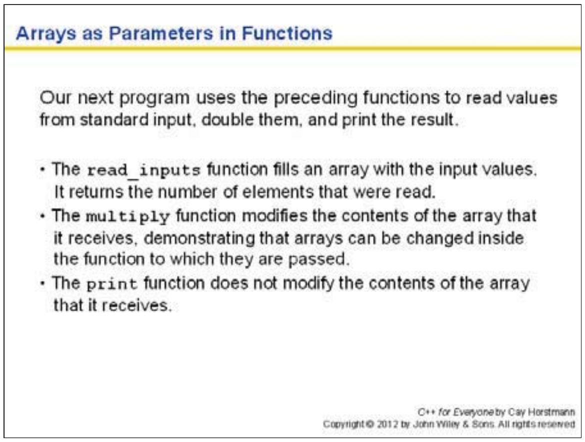
Here is a call to the reference parameter version of `append_inputs`:

```
append_inputs(values, MAXIMUM_NUMBER_OF_VALUES,
              current_size);
```

As before, after the call, the `current_size` variable specifies how many are in the array.

C++ for Everyone by Cay Horstmann

Copyright © 2012 by John Wiley & Sons. All rights reserved



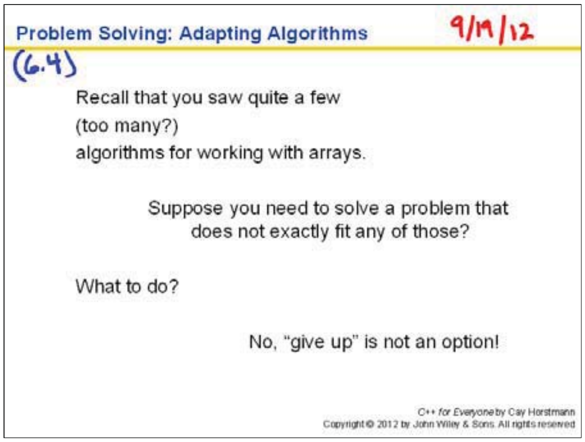
Arrays as Parameters in Functions

Our next program uses the preceding functions to read values from standard input, double them, and print the result.

- The `read_inputs` function fills an array with the input values. It returns the number of elements that were read.
- The `multiply` function modifies the contents of the array that it receives, demonstrating that arrays can be changed inside the function to which they are passed.
- The `print` function does not modify the contents of the array that it receives.

C++ for Everyone by Cay Horstmann

Copyright © 2012 by John Wiley & Sons. All rights reserved



Problem Solving: Adapting Algorithms

9/19/12

(6.4)

Recall that you saw quite a few
(too many?)
algorithms for working with arrays.

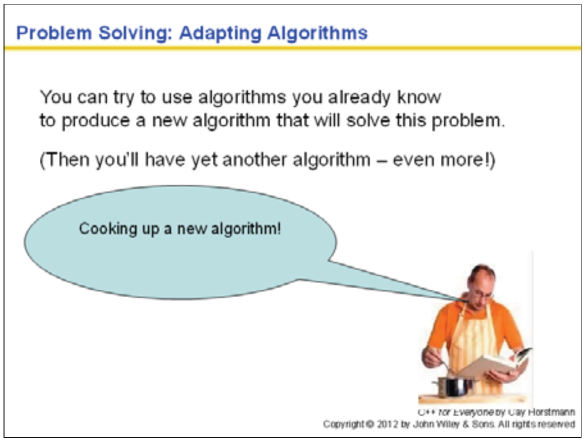
Suppose you need to solve a problem that does not exactly fit any of those?

What to do?

No, "give up" is not an option!

C++ for Everyone by Cay Horstmann

Copyright © 2012 by John Wiley & Sons. All rights reserved



Problem Solving: Adapting Algorithms

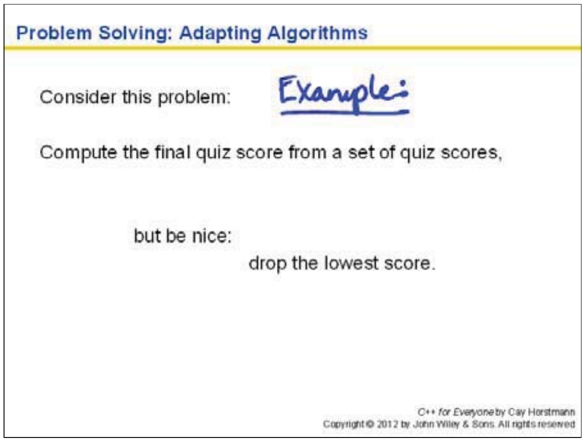
You can try to use algorithms you already know to produce a new algorithm that will solve this problem.

(Then you'll have yet another algorithm – even more!)

Cooking up a new algorithm!

C++ for Everyone by Cay Horstmann

Copyright © 2012 by John Wiley & Sons. All rights reserved



Problem Solving: Adapting Algorithms

Consider this problem: Example:

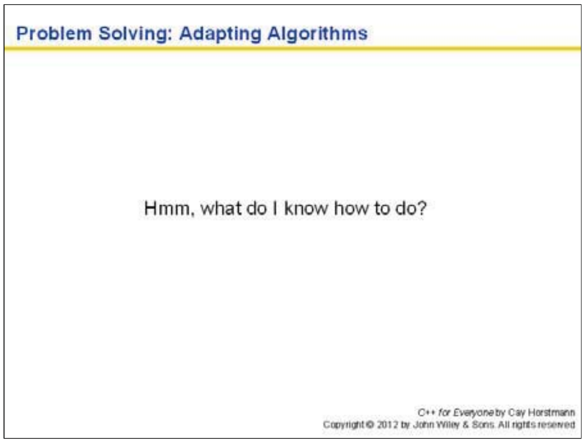
Compute the final quiz score from a set of quiz scores,

but be nice:

drop the lowest score.

C++ for Everyone by Cay Horstmann

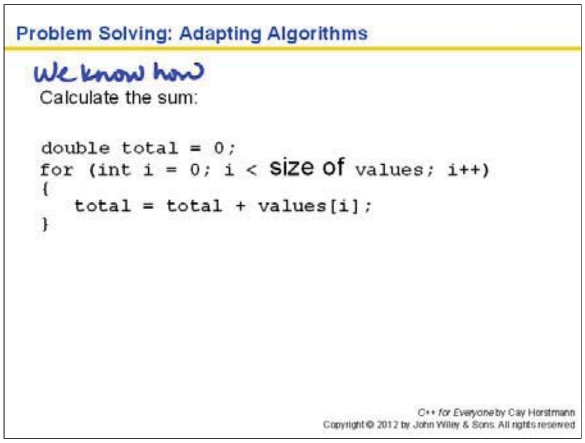
Copyright © 2012 by John Wiley & Sons. All rights reserved



Problem Solving: Adapting Algorithms

Hmm, what do I know how to do?

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



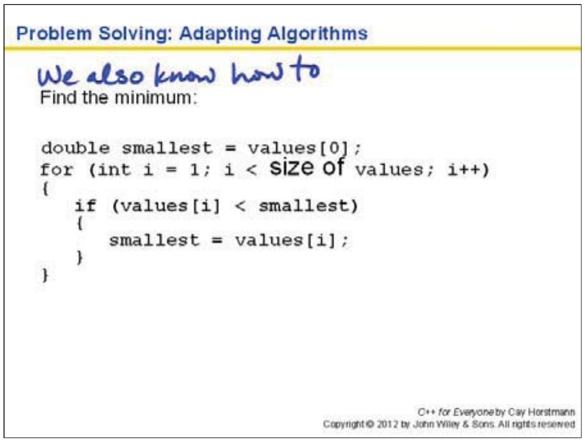
Problem Solving: Adapting Algorithms

We know how

Calculate the sum:

```
double total = 0;
for (int i = 0; i < size of values; i++)
{
   total = total + values[i];
}
```

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



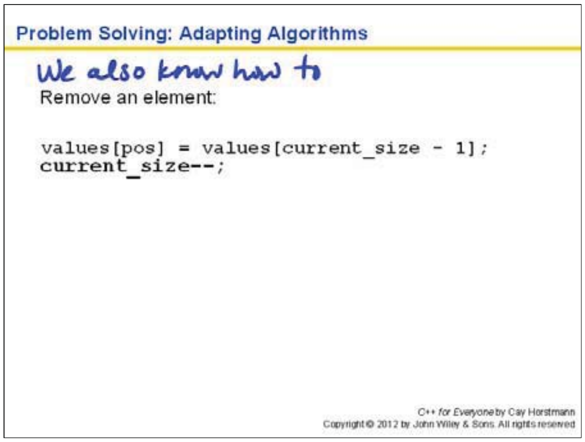
Problem Solving: Adapting Algorithms

We also know how to

Find the minimum:

```
double smallest = values[0];
for (int i = 1; i < size of values; i++)
{
   if (values[i] < smallest)
   {
      smallest = values[i];
   }
}
```

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



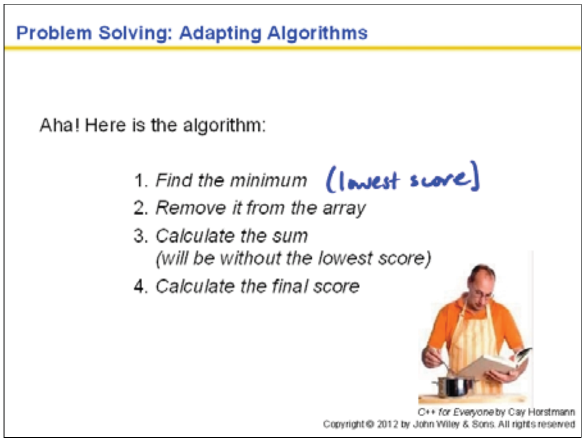
Problem Solving: Adapting Algorithms

We also know how to

Remove an element:

```
values[pos] = values[current_size - 1];
current_size--;
```

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved

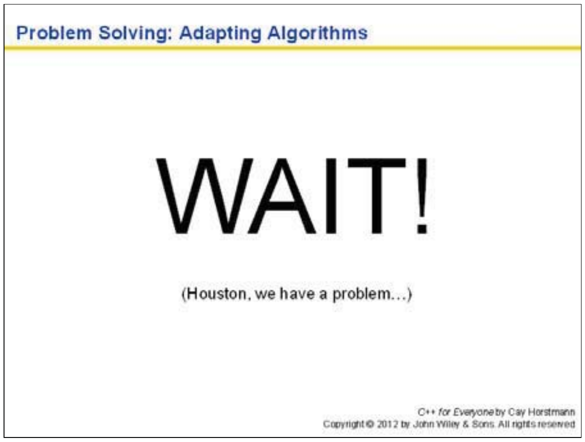


Problem Solving: Adapting Algorithms

Aha! Here is the algorithm:

1. *Find the minimum* (lowest score)
2. *Remove it from the array*
3. *Calculate the sum*
   (will be without the lowest score)
4. *Calculate the final score*

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved

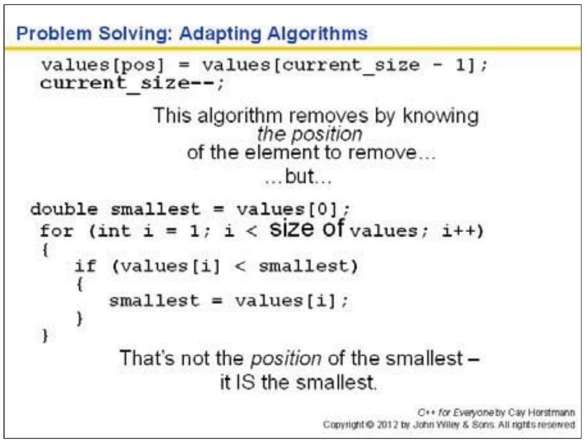


Problem Solving: Adapting Algorithms

WAIT!

(Houston, we have a problem…)

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



Problem Solving: Adapting Algorithms

```
 values[pos] = values[current_size - 1];
 current_size--;
```

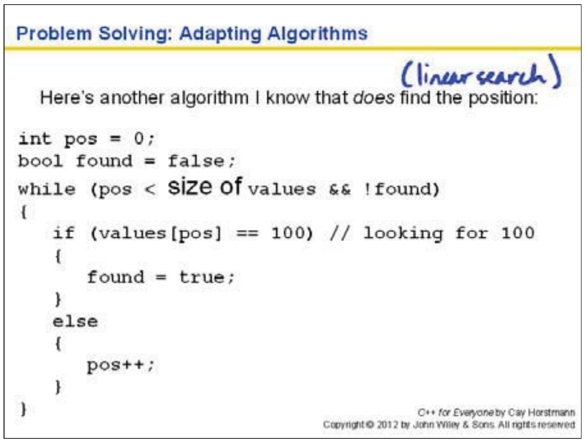
This algorithm removes by knowing *the position* of the element to remove…

…but…

```
double smallest = values[0];
 for (int i = 1; i < size of values; i++)
 {
    if (values[i] < smallest)
    {
       smallest = values[i];
    }
 }
```

That's not the *position* of the smallest – it IS the smallest.

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



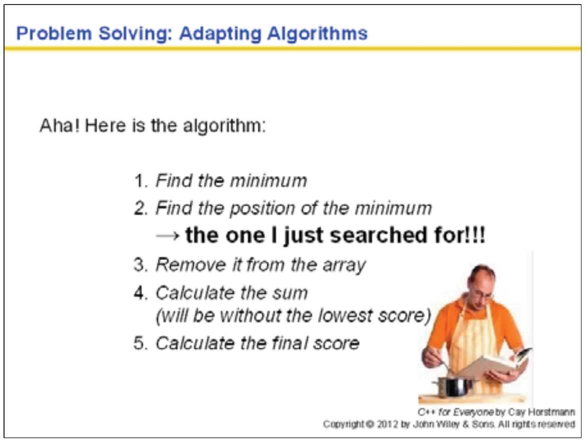
Problem Solving: Adapting Algorithms

(linear search)

Here's another algorithm I know that *does* find the position:

```
int pos = 0;
bool found = false;
while (pos < size of values && !found)
{
   if (values[pos] == 100) // looking for 100
   {
      found = true;
   }
   else
   {
      pos++;
   }
}
```

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved

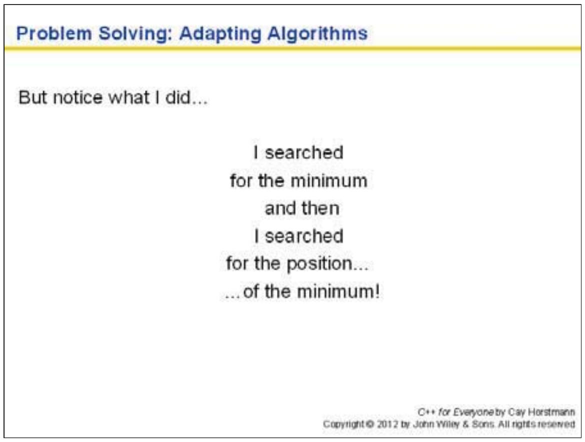


Problem Solving: Adapting Algorithms

Aha! Here is the algorithm:

1. *Find the minimum*
2. *Find the position of the minimum*
   → **the one I just searched for!!!**
3. *Remove it from the array*
4. *Calculate the sum*
   (will be without the lowest score)
5. *Calculate the final score*

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved

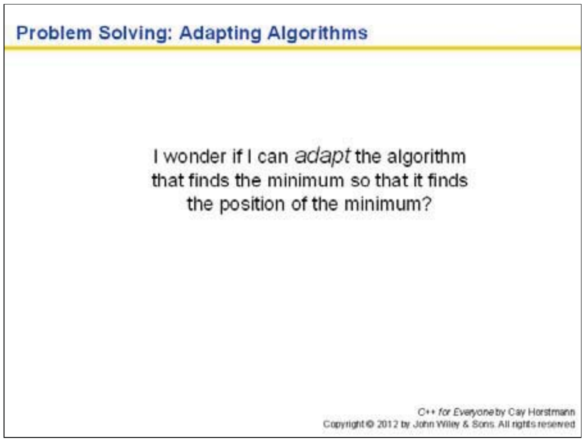


Problem Solving: Adapting Algorithms

But notice what I did…

I searched
for the minimum
and then
I searched
for the position…
…of the minimum!

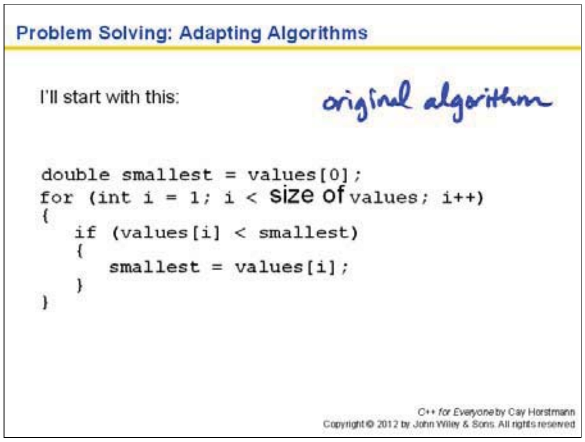
C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



Problem Solving: Adapting Algorithms

I wonder if I can *adapt* the algorithm that finds the minimum so that it finds the position of the minimum?

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved

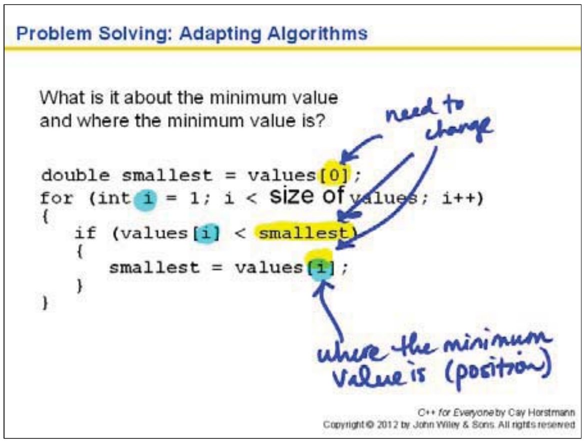


Problem Solving: Adapting Algorithms

I'll start with this: original algorithm

```
double smallest = values[0];
for (int i = 1; i < size of values; i++)
{
   if (values[i] < smallest)
   {
      smallest = values[i];
   }
}
```

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



Problem Solving: Adapting Algorithms

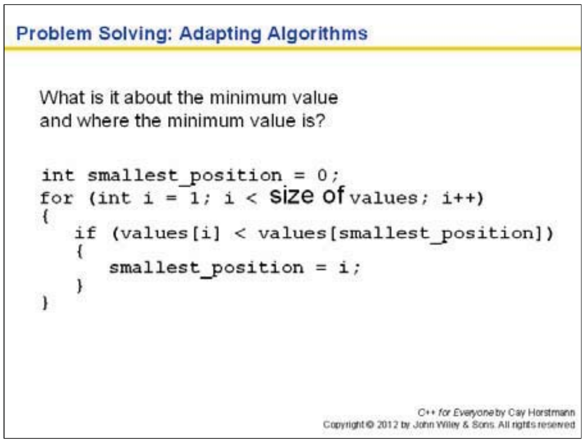
What is it about the minimum value
and where the minimum value is?

need to change

```
double smallest = values[0];
for (int i = 1; i < size of values; i++)
{
   if (values[i] < smallest)
   {
      smallest = values[i];
   }
}
```

where the minimum value is (position)

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved

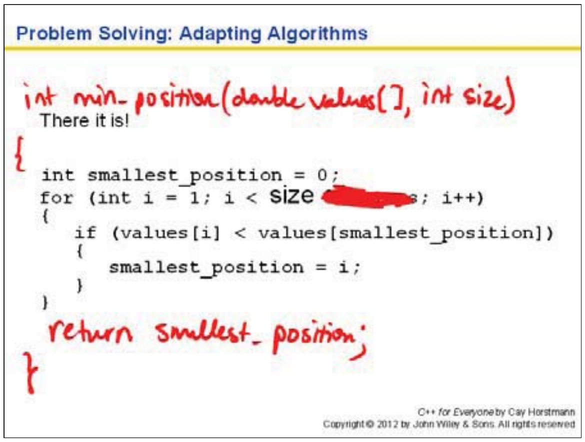


Problem Solving: Adapting Algorithms

What is it about the minimum value
and where the minimum value is?

```
int smallest_position = 0;
for (int i = 1; i < size of values; i++)
{
   if (values[i] < values[smallest_position])
   {
      smallest_position = i;
   }
}
```

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



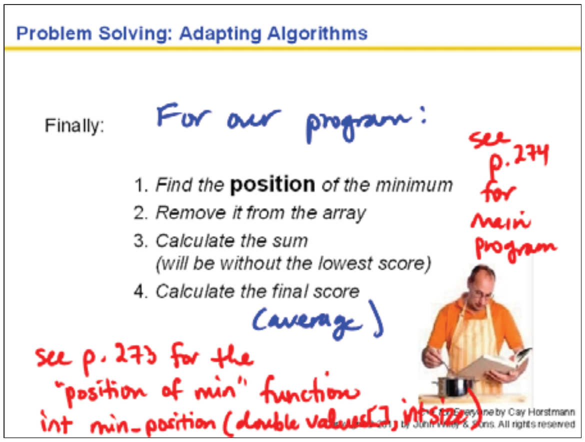
Problem Solving: Adapting Algorithms

int min_position(double values[], int size)

There it is!

```
{
int smallest_position = 0;
for (int i = 1; i < size [redacted]s; i++)
{
   if (values[i] < values[smallest_position])
   {
      smallest_position = i;
   }
}
return smallest_position;
}
```

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



Problem Solving: Adapting Algorithms

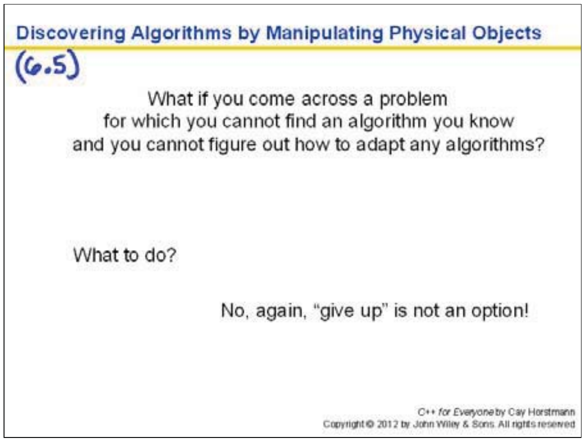
Finally: For our program:

see p.274 for main program

1. *Find the* **position** *of the minimum*
2. *Remove it from the array*
3. *Calculate the sum*
   (will be without the lowest score)
4. *Calculate the final score*
   (average)

see p.273 for the "position of min" function
int min_position(double values[], int size)

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



Discovering Algorithms by Manipulating Physical Objects

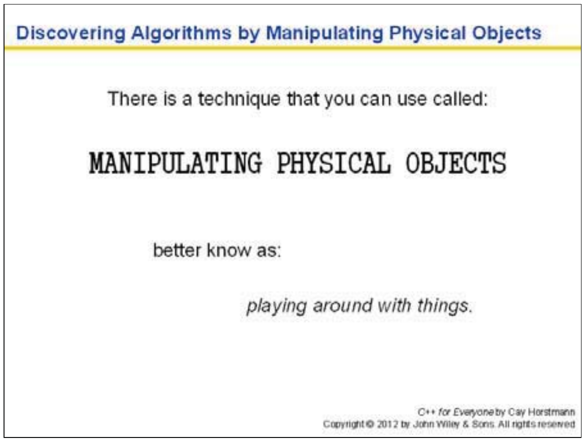
(6.5)

What if you come across a problem
for which you cannot find an algorithm you know
and you cannot figure out how to adapt any algorithms?

What to do?

No, again, "give up" is not an option!

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



Discovering Algorithms by Manipulating Physical Objects

There is a technique that you can use called:

MANIPULATING PHYSICAL OBJECTS

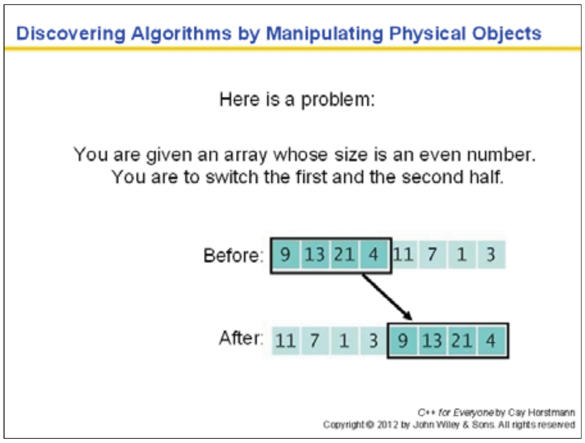
better know as:

playing around with things.

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved

Discovering Algorithms by Manipulating Physical Objects

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved

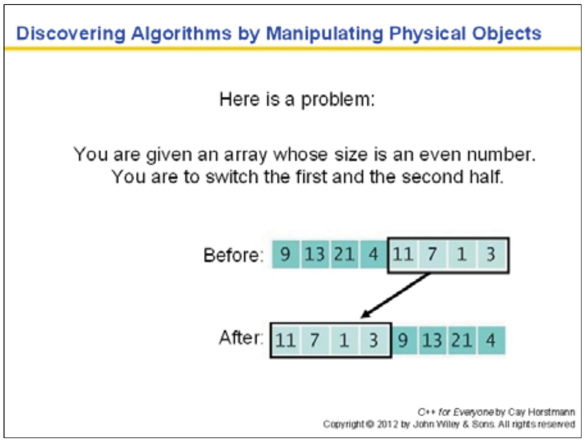


Discovering Algorithms by Manipulating Physical Objects

Here is a problem:

You are given an array whose size is an even number.
You are to switch the first and the second half.

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



Discovering Algorithms by Manipulating Physical Objects

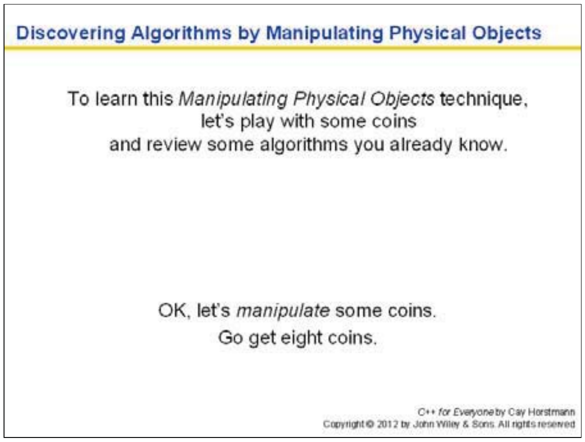
Here is a problem:

You are given an array whose size is an even number.
You are to switch the first and the second half.

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved

Discovering Algorithms by Manipulating Physical Objects

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved

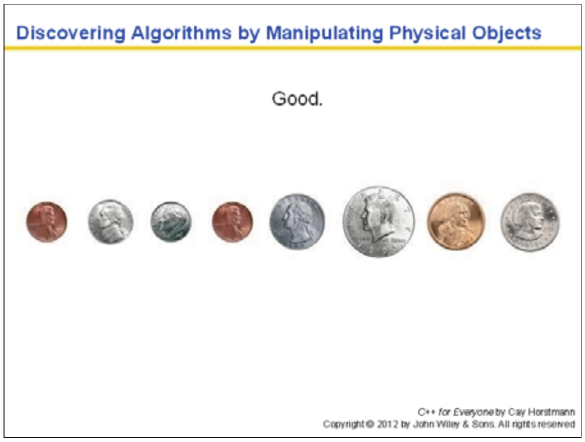


Discovering Algorithms by Manipulating Physical Objects

To learn this *Manipulating Physical Objects* technique,
let's play with some coins
and review some algorithms you already know.

OK, let's *manipulate* some coins.
Go get eight coins.

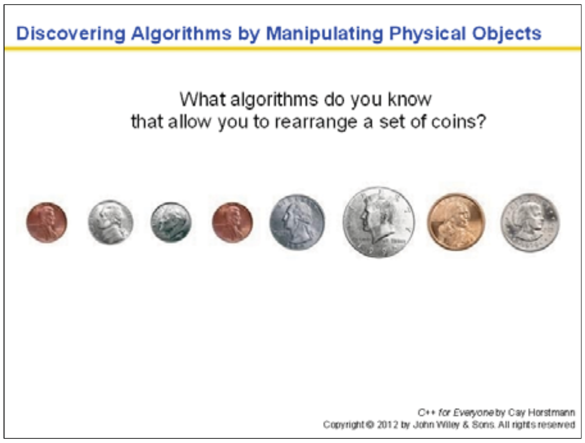
C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



Discovering Algorithms by Manipulating Physical Objects

Good.

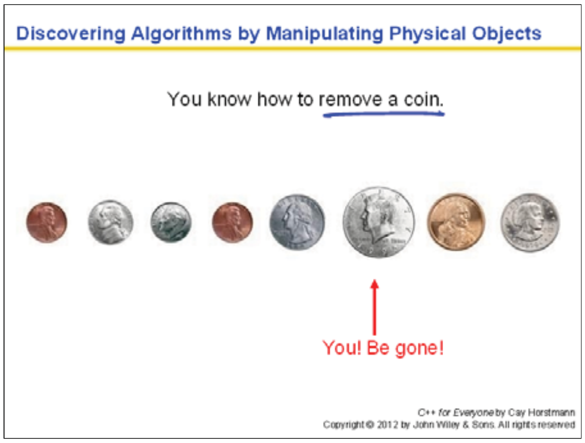
C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



Discovering Algorithms by Manipulating Physical Objects

What algorithms do you know
that allow you to rearrange a set of coins?

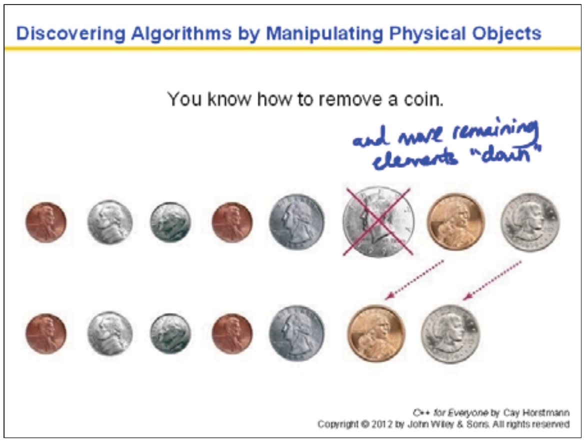
C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



Discovering Algorithms by Manipulating Physical Objects

You know how to remove a coin.

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved

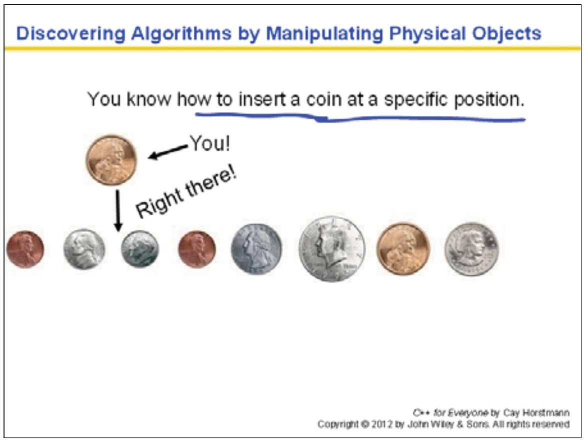


Discovering Algorithms by Manipulating Physical Objects

You know how to remove a coin.

and move remaining
elements "down"

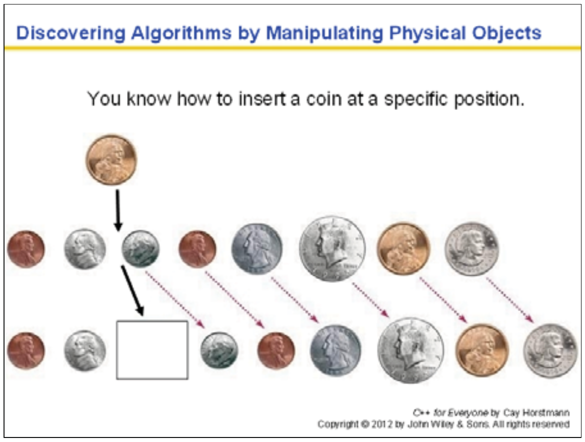
C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



Discovering Algorithms by Manipulating Physical Objects

You know how to insert a coin at a specific position.

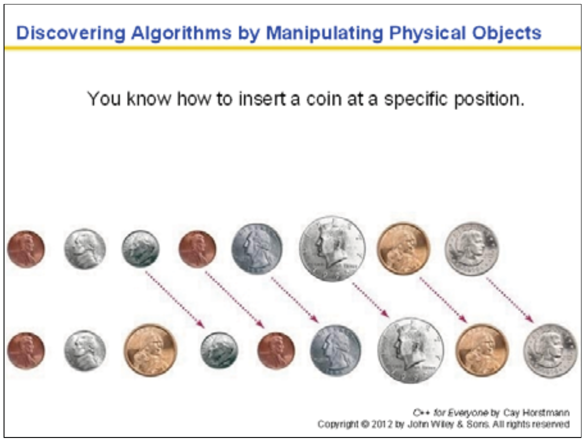
C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



Discovering Algorithms by Manipulating Physical Objects

You know how to insert a coin at a specific position.

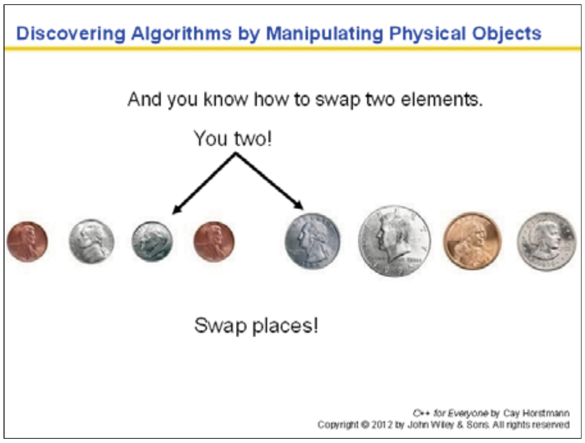
C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



Discovering Algorithms by Manipulating Physical Objects

You know how to insert a coin at a specific position.

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



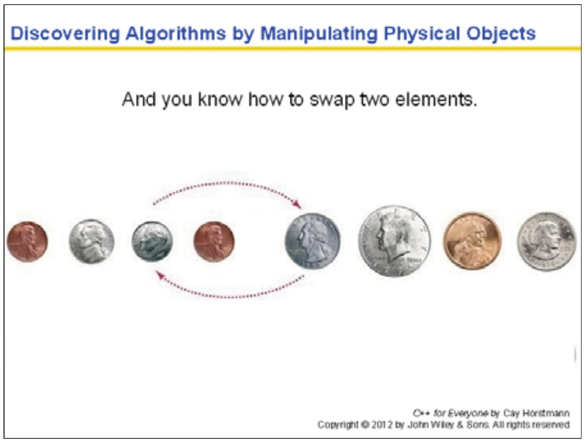
Discovering Algorithms by Manipulating Physical Objects

And you know how to swap two elements.

You two!

Swap places!

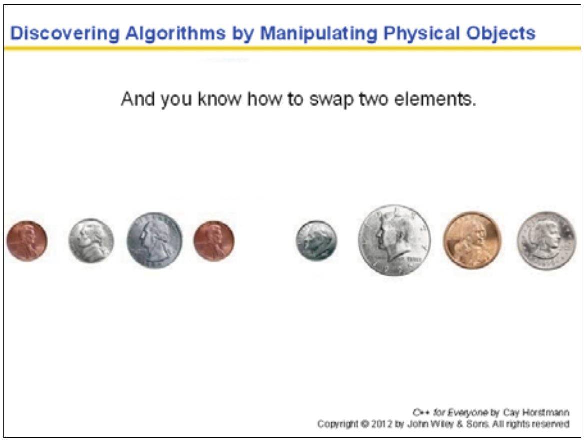
C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



Discovering Algorithms by Manipulating Physical Objects

And you know how to swap two elements.

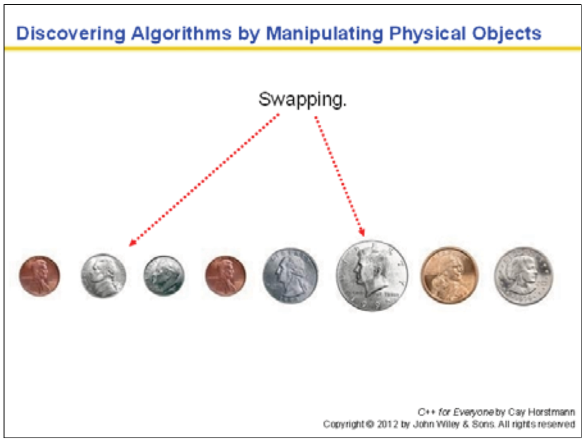
C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



Discovering Algorithms by Manipulating Physical Objects

And you know how to swap two elements.

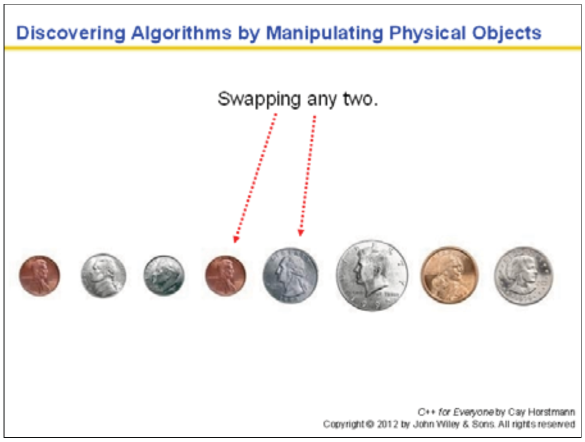
C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



Discovering Algorithms by Manipulating Physical Objects

Swapping.

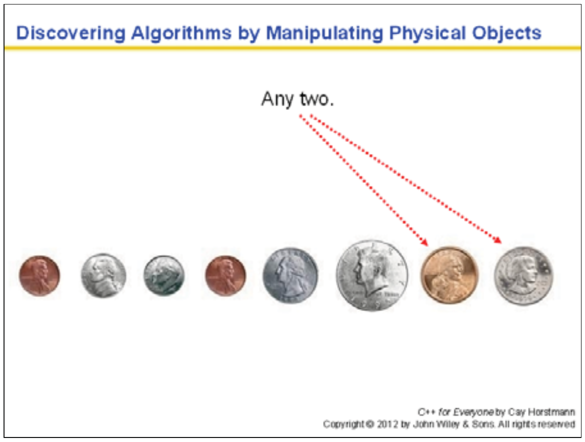
C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



Discovering Algorithms by Manipulating Physical Objects

Swapping any two.

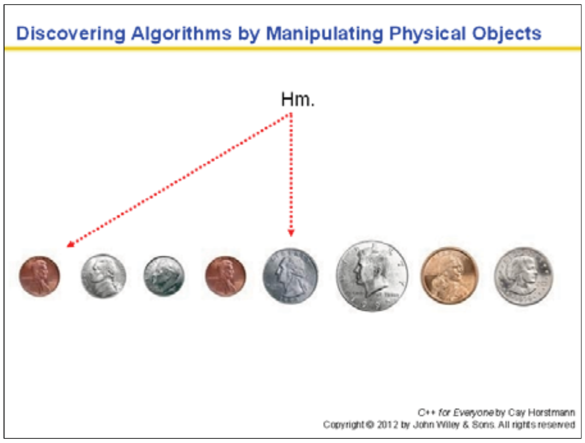
C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



Discovering Algorithms by Manipulating Physical Objects

Any two.

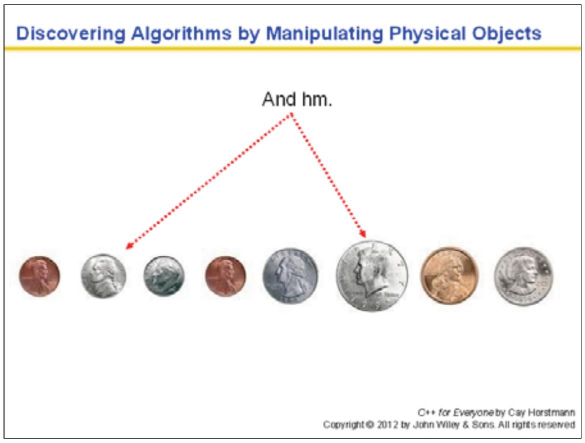
C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



Discovering Algorithms by Manipulating Physical Objects

Hm.

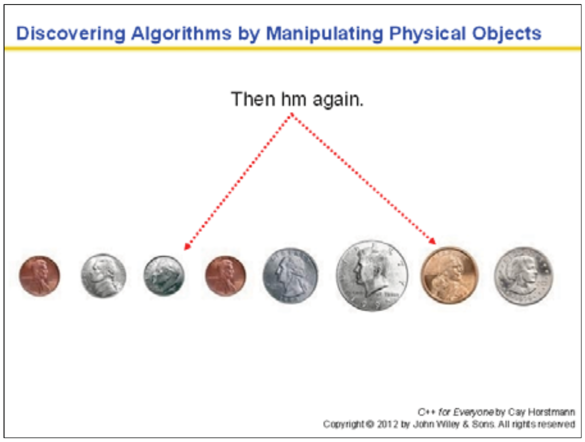
C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



Discovering Algorithms by Manipulating Physical Objects

And hm.

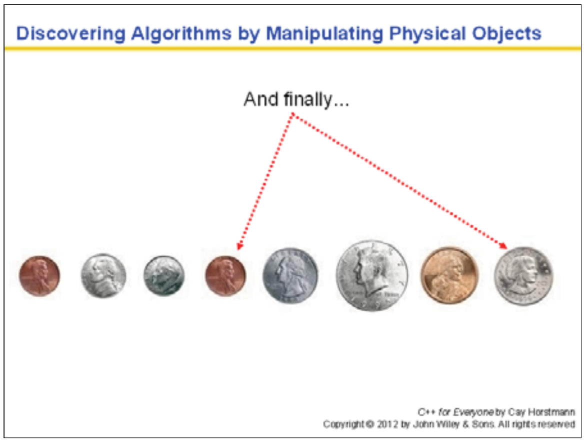
C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



Discovering Algorithms by Manipulating Physical Objects

Then hm again.

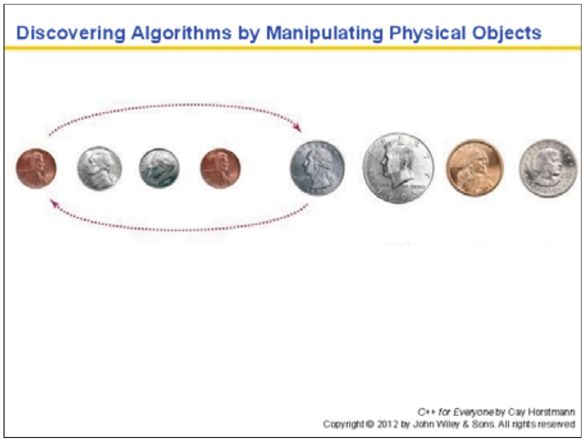
C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



Discovering Algorithms by Manipulating Physical Objects

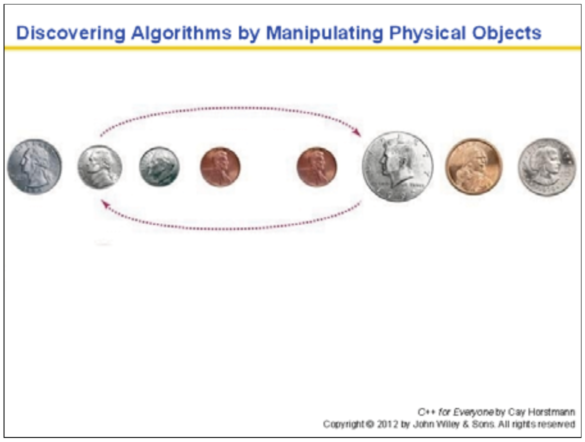
And finally…

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



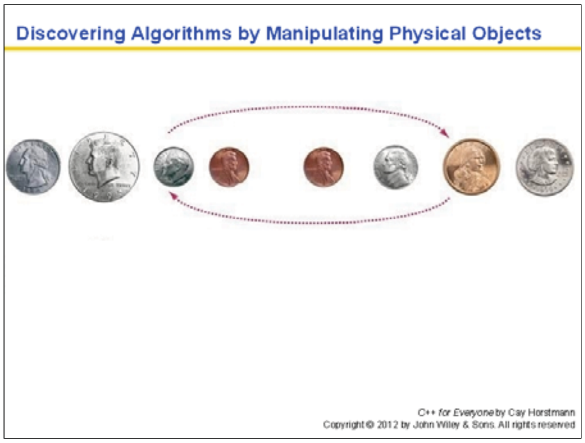
Discovering Algorithms by Manipulating Physical Objects

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



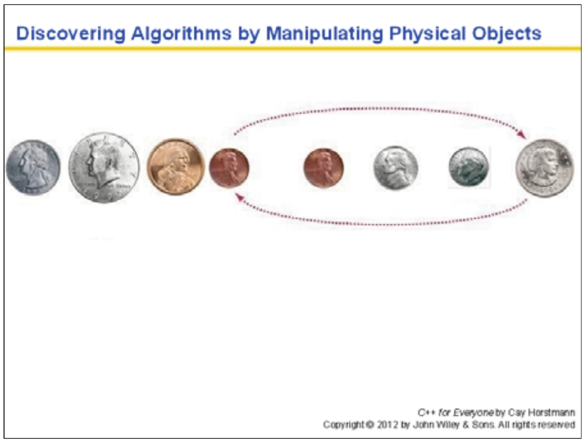
Discovering Algorithms by Manipulating Physical Objects

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



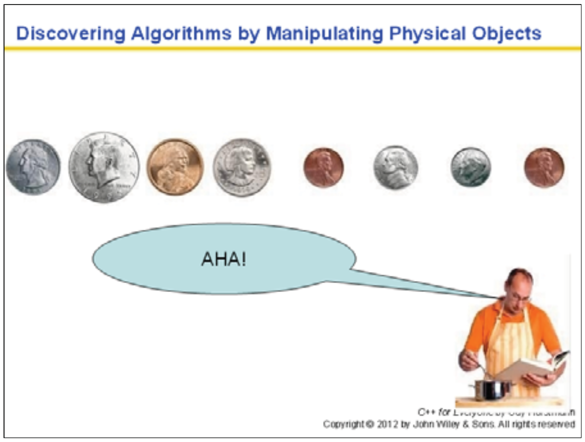
Discovering Algorithms by Manipulating Physical Objects

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



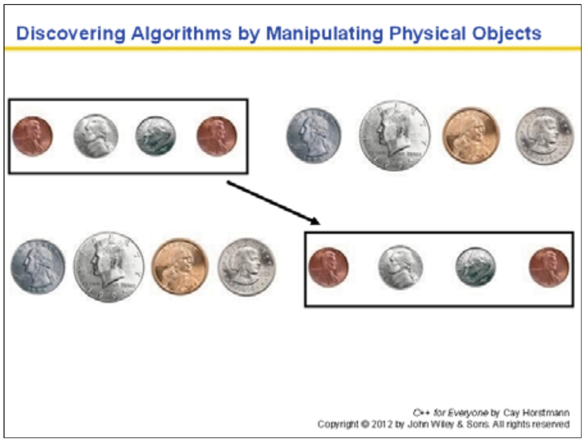
Discovering Algorithms by Manipulating Physical Objects

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



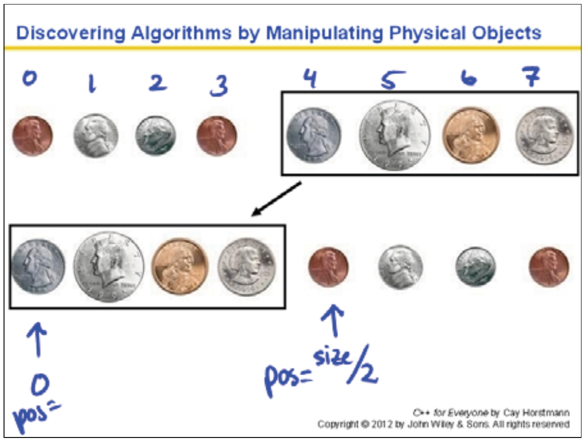
Discovering Algorithms by Manipulating Physical Objects

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



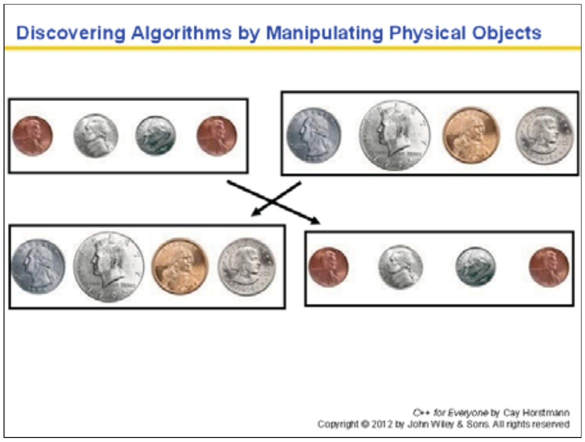
Discovering Algorithms by Manipulating Physical Objects

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



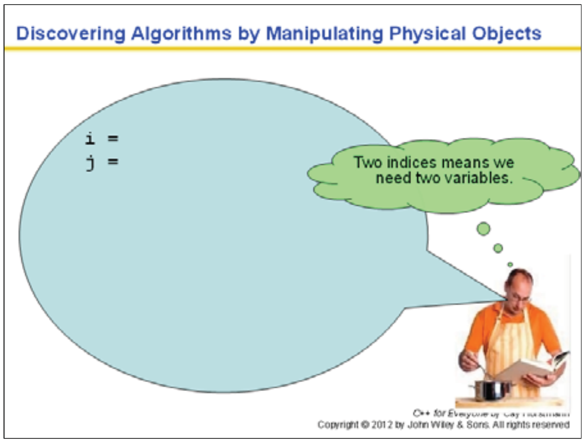
Discovering Algorithms by Manipulating Physical Objects

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



Discovering Algorithms by Manipulating Physical Objects

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved

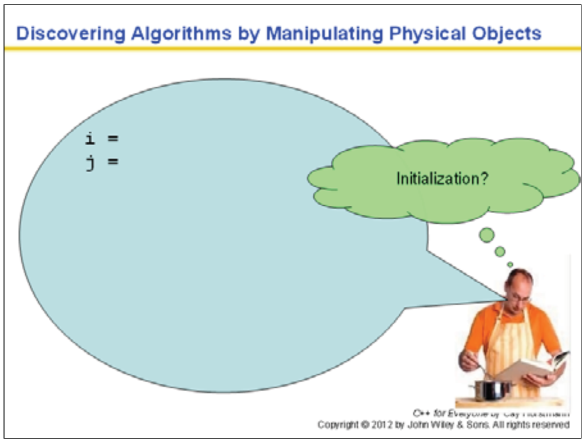


Discovering Algorithms by Manipulating Physical Objects

```
i =
j =
```

Two indices means we need two variables.

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved

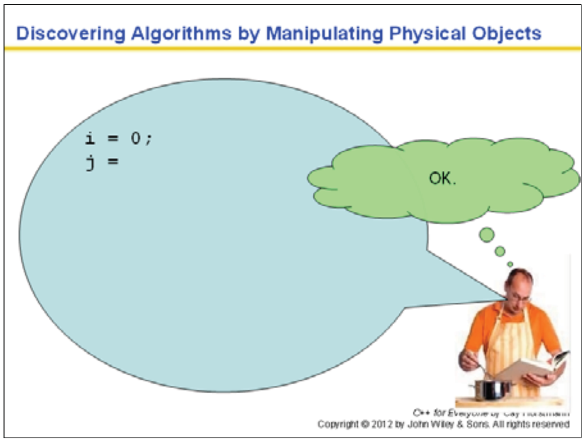


Discovering Algorithms by Manipulating Physical Objects

```
i =
j =
```

Initialization?

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved

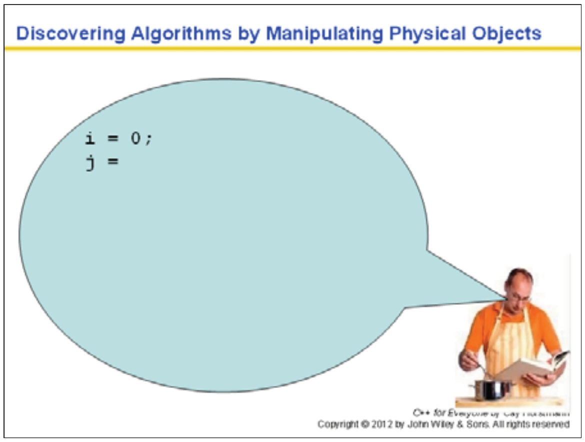


Discovering Algorithms by Manipulating Physical Objects

```
i = 0;
j =
```

OK.

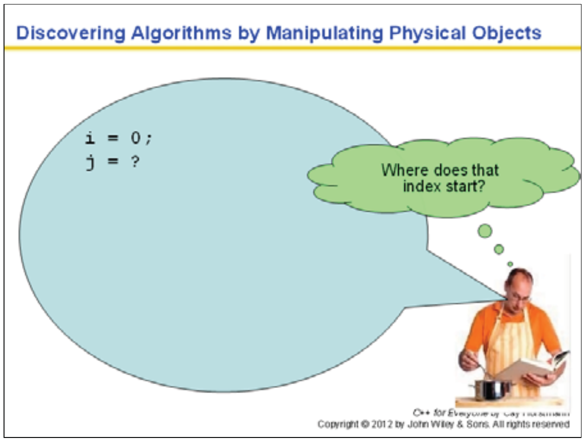
C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



Discovering Algorithms by Manipulating Physical Objects

```
i = 0;
j =
```

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved

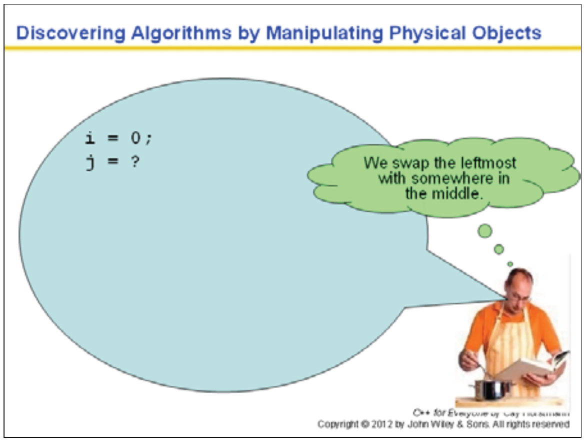


Discovering Algorithms by Manipulating Physical Objects

```
i = 0;
j = ?
```

Where does that index start?

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved

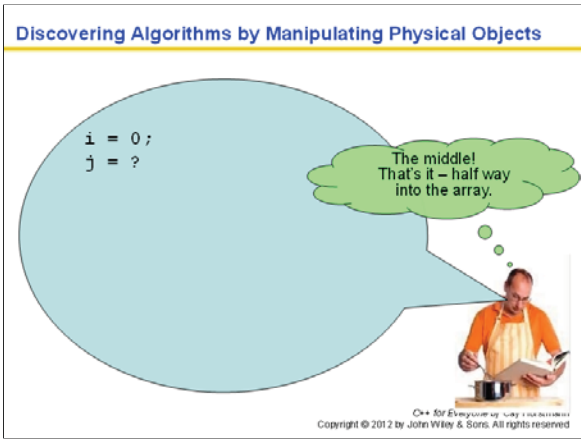


Discovering Algorithms by Manipulating Physical Objects

```
i = 0;
j = ?
```

We swap the leftmost with somewhere in the middle.

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



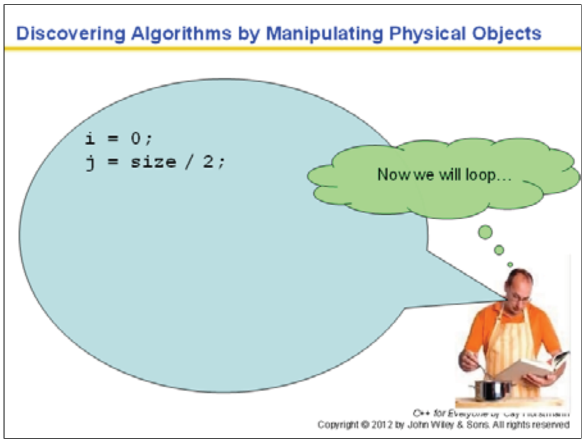
Discovering Algorithms by Manipulating Physical Objects

```
i = 0;
j = ?
```

The middle!
That's it – half way
into the array.

C++ for Everyone by Cay Horstmann

Copyright © 2012 by John Wiley & Sons. All rights reserved



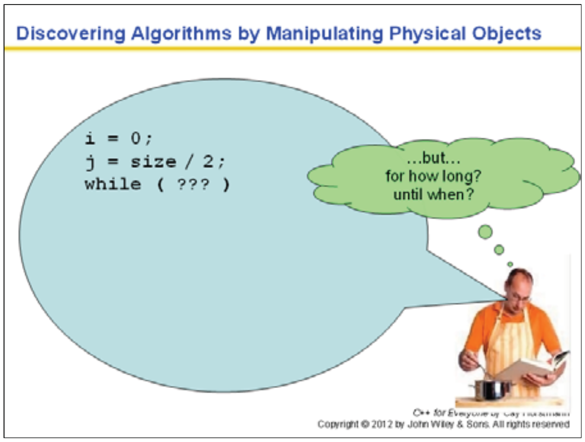
Discovering Algorithms by Manipulating Physical Objects

```
i = 0;
j = size / 2;
```

Now we will loop…

C++ for Everyone by Cay Horstmann

Copyright © 2012 by John Wiley & Sons. All rights reserved



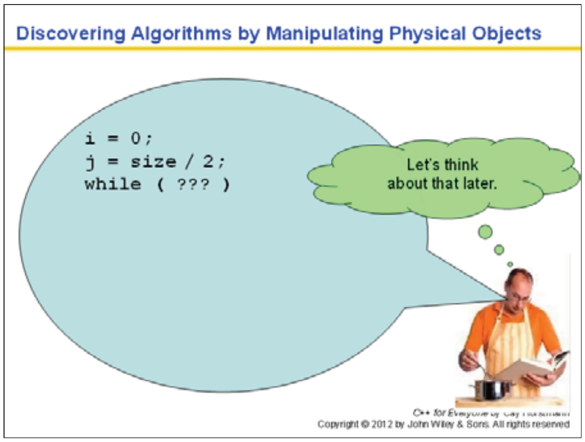
Discovering Algorithms by Manipulating Physical Objects

```
i = 0;
j = size / 2;
while ( ??? )
```

…but…
for how long?
until when?

C++ for Everyone by Cay Horstmann

Copyright © 2012 by John Wiley & Sons. All rights reserved



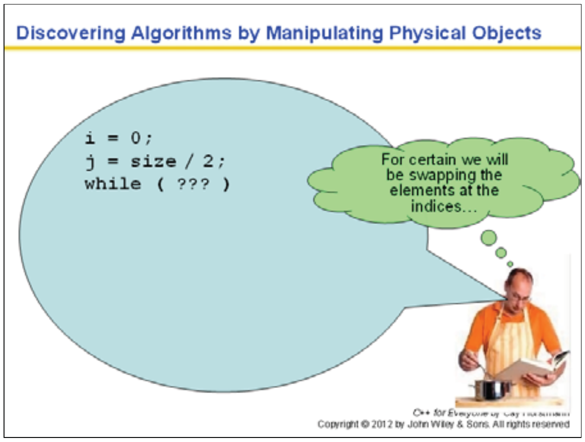
Discovering Algorithms by Manipulating Physical Objects

```
i = 0;
j = size / 2;
while ( ??? )
```

Let's think
about that later.

C++ for Everyone by Cay Horstmann

Copyright © 2012 by John Wiley & Sons. All rights reserved



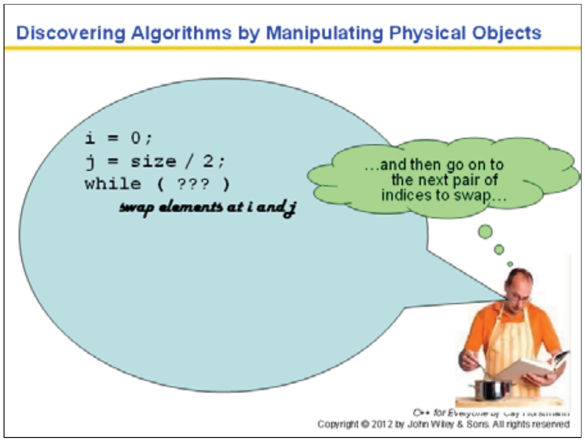
Discovering Algorithms by Manipulating Physical Objects

```
i = 0;
j = size / 2;
while ( ??? )
```

For certain we will
be swapping the
elements at the
indices…

C++ for Everyone by Cay Horstmann

Copyright © 2012 by John Wiley & Sons. All rights reserved



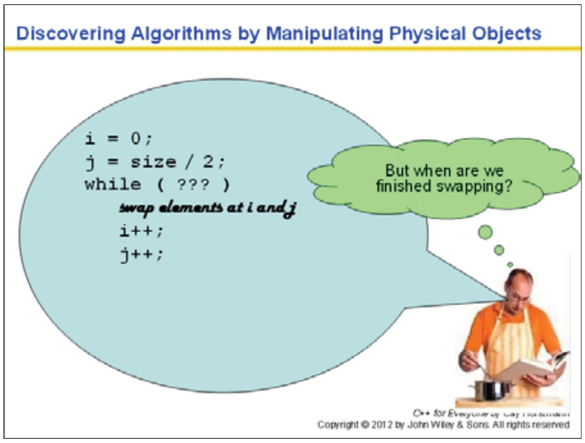
Discovering Algorithms by Manipulating Physical Objects

```
i = 0;
j = size / 2;
while ( ??? )
    swap elements at i and j
```

…and then go on to
the next pair of
indices to swap…

C++ for Everyone by Cay Horstmann

Copyright © 2012 by John Wiley & Sons. All rights reserved

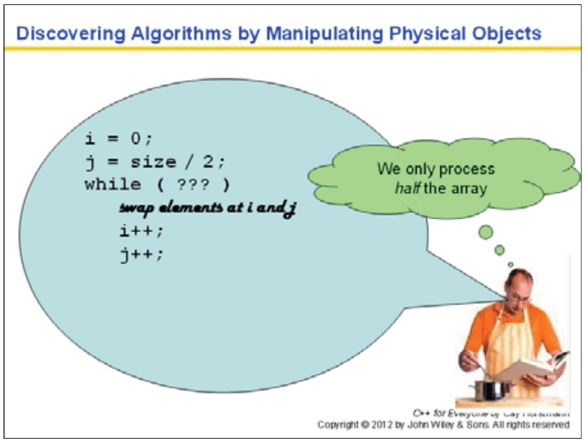


Discovering Algorithms by Manipulating Physical Objects

```
i = 0;
j = size / 2;
while ( ??? )
    swap elements at i and j
    i++;
    j++;
```

But when are we finished swapping?

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved

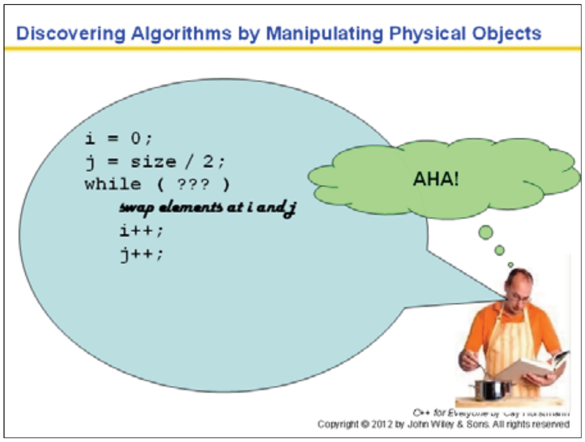


Discovering Algorithms by Manipulating Physical Objects

```
i = 0;
j = size / 2;
while ( ??? )
    swap elements at i and j
    i++;
    j++;
```

We only process *half* the array

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved

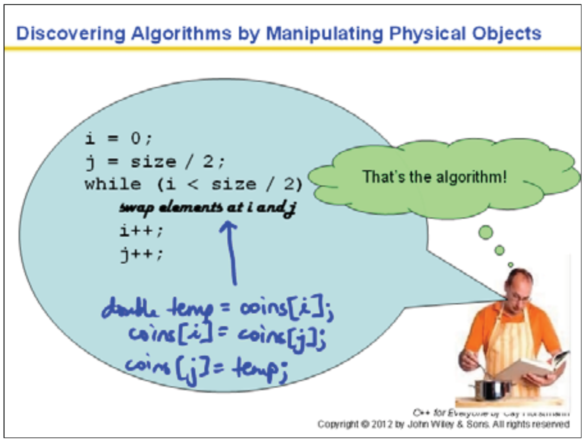


Discovering Algorithms by Manipulating Physical Objects

```
i = 0;
j = size / 2;
while ( ??? )
    swap elements at i and j
    i++;
    j++;
```

AHA!

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



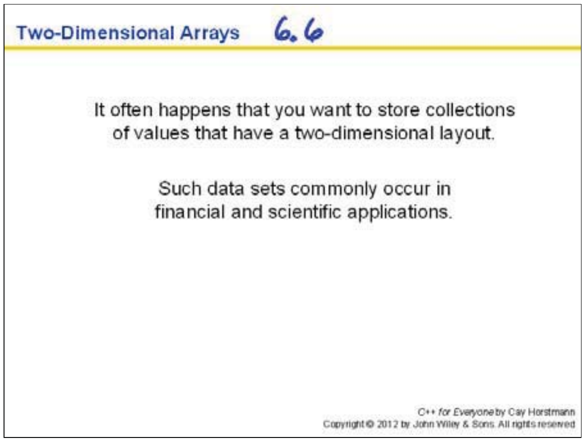
Discovering Algorithms by Manipulating Physical Objects

```
i = 0;
j = size / 2;
while (i < size / 2)
    swap elements at i and j
    i++;
    j++;
```

That's the algorithm!

```
double temp = coins[i];
  coins[i] = coins[j];
  coins[j] = temp;
```

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved

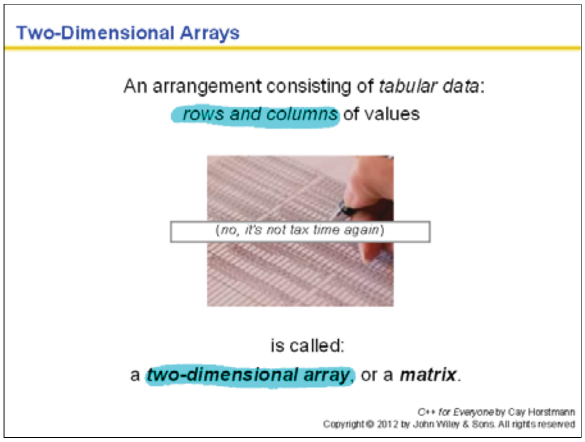


Two-Dimensional Arrays 6.6

It often happens that you want to store collections of values that have a two-dimensional layout.

Such data sets commonly occur in financial and scientific applications.

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved



Two-Dimensional Arrays

An arrangement consisting of *tabular data*:
rows and columns of values

(no, it's not tax time again)

is called:
a ***two-dimensional array***, or a ***matrix***.

C++ for Everyone by Cay Horstmann
Copyright © 2012 by John Wiley & Sons. All rights reserved