Returning a Pointer

The `biggest()` function returns a pointer to the largest item in the array. We don't want to allow the element to change, and we don't want the pointer to be used to modify other elements, so the return type is `const double* const`.

When you **call `biggest()`**, you will **dereference** the returned pointer to get the value.

```
cout << *(biggest(da, 5)) << endl;
```

Let's **apply the steps** in the extreme values algorithm to this problem.

1. Save the first value as the largest. You need two variables to do this:

   ```
   const double *p = a;
   double largest = *p;
   ```

2. Now, loop through each **remaining element** like this:

   ```
   for (size_t i = 1; i < len; ++i)...
   ```

3. Each time through the loop, check to see if the current element is larger than the saved value, and, if so, update the saved values. Because you want to return a pointer, you'll need to update **both** `largest` and `p`. Note the use of the address operator.

   ```
   if (a[i] > largest) {
       p = &a[i];
       largest = a[i];
   }
   ```

4. Finally, simply return the pointer `p`.

This is the same scheme used by the standard library algorithms `min_element()` and `max_element()`. When called using arrays, they return a pointer in exactly this manner.