1. Fix the following code. Change only what’s actually broken. Ignore poor style.

```cpp
#include <iostream>
using std::cout;
using std::endl;

int main ()
{
    int len = 5;
    char name[len] = "Chris";
    cout << "Hello, " << name
        << ", or should I say, " << "olleh" << ".";
    printReversed( name, len );
    cout << "?" << endl;
    return 0;
}

void printReversed( char name[], int n )
{
    for ( int i = n; i > 0; i - 1 )
    {
        cout << name[ i ];
    }
}
```

4 points – half a point for each thing found, minus half a point for each incorrect correction.

Most people missed the missing prototype. Quite a few missed the incorrect array length (a classic C/C++ bug). Many people deleted the embedded string quotes, but that’s not fixing the code, that’s changing what it does. Many of those who did fix the quotes, used / instead of \, which I allowed, though it’s wrong. Two people noticed that “hello” was not correctly reversed (oops).
2. Define \( \text{bool inArray( int k, const int a[], int n )} \) to return true if \( k \) is equal to at least one of \( a[0] \ldots a[n-1] \).

\[
\begin{align*}
\text{bool inArray( int m, const int a[], int n )} \\
\text{\{} \\
\text{\quad for ( int i = 0; i < n; ++i )} \\
\text{\quad \quad if ( m == a[ i ] )} \\
\text{\quad \quad \quad return true; } \\
\text{\quad } \\
\text{\quad return false; } \\
\text{\}}
\end{align*}
\]

3 points. Half a point off for each mistake.

A number of people returned 1 and 0. 1 and 0 have to be converted to true and false, so this is less clear and less efficient than using true and false directly. No points taken off for that.

3. Using \( \text{inArray} \), define \( \text{int copyUnique( const int a[], int b[], int size )} \) to copy into \( b[] \) each unique element of \( a[0] \ldots a[size-1] \), and return the number of unique elements copied. Example: if \( a[] = \{ 2, 3, 2, 3, 4, 2, 5, 4 \} \), it would copy 2, 3, 4, and 5 into \( b[] \) and return 4.

\[
\begin{align*}
\text{int copyUnique( const int a[], int b[], int size )} \\
\text{\{} \\
\text{\quad int k = 0; } \\
\text{\quad for ( int i = 0; i < size; ++i )} \\
\text{\quad \quad if ( !\text{inArray( a[ i ], b, k )} )} \\
\text{\quad \quad \quad b[ k++ ] = a[ i ]; } \\
\text{\quad \quad \quad return k; } \\
\text{\quad } \\
\text{\quad return k; } \\
\text{\}}
\end{align*}
\]

4 points. Half a point off for each mistake.

The most common error was passing size instead of \( k \) to \( \text{inArray()} \). \( b[] \) only has \( k \) elements. Another common error was \( b[i] = a[i] \) instead of \( b[k] = a[i] \). Also common was passing \( a \) instead of \( b \) to \( \text{inArray()} \). That will work if you pass \( i-1 \) instead of \( i \), but it’s less efficient and no one did that anyway. Some people wrote “if \( \text{inArray} \) do nothing else add \( a[i] \) to \( b \)” or the equivalent, which is more roundabout than “if not \( \text{inArray} \) add \( a[i] \) to \( b \).” No points were taken off for this.