

Built-in arrays

Solutions

- Describe what is meant by a built-in array in C++
 - Built-in arrays are similar to `std::array`, but they do not keep a record of the element count
 - This must be managed by the programmer
- Give one example of how built-in array differs from `std::vector`
 - Number of elements is fixed and must be known in advance

- Convert the code below into a full working program. Experiment with printing and assigning different elements until you are confident with it

```
int a[] = {1, 2, 3, 4, 5};  
cout << a[0];  
cout << a[3];  
a[2] = 6;  
for (int i = 0; i < 5; ++i)  
    cout << a[i] << ", ";
```

- What happens if you use `auto` instead of `int` in the array definition?

- What happens if you use `auto` instead of `int` in the array definition?
 - `auto a[] = {1, 2, 3, 4, 5};`
 - Does not compile
 - Interpreted as “array of `auto`”, not “array of (deduced type)”

- Are the following statements legal C++? If they are legal, what will happen when they are executed?
 - Same answer as for `std::array`
 - All are legal, but out-of-range accesses have undefined behaviour

```
int a[] = {1, 2, 3, 4, 5};  
cout << a[-2];  
a[20] = 0;  
int c[5];  
cout << c[3];
```

- Write a program to show that printing the name of a built-in array gives the same result as printing the address of its first element

```
int a[] = {1, 2, 3, 4, 5};  
cout << "a = " << a << endl;  
cout << "&a[0] = " << &a[0] << endl; // Both print a value like 0x61fe00
```

- Write a program which tries to assign one array to another

```
int b[] = a; // Neither of these statements compiles  
int b[5] = a;
```

- Give some advantages of built-in arrays compared to `std::vector`
 - More efficient (as with `std::array`)
- Give an advantage of built-in arrays compared to `std::array`
 - Number of elements does not need to be known in advance
 - Very large arrays can be allocated on the heap
 - We can pass arrays of any size to a function (with `std::array`, the number of elements must match)
 - Compatible with C and old C++ code

- Give some disadvantages of arrays compared to `std::vector`
 - Less flexible than `std::vector`
 - If allocated on heap, the programmer needs to manage the memory used manually
- Give some disadvantages of arrays compared to `std::array`
 - Programmer needs to keep track of number of elements