

# Array Selection

Array selection uses the **subscript operator** as in Java. The result of selection expression is an *lvalue*, which means that you can assign new values to it, like this:

```
const int kLen = 6;
int a[kLen];
for (size_t i = 0; i < kLen; ++i) {
    a[i] = 10 * i;
}
```

0	1	2	3	4	5
0	10	20	30	40	50

**C++ performs no bounds checking** on array selection. Unlike **vector**, there is no **safe, range-checked alternative**, such as **at()**. If the array index is out of range, your compiler decides where the element **would be** in memory, and performs the requested operation, leading to **undefined results**.

Worse still, if you assign a value to that index position, you **can overwrite** the contents of memory used by some other part of the program. Writing beyond the end of an array is one of the primary vulnerabilities used to attack computer systems.



This course content is offered under a CC Attribution Non-Commercial license. Content in this course can be considered under this license unless otherwise noted.