Cheatsheets / **Learn C++**

Vectors

Index

An index refers to an element's position within an ordered list, like a vector or an array. The first element has an index of 0.
A specific element in a vector or an array can be accessed using its index, like `name[index]` .

```
std::vector<double> order = {3.99, 12.99, 2.49};

// What's the first element?
std::cout << order[0];

// What's the last element?
std::cout << order[2];
```

Vectors

In C++, a vector is a dynamic list of items, that can shrink and grow in size. It is created using `std::vector<type> name;` and it can only store values of the same type.
To use vectors, it is necessary to `#include` the `vector` library.

```
#include <iostream>
#include <vector>

int main() {

  std::vector<int> grades(3);

  grades[0] = 90;
  grades[1] = 86;
  grades[2] = 98;

}
```

`.push_back()` & `.pop_back()`

The following functions can be used to add and remove an element in a vector:

- `.push_back()` to add an element to the "end" of a vector
- `.pop_back()` to remove an element from the "end" of a vector

```
std::vector<std::string> wishlist;

wishlist.push_back("Oculus");
wishlist.push_back("Telecaster");

wishlist.pop_back();

std::cout << wishlist.size();
// Prints: 1
```

Vector Type

During the creation of a C++ vector, the data type of its elements must be specified. Once the vector is created, the type cannot be changed.

`.size()` Function

The `.size()` function can be used to return the number of elements in a vector, like `name.size()`.

```
std::vector<std::string> employees;

employees.push_back("michael");
employees.push_back("jim");
employees.push_back("pam");
employees.push_back("dwight");

std::cout << employees.size();
// Prints: 4
```

Print Share