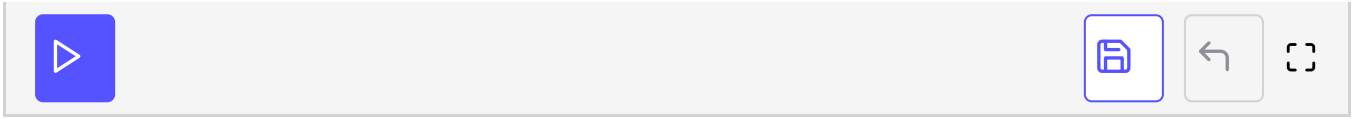
Assign and Swap

This lesson deals with ways to update and swap values in containers.

You can assign new elements to existing containers or swap two containers. For the assignment of a container `cont2` to a container `cont`, there exists the copy assignment `cont= cont2` and the move assignment `cont= std::move(cont2)`.This statement copies the value of cont2 to cont and the value of cont2 becomes empty after the move operation.A special form of assignment is the one with an initializer list: `cont= {1, 2, 3, 4, 5}`. That's not possible for `std::array`, but you can instead use the aggregate initialization. The function `swap` exists in two forms. You have it as a method `cont(swap(cont2))` or as a function template `std::swap(cont, cont2)`.

```
// containerAssignmentAndSwap.cpp
#include <iostream>
#include <set>

int main(){
  std::set<int> set1{0, 1, 2, 3, 4, 5};
  std::set<int> set2{6, 7, 8, 9};

  for (auto s: set1) std::cout << s << " "; // 0 1 2 3 4 5
  std::cout << "\n";
  for (auto s: set2) std::cout << s << " "; // 6 7 8 9
  std::cout << "\n";

  set1= set2;
  for (auto s: set1) std::cout << s << " "; // 6 7 8 9
  std::cout << "\n";
  for (auto s: set2) std::cout << s << " "; // 6 7 8 9
  std::cout << "\n";

  set1= std::move(set2);//moves value of set2 in set1 and set2 becomes empty
  for (auto s: set1) std::cout << s << " "; // 6 7 8 9
  std::cout << "\n";
  for (auto s: set2) std::cout << s << " ";//prints null since set2 becomes empty after the m

  set2= {60, 70, 80, 90};
  for (auto s: set1) std::cout << s << " "; // 6 7 8 9
  std::cout << "\n";
  for (auto s: set2) std::cout << s << " "; // 60 70 80 90
  std::cout << "\n";

  std::swap(set1, set2);
```

```
  for (auto s: set1) std::cout << s << " "; // 60 70 80 90
  std::cout << "\n";
  for (auto s: set2) std::cout << s << " "; // 6 7 8 9

  std::cout << "\n";
  return 0;
}
```



Assignment and swap