Improvements for Maps and Sets

In the next few lessons, we will test the functionality of the new features introduced for maps and sets.

WE'LL COVER THE FOLLOWING

- Splicing

In the Standard there are two notable features for maps and sets:

- Splicing Maps and Sets - P0083
- New emplacement routines - N4279

Splicing

You can now move nodes from one tree-based container (maps/sets) into other ones, without additional memory overhead/allocation.

For example:

```
#include <set>
#include <string>
#include <iostream>
struct User {
  std::string name;
  User(std::string s) : name(std::move(s))
  { std::cout << "User::User(" << name << ")\n";
  }
  ~User() {
    std::cout << "User::~User(" << name << ")\n";
  }
  User(const User& u) : name(u.name) {
    std::cout << "User::User(copy, " << name << ")\n";
  }
  friend bool operator<(const User& u1, const User& u2)
  {
    return u1.name < u2.name;
  }
};
int main()
{
  std::set<User> setNames;
  setNames.emplace("John");
  setNames.emplace("John");
  setNames.emplace("Alex");
  setNames.emplace("Bartek");
  std::set<User> outSet;

  std::cout << "move John...\n";
  // move John to the outSet
  auto handle = setNames.extract(User("John"));
  outSet.insert(std::move(handle));

  for (auto& elem : setNames)
    std::cout << elem.name << '\n';

  std::cout << "cleanup...\n";
}
```

In the above example, one element “John” is extracted from `setNames` into `outSet`. The extract method moves the found node out of the set and physically detaches it from the container. Later the extracted node can be inserted into a container of the same type.

Before C++17, if you wanted to move an object from one map (or set) and put it into another map (or set), you had to remove it from the first container and then copy/move into another. With the new functionality, you can manipulate tree nodes (by using implementation-defined type `node_type)` and leave objects unaffected. Such a technique allows you to handle non-movable elements and of course, is more efficient.

Improvements have also been made to the emplacement routines. We’ll discuss those next.