Minimum and Maximum

Let's take a look at the functions C++ provides to check the minimum and maximum in a range.

We can determine the minimum and maximum elements of the range with `std::min_element` and `std::max_element`. If we want to find both the minimum and maximum pair of a range then we can use the algorithm, `std::minmax_element`. Each algorithm can be configured with a binary predicate.

`std::min_element`: Returns the minimum element of the range.

```
constexpr FwdIt min_element(FwdIt first, FwdIt last)
FwdIt min_element(ExePol pol, FwdIt first, FwdIt last)

constexpr FwdIt min_element(FwdIt first, FwdIt last, BinPre pre)
FwdIt min_element(ExePol pol, FwdIt first, FwdIt last, BinPre pre)
```

`std::max_element`: Returns the maximum element of the range.

```
constexpr FwdIt max_element(FwdIt first, FwdIt last)
FwdIt max_element(ExePol pol, FwdIt first, FwdIt last)

constexpr FwdIt max_element(FwdIt first, FwdIt last, BinPre pre)
FwdIt max_element(ExePol pol, FwdIt first, FwdIt last, BinPre pre)
```

`std::minmax_element`: Returns the pair `std::min_element` and `std::max_element` of the range.

```
constexpr pair<FwdIt, FwdIt> minmax_element(FwdIt first, FwdIt last)
pair<FwdIt, FwdIt> minmax_element(ExePol pol, FwdIt first, FwdIt last)

constexpr pair<FwdIt, FwdIt> minmax_element(FwdIt first, FwdIt last, BinPre pre)
pair<FwdIt, FwdIt> minmax_element(ExePol pol, FwdIt first, FwdIt last, BinPre pre)
```

If the range has more than one minimum or maximum element, the first one is returned.

```
#include <algorithm>
#include <cstdlib>
#include <iostream>
#include <string>
#include <vector>
#include <sstream>

std::string toString(int i){
  std::stringstream buff;
  buff.str("");
  buff << i;
  std::string val= buff.str();
  return val;
}

int toInt(const std::string& s){
  std::stringstream buff;
  buff.str("");
  buff << s;
  int value;
  buff >> value;
  return value;
}


int main(){

  std::cout << std::endl;

  std::vector<int> myInts;
  std::vector<std::string> myStrings{"94", "5", "39", "-4", "-49", "1001", "-77", "23", "0",
  std::transform(myStrings.begin(), myStrings.end(), std::back_inserter(myInts), toInt);

  for (auto i: myInts) std::cout << i << " ";

  std::cout << "\n\n";

  auto paInt= std::minmax_element(myInts.begin(), myInts.end());
  std::cout << "std::minmax_element(myInts.begin(), myInts.end(): " << "(" << *paInt.first <<

  auto paStr= std::minmax_element(myStrings.begin(), myStrings.end());
  std::cout << "std::minmax_element(myStrings.begin(), myStrings.end(): " << "(" << *paStr.fi

  auto paStrAsInt= std::minmax_element(myStrings.begin(), myStrings.end(), [](std::string a,
  std::cout << "std::minmax_element(myStrings.begin(), myStrings.end(): " << "(" << *paStr.fi

  std::cout << std::endl;

}
```

Minimum and maximum algorithms

In the next lesson, we'll discuss different permutations in a range.