Transform Ranges

Now we will study std::transform which is used to perform transformations on a range.

The `std::transform` algorithm applies a unary or binary callable to a range and copies the modified elements to the destination range.

Applies the unary callable `fun` to the elements of the input range and copies the result to `result`:

```
OutIt transform(InpIt first1, InpIt last1, OutIt result, UnFun fun)
FwdIt2 transform(ExePol pol, FwdIt first1, FwdIt last1, FwdIt2 result, UnFun fun)
```

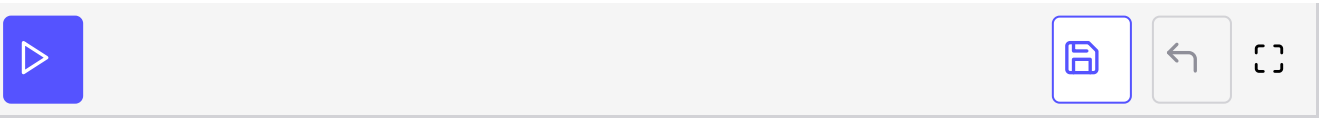
Applies the binary callable `fun` to both input ranges and copies the result to `result`:

```
OutIt transform(InpIt1 first1, InpIt1 last1, InpIt2 first2, OutIt result, BiFun fun)
FwdIt3 transform(ExePol pol, FwdIt1 first1, FwdIt1 last1, FwdIt2 first2, FwdIt3 result, BiFur
```

The difference between the two versions is that the first version applies the callable to each element of the range; the second version applies the callable to pairs of both ranges in parallel. The returned iterator points to one position after the last transformed element.

```
#include <algorithm>
#include <cctype>
#include <iostream>
#include <string>
#include <vector>

int main(){

  std::cout << std::endl;

  std::string str{"abcdefghijklmnopqrstuvwxyz"};

  std::cout << str << std::endl;

  std::transform(str.begin(), str.end(), str.begin(), [](char c){ return std::toupper(c); });
```

```
  std::cout << str << std::endl;

  std::cout << std::endl;

  std::vector<std::string> vecStr{"Only", "for", "testing", "purpose", "."};
  std::vector<std::string> vecStr2(5, "-");

  std::vector<std::string> vecRes;

  std::transform(vecStr.begin(), vecStr.end(),
         vecStr2.begin(),
         std::back_inserter(vecRes),
         [](std::string a, std::string b){ return std::string(b) + a + b; });

  for ( auto str: vecRes ) std::cout << str << std::endl;

  std::cout << std::endl;

}
```



Transform algorithms