

Problem 1: In `vecfunc.cpp`, implement the following functions.

```
bool isfactor(int n, const vector<int> & vec)
int myMinimum(const vector<int> & vec)
void myPermutation(vector<int> & vec)
void mySimplify(vector<int> & vec)
```

where

- (20pt) `isfactor` returns a bool type indicating whether `n` is a common factor of all the elements in `vec`, in other words, all the elements in `vec` are multiples of `n`. If so, return true, otherwise return false.
- (20pt) `myMinimum` returns the minimum element in `vec`.
- (20pt) `myPermutation` applies a special permutation to the numbers: moving the last element to the beginning. For example, a vector 1 2 3 will become 3 1 2 after the function call.
- (20pt) `mySimplify` modifies the vector by dividing each element by their maximum common factor. For example, a vector 3 6 9 will become 1 2 3 after the function call.

Hint: the function `isfactor` and `myMinimum` may help you find the maximum common factor.

Requirements:

1. You should use the given template. Do not modify the code below the line in the template. Do not forget to declare the ownership in the beginning of `vecfunc.cpp`.
2. For simplicity, we suppose the user input integers are always positive, and of size at least 2. Therefore, in the above function parameters, `vec` is always a vector of positive integers with size at least 2. You don't need to worry about other cases.

Here are examples of sample runs (the second line is user input)

```
Please input a sequence of postive integers, ending with Q:
3 6 9 Q
min element: 3
12 3 6 9
4 1 2 3
```

or

```
Please input a sequence of postive integers, ending with Q:
2 2 Q
min element: 2
2 2
1 1
```

or

```
Please input a sequence of postive integers, ending with Q:
3 3 12 18 Q
min element: 3
18 3 3 12
6 1 1 4
```

Instructions:

- All code must be written originally by yourself. You are not allowed to (even partially) copy code from anyone else. Incident of cheating or plagiarism will be reported to the Dean's office and results in a zero grade in this assignment.
- (2.5pt) Fill in the template `vecfunc.cpp` without changing the file name, and only submit it to Gradescope.
- (5pt) Declare the ownership in the beginning of your file. See details in HW1.
- (80pt) Implement all functions correctly.
- (12.5pt) Write your code with good coding practices, including commenting your code, using descriptive variable names, using constant variables, etc.
- Code compiles with Visual Studio 2022 and solves the questions. Students may lose the majority of points if their code doesn't compile with VS 2022. To receive full credits, the output must look EXACTLY the same as instructed above, including words, spaces, symbols, etc. Your code should not only work for the above examples, but also work for other different inputs.