Our very own `Pic10b::vector` class

Winter 2018

Figure 1: *Manolo Sánchez* about to enter *La Plaza de San Ángel* (top). *Manolo 'calaca' Sánchez* arriving in *The Land of the Remembered* (bottom).

About the `.pdf` version of this document

If you are reading the online [markdown] version of this document, click here to access the location where the `.pdf` version is hosted; then click on the `view raw` link to download a copy of this document.[1]

The backstory

María Posada is 100% sure *Manolo* will find a way to come back to *San Ángel*. However, she has no clue of which *Manolo* will eventually show up. Will it be the original *Manolo*, or will it be the *'calaca'* version? She would like to know in advance so that she can make the appropriate preparations for his comeback party. She could take the easy way out, and buy two versions of every item needed for the party. For example, she could buy two cakes: one black and white in case *Manolo 'calaca'* returns; and one with multiple colors in case the original *Manolo* shows up.

Oh, what a dilemma! If she only knew about templates, her life would be much easier…

The assignment

You are to template the `Pic10b::vector` class that we have been studying during lecture. You can either

- start your code from scratch, or
- adapt this *non-templated* implementation, or
- base your work on the `MyVector` class available at the "Old code [deprecated]" section of this course CCLE website.

I suggest you start by making sure your implementation works for `int`, as well as `double` types. The drivers below should help you test your functions.

- `01-driver_int.cpp`: a driver for the class `Pic10b::vector<int>`.
- `02-driver_double.cpp`: a driver for the class `Pic10b::vector<double>`.

When implemented correctly, the drivers combined with your implementation should output something similar to the contents of the following files:

- `output_int.txt`: console output produced by `01_driver_int.cpp`.
- `output_double.txt`: console output produced by `02_driver_double.cpp`.

[1] The embedded link does not work in the `.pdf` document.

Notice that your program **is expected to display messages to the console every time one of *the big 4* functions is called**. The message should:

a) start with 3 or more lower case *'ex'* characters without spaces between them; and
b) indicate which one of *the big 4* functions is being called.

For example, if the destructor is called, valid messages include:

```
xxx: Destructor...
xxxxxxxx: This is the destructor :xxxxxxxx
```

Failure to follow this guidelines might result in points being deducted from your final score.

Once your program works as expected with numeric types, you should then specialize some of the functions so that they can handle strings. Assuming we have the following setup

```
using std::string;
...

Pic10b::vector<string> v;
v.push_back("entry_1");
v.push_back("entry_2");
v.push_back("entry_3");
```

the functions that need to be specialized, as well as details about their implementations are listed below:

i. `operator<<` acting on objects of type `std::ostream` and `Pic10b::vector<string>`.

   Unlike the overload that handles *numeric* data types, this function should replace the delimiting curly braces (`{,}`) with square brackets (`[,]`). In addition, blank spaces should be inserted to make the contents of the vector more readable.

```
cout << v ;
// displays: [ entry_1, entry_2, entry_3 ]
//
// note(s):
//     - there is a blank space after '[', and one before ']', and
//     - there is no trailing comma after entry_3.
//
//     For reference, numeric types are displayed {1, 4, 6, 3}.
```

ii. `operator*` acting on two objects of type `string` and `Pic10b::vector<string>`.

   This operator should return a `Pic10b::vector<string>` object whose k-th entry is the concatenation (*i.e.,* string addition) of the string parameter and the k-th entry of the

`Pic10b::vector<string>` parameter. This concatenation should add padding (blank spaces), and it should preserve the order in which the parameters were passed.

```
string a = "This is:";
string b = "this is!";
cout << a * v;
// displays: [ This is: entry_1, This is: entry_2, This is: entry_3 ]
cout << v * b;
// displays: [ entry_1 this is!, entry_2 this is!, entry_3 this is! ]
//
// note(s):
//     - there is a blank space between 'a' and 'v[k]', and
//     - there is a blank space between 'v[k]' and 'b'.
```

Your are free to decide whether to implement the operators above as member or non-member functions. The driver below should help you test your specialized templates.

- `03-driver_string.cpp`. A driver for the class `Pic10b::vector<std::string>`.

When implemented correctly, your program should then output something similar to the contents of `output_string.txt`.

What is this assignment about?

Template functions, template classes and specialized templates.

Technical note

When working with templates the usual three-file-layout (`library.h` + `library.cpp` + `driver.cpp`) produces a link error. You should instead use a two-file-layout (`library.h` + `driver.cpp`), where `library.h` contains the class interface, as well as the details of the class implementation.

Submission

Upload your completed file `pic10b_vector.h` (all lowercase) to CCLE. Notice that there is no need to upload any of the drivers, as I already have those files. *If your file is named differently, your homework* ***will not be graded****.*

Your code should contain useful comments as well as your name, the date, and a brief description of what the program does. The files uploaded to CCLE will be automatically

collected at the date and time listed in the assignment description (*"Grading summary"* table located at the bottom of the page).

Grading rubric

Category	Description	Points
Correctness	No errors when tested against numeric types.	4
`vector<int>`	The output resembles the provided file.	5
`vector<double>`	The output resembles the provided file.	5
`vector<std::string>`	The output resembles the provided file.	5
Coding style	The code is efficient and easy to follow.	4
Total		20 (max)

Note:

Even though the total points adds up to 22, the maximum score you could receive for this assignment is 20. I expect the specialization of the template functions to handle strings to be a little more complicated than the corresponding functions that handle numeric types. In principle, even if your assignment does not handle strings very well you could still get full credit for this assignment.

List of files

The following files are included with this assignment:

- Current directory:
  - `readme.md.pdf`: this file.
  - `readme.md`: this file's source code (a combination of `yaml` + `markdown` + `pandoc` + LaTeX).
  - `create_pdf.sh`: `pandoc` command needed to generate a `.pdf` file from `readme.md`. *You can safely ignore this file!*
- `src` directory:
  - `pic10b_vector_non_template.cpp`: our very own (non-template) vector [of `int`s].
- `drivers` directory:

  - `01-driver-int.cpp`: your submission will be tested against this very same file.
  - `02-driver-double.cpp`: your submission will also be tested against this file.
  - `03-driver-string.cpp`: your submission… blah, blah, blah.

- `samples` directory:

  - `output-int.txt`
  - `output-double.txt`
  - `output-string.txt`

  Compare the output of your project to the contents of these files. It is OK if the number of debuggin messages (*e.g.,* "`xxx: Constructor`") do not match.

- `pics` directory:

  - `Manolo-vivo.png`: *Manolo Sánchez*, son of Carlos *Sánchez*[2].
  - `Manolo-calaca.png` *Manolo 'calaca' Sánchez*. He is the result of a [double?] bite from a two-headed snake.

[2]Who in turn is son of *Luis "el super macho" Sánchez*. I believe he also has two brothers named *Jorge* and *Carmelo*, but do not quote me on this.