Pratices

MATH3013 Mathematical Modeling

October 18, 2022

Contents

The purpose is to provide a chance to students for practical applications of C++ language. All the practices are coming from the textbook.

It is encouraged to use as much knowledge introduced in class meetings and from the textbook as possible, to finish the practice. Please organize your source code clearly (better following the suggestion from the textbook). It is better to prepare a readme file to give the infromation on your developement environment and how to run the code.

1 Developing environment

Developing environment is not mandatory, you can select Windows, MacOS, Linux as you wish, but I strongly recommend you to select linux.

1.1 Windows

- Dev C++ 5.11. A tiny C++ IDE using the MinGW, which can only support C++11, but it can only use in Windows.
- Clion 2021.1.2. A powerful IDE from JetBrains helps you develop in C and C++ on Linux, macOS and Windows. You can use C++14 or C++17.
- vscode. Visual Studio Code is a code editor redefined and optimized for building and debugging modern web and cloud applications. Visual Studio Code is free, but Visual Studio Code is just an editor, you must install the g++ compiler.

1.2 Mac OS

- Clion 2021.1.2. A powerful IDE from JetBrains helps you develop in C and C++ on Linux, macOS and Windows. You can use C++14 or C++17.
- vscode. Visual Studio Code is a code editor redefined and optimized for building and debugging modern web and cloud applications. Visual Studio Code is free, but Visual Studio Code is just an editor, you must install the g++ compiler.
- Xcode. Xcode is Apple's integrated development environment (IDE) for macOS, it is huge.

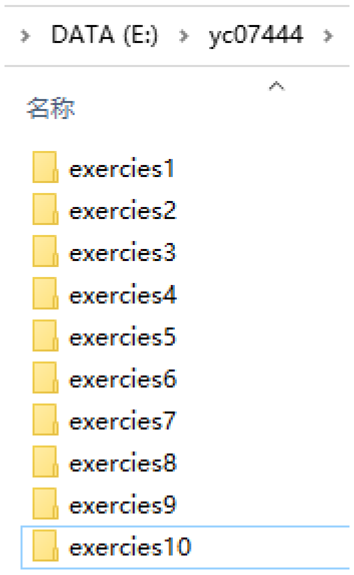
1.3 Linux

- Clion 2021.1.2. A powerful IDE from JetBrains helps you develop in C and C++ on Linux, macOS and Windows. You can use C++14 or C++17.
- vscode. Visual Studio Code is a code editor redefined and optimized for building and debugging modern web and cloud applications. Visual Studio Code is free, but Visual Studio Code is just an editor, you must install the g++ compiler.
- Emacs. An extensible, customizable, free/libre text editor.

2 How to orginize your files

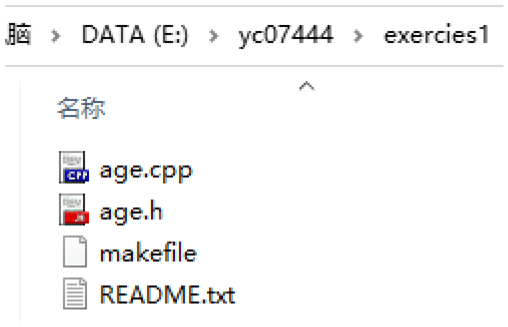
The top direction must be your student code, the second layer direction is the name of the exercise.

For example, my student code is **yc07444**, the exercises are **exercise1**, **exercise2**, ..., **exercise10**. The following picture shows the architecture of the whole direction.


> DATA (E:) > yc07444 >

名称

exercies1
exercies2
exercies3
exercies4
exercies5
exercies6
exercies7
exercies8
exercies9
exercies10


脑 > DATA (E:) > yc07444 > exercies1

名称

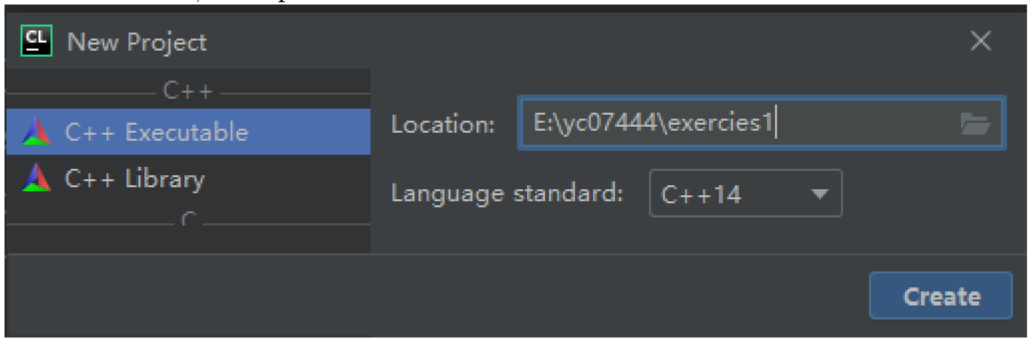
age.cpp
age.h
makefile
README.txt

3 CLion step by step

In this section, I will techer you how to create a project using CLion.

3.1 Create a project

From the menu "File" → "New Project", please select the correct direction which is described in 2, select the C++14 or C++17, then press the button "create".



Location: E:\yc07444\exercies1

Language standard: C++14

Now, the project for an exercise is done, you can enjoy your coding now.

4 Exercise 1: Age

4.1 Name of exercise

The project name is "exercise1", this project should in the direction "your student number/exercise1".

4.2 Requirements

Please refer to the textbook pp 63.

Write a program that asks input from the keyboard and prints the result on the screen and writes it to a file which name is age.txt in the same direction of this project. The question is: “What is your age?

5 Exercise 2: Polynomial

5.1 Name of exercise

The project name is "exercise2", this project should in the direction "your student number/exercise2".

5.2 Requirements

Please refer to the textbook pp 104.

Write a class for polynomials that should at least contain:

- A constructor giving the degree of the polynomial;
- A dynamic array/vector/list of double to store the coefficients;
- A destructor; and
- A output function for ostream.

Further members like arithmetic operations are optional.

6 Exercise 3: Move Assignment

6.1 Name of exercise

The project name is "exercise3", this project should in the direction "your student number/exercise3".

6.2 Requirements

Please refer to the textbook pp 104.

Write a move assignment operator for the polynomial in Exercise 2.8.1. Define the copy constructor as default. To test whether your assignment is used write a function polynomial f(double c2, double c1, double c0) that takes three coefficients and returns a polynomial. Print out a message in your move assignment or use a debugger to make sure your assignment is used.

7 Exercise 4: Initializer List

7.1 Name of exercise

The project name is "exercise4", this project should in the direction "your student number/exercise4".

7.2 Requirements

Please refer to the textbook pp 105.

Expand the program from Exercise 2 with a constructor and an assignment operator for a initializer list. The degree of the polynomial should be the length of the initializer list minus one afterward.

8 Exercise 5: Generic Stack

8.1 Name of exercise

The project name is "exercise5", this project should in the direction "your student number/exercise5".

8.2 Requirements

Please refer to the textbook pp 161.

Write a stack implementation for a generic value type. The maximal size of the stack is defined in the class (hard-wired). Provide the following functions:

- Constructor;
- Destructor if necessary;
- top: show last element;

- pop: remove last element (without returning);
- push: insert new element;
- clear: delete all entries;
- size: number of elements;
- full: whether stack is full;
- empty: whether stack is empty.

Stack over- or underflow must throw an exception.

9 Exercise 6: Iterator of a Vector

9.1 Name of exercise

The project name is "exercise6", this project should in the direction "your student number/exercise6".

9.2 Requirements

Please refer to the textbook pp 162.

Add the methods begin() and end() for returning a begin and end iterator to class vector. Add the types iterator and const_iterator to the class as well. Note that pointers are models of the concept of random-access iterators. Use the STL function sort for ordering vector entries to demonstrate that your iterators work as they should.

Stack over- or underflow must throw an exception.

10 Exercise 7: Odd Iterator

10.1 Name of exercise

The project name is "exercise7", this project should in the direction "your student number/exercise7".

10.2 Requirements

Please refer to the textbook pp 162.

Write an iterator class for odd numbers named odd_iterator. The class must model (realize) the ForwardIterator concept `http://www.sgi.com/tech/stl/ForwardIterator.html`. That means it must provide the following members:

- Default and copy constructor;
- operator++ to the next odd element, as pre-increment and post-increment;
- operator* as dereference which returns an (odd) int;
- operator== and operator!=; and
- operator=.

with the obvious semantics. In addition, the class should contain a constructor that accepts an int value. This value will be returned in the dereference operator (as long as the iterator is not incremented). This constructor should throw an exception if the value is even. Likewise the default constructor should initialize the internal value with 1 to provide a legal state.

11 Exercise 8: Sorting by Mangnitude

11.1 Name of exercise

The project name is "exercise8", this project should in the direction "your student number/exercise8".

11.2 Requirements

Please refer to the textbook pp 215.

Create a vector of double and initialize it with the values -9.3, -7.4, -3.8, -0.4, 1.3, 3.9, 5.4, 8.2. You can use an initializer list. Sort the values by magnitude.

12 Exercise 9: STL Container

12.1 Name of exercise

The project name is "exercise9", this project should in the direction "your student number/exercise9".

12.2 Requirements

Please refer to the textbook pp 216.

Create a std::map for phone numbers; i.e., map from a string to an unsigned long. Fill the map with at least four entries. Search for an existing and a non-existing name. Also search for an existing and a non-existing number.

13 Exercise 10: Inheritance Vector Class

13.1 Name of exercise

The project name is "exercise10", this project should in the direction "your student number/exercise10".

13.2 Requirements

Please refer to the textbook pp 320.

Revise the vector example from Chapter 2. Introduce the base class vector_expression for size and operator(). Make vector inherit from this base class. Then make a class ones that is a vector of all ones and also inherits from vector_expression.

14 Remark

- Please prepare "documentation + source code" package for the submission of the practice. In the documentation, please try your best to give the instruction on what your code can do, and how to use it.
- You are encourage to use *github*, *gitlab*, *bitbucket*, *gitee*, etc. to organize your project. If you use above mentioned online tool, you may just send the link of the repository to us for the submission of the project.
- You can also package your practice, and mail to TA **yc07444@um.edu.mo**. The title of email must be "pratices of *your student number*".