南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

C/C++ Program Design

Lab 4, Compound Types

廖琪梅, 王大兴, 于仕琪

Compound Types

- Array, C-style string, string, structure, enumeration
- Input and output
- cmake

Array

- Arrays are fixed-size collections consisting of data items of the same type.
- The index of an array is from 0.
- C++ compiler does not report whether the array is out-off-bounds.

Array

```
#include <iostream>
using std::cin;
using std::cout;
using std::endl;

int main() {
    int arr[10];
    arr[0] = 0;
    arr[1] = 1;

    // print the first element in arr
    cout<<arr[0]<<endl;

    // this is wrong:
    // cout<<arr[10]<<endl;

return 0;
}
```

wdx@DESKTOP-R133B5N: ~/Cpp

```
wdx@DESKTOP-R133B5N:~/Cpp$ g++ -o main main.cpp && ./main
0
wdx@DESKTOP-R133B5N:~/Cpp$
```

C-style string and string

- ***C-style string*** is an array of characters whose last character must be a null character denoted as \0.
- ***string*** is a class of C++, it can be used as a type.

C-style string and string

```
#include <iostream>
#include <string>
using namespace std;

int main() {
    // This is c-string
    char str1[100];

    cin>>str1;
    cout<<str1<<endl;

    // This is string
    string str2;

    cin>>str2;
    cout<<str2<<endl;

return 0;
}
```

wdx@DESKTOP-R133B5N: ~/Cpp

```
wdx@DESKTOP-R133B5N:~/Cpp$ g++ -o main main.cpp && ./main
123
123
456
456
wdx@DESKTOP-R133B5N:~/Cpp$
```

Structure

- ***Structure*** is a collection which can hold items of more than one data type.
- A structure is a user-definable type, first declare a structure and then define a variable of the structure.
- Use ***membership operator(.)*** to access the member of a structure.
- If the structure variables are from the same structure, assignment operation of them can be done by equal sign(=).
- Arrays of structures are the arrays whose elements are structures.

Structure

```
#include <iostream>
using namespace std;

struct Rectangle {
    int width;
    int height;
};

int main() {
    Rectangle rec;

    rec.width = 1;
    rec.height = 2;

    cout<<rec.width<<" "<<rec.height<<endl;

return 0;
}
```

wdx@DESKTOP-R133B5N: ~/Cpp

```
wdx@DESKTOP-R133B5N:~/Cpp$ g++ -o main main.cpp && ./main
1 2
wdx@DESKTOP-R133B5N:~/Cpp$
```

Enumeration

- If you want to model seven days in a week, you can define a enumeration.

```
#include <iostream>
using namespace std;

enum Days { SUNDAY, MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY };

int main() {

    Days today = MONDAY;
    Days tomorrow = TUESDAY;

    cout<<"Today is: "<<today<<endl;

return 0;
}
```

```
wdx@DESKTOP-R133B5N: ~/Cpp
wdx@DESKTOP-R133B5N:~/Cpp$ g++ -o main main.cpp && ./main
1 2
wdx@DESKTOP-R133B5N:~/Cpp$ g++ -o main main.cpp && ./main
Today is: 1
```

Keyboard input and terminal output of string

1. C: scanf & printf

   %d ----int

   %f ----float

   %c -----char

   %s -----string

```c
scanf_printf.c > ...
#include <stdio.h>

int main()
{
    char str[20];
    printf("Enter a string:\n");
    scanf("%s", str);
    printf("You entered: %s\n",str);

    return 0;

}
```

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/csourcecode/2021Spring/lab03/ExampleCode$ gcc scanf_printf.c
maydlee@LAPTOP-U1MO0N2F:/mnt/d/csourcecode/2021Spring/lab03/ExampleCode$ ls
a.out        cin_cout.cpp     getline_get.cpp  onedarray.cpp  pointer_array.cpp      scanf_p
address.cpp  get_getline.cpp  gets_puts.c      pointer.cpp    pointer_structure.cpp  string.
maydlee@LAPTOP-U1MO0N2F:/mnt/d/csourcecode/2021Spring/lab03/ExampleCode$ ./a.out
Enter a string:
Computer
You entered: Computer
maydlee@LAPTOP-U1MO0N2F:/mnt/d/csourcecode/2021Spring/lab03/ExampleCode$ ./a.out
Enter a string:
Computer Science
You entered: Computer
maydlee@LAPTOP-U1MO0N2F:/mnt/d/csourcecode/2021Spring/lab03/ExampleCode$
```

Why only Computer?

scanf uses whitespace—spaces, tabs, and newlines to separate a string.

2. C: gets & puts

```
fgets(str, 20, stdin);
```

```c
gets_puts.c > ...
#include <stdio.h>

int main()
{
    char str[20];
    printf("Enter a string:\n");
    gets(str);
    printf("You entered: ");
    puts(str);

    return 0;
}
```

There is a warning due to using gets(). You can use fgets() function instead.

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/csourcecode/2021Spring/lab03/ExampleCode$ gcc gets_puts.c
gets_puts.c: In function ‘main’:
gets_puts.c:7:2: warning: implicit declaration of function ‘gets’; did you mean ‘fgets’? [-W
    7 |  gets(str);
      |  ^~~~
      |  fgets
/usr/bin/ld: /tmp/ccudF3zf.o: in function `main':
gets_puts.c:(.text+0x34): warning: the `gets' function is dangerous and should not be used.
maydlee@LAPTOP-U1MO0N2F:/mnt/d/csourcecode/2021Spring/lab03/ExampleCode$ ./a.out
Enter a string:
Computer Science
You entered: Computer Science
maydlee@LAPTOP-U1MO0N2F:/mnt/d/csourcecode/2021Spring/lab03/ExampleCode$
```

Use gets to gain a sentence with a space. gets() stops reading input when it encounters a newline or End of file.

3. C++: cin & cout

cin_cout.cpp > ...

```cpp
#include <iostream>
using namespace std;

int main()
{
    char str[100];

    cout << "Enter a string:";
    cin >> str;
    cout << "You entered: " << str << endl;

    cout << "Enter an other string:";
    cin >> str;
    cout << "You entered: " << str << endl;

    return 0;
}
```

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/csourcecode/2021Spring/lab03/ExampleCode$ g++ cin_cout.cpp
maydlee@LAPTOP-U1MO0N2F:/mnt/d/csourcecode/2021Spring/lab03/ExampleCode$ ./a.out
Enter a string:C++
You entered: C++
Enter an other string:Programming is fun
You entered: Programming
maydlee@LAPTOP-U1MO0N2F:/mnt/d/csourcecode/2021Spring/lab03/ExampleCode$
```

The cin is to use whitespace-- spaces, tabs, and newlines to separate a string.

4. C++: cin.getline( ) & cin.get( )

getline_get.cpp > ...

```cpp
#include <iostream>
using namespace std;

int main()
{
    char str[20];

    cout << "Enter a string:";
    cin.getline(str, 20);
    cout << "You entered: " << str << endl;

    cout << "Enter an other string:";
    cin.get(str, 20);
    cout << "You entered: " << str << endl;

    return 0;
}
```

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/csourcecode/2021Spring/lab03/ExampleCode$ g++ getline_get.cpp
maydlee@LAPTOP-U1MO0N2F:/mnt/d/csourcecode/2021Spring/lab03/ExampleCode$ ./a.out
Enter a string:C and C++
You entered: C and C++
Enter an other string:Programming is fun.
You entered: Programming is fun.
```

4. C++: cin.getline( ) & cin.get( )

getline_get.cpp > ...

```cpp
#include <iostream>
using namespace std;

int main()
{
    char str[20];

    cout << "Enter a string:";
    cin.getline(str, 20);
    cout << "You entered: " << str << endl;

    cout << "Enter an other string:";
    cin.get(str, 20);
    cout << "You entered: " << str << endl;

    return 0;
}
```

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/csourcecode/2021Spring/lab03/ExampleCode$ ./a.out
Enter a string:C++ and c
You entered: C++ and c
Enter an other string:C programming is funning.
You entered: C programming is fu
```

If the length of input string is greater than 20, it can only store first 19 characters in str.

4. C++: cin.getline( ) & cin.get( )

get_getline.cpp > ...

```cpp
#include <iostream>
using namespace std;

int main()
{
    char str[20];

    cout << "Enter a string:";
    cin.get(str, 20);
    cout << "You entered: " << str << endl;

    cout << "Enter an other string:";
    cin.getline(str, 20);
    cout << "You entered: " << str << endl;

    return 0;
}
```

getline() and get() both read an entire input line—that is, up until a newline character. However, getline() discard the newline character, whereas get() leave it in the input queue.

Program runs without entering another string

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/csourcecode/2021Spring/lab03/ExampleCode$ g++ get_getline.cpp
maydlee@LAPTOP-U1MO0N2F:/mnt/d/csourcecode/2021Spring/lab03/ExampleCode$ ./a.out
Enter a string:C and C++
You entered: C and C++
Enter an other string:You entered:
```

get_getline.cpp > ...

```cpp
#include <iostream>
using namespace std;

int main()
{
    char str[20];

    cout << "Enter a string:";
    cin.get(str, 20);
    cout << "You entered: " << str << endl;

    cin.get();
    cout << "Enter an other string:";
    cin.getline(str, 20);
    cout << "You entered: " << str << endl;

    return 0;
}
```

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/csourcecode/2021Spring/lab03/ExampleCode$ g++ get_getline.cpp
maydlee@LAPTOP-U1MO0N2F:/mnt/d/csourcecode/2021Spring/lab03/ExampleCode$ ./a.out
Enter a string:C and C++
You entered: C and C++
Enter an other string:Programming is fun.
You entered: Programming is fun.
```

C++ string using string data type

string.cpp > ...

```cpp
#include <iostream>
using namespace std;

int main()
{
    string str;
    cout << "Enter a string:";
    getline(cin, str);
    cout << "You entered: " << str << endl;

    return 0;
}
```

getline() function takes the input stream as the first parameter which is cin and str as the location of the line to be stored.

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/csourcecode/2021Spring/lab03/ExampleCode$ g++ string.cpp
maydlee@LAPTOP-U1MO0N2F:/mnt/d/csourcecode/2021Spring/lab03/ExampleCode$ ./a.out
Enter a string:Computer Science
You entered: Computer Science
```

What is CMake?

CMake is an open-source, cross-platform family of tools designed to build, test and package software. **CMake** is used to control the software compilation process using simple platform and compiler independent configuration files, and generate native makefiles and workspaces that can be used in the compiler environment of your choice.

For more information https://cmake.org/

CMake needs **CMakeLists.txt** to run properly.

A CMakeLists.txt consists of **commands** , **comments** and **spaces**.

- The **commands** include command name, brackets and parameters , the parameters are separated by spaces. Commands are not case sensitive.
- **Comments** begins with '#'.

1. A single source file in a project

The most basic project is an executable built from source code files. For simple projects, a three-line **CMakeLists.txt** file is all that is required.

```
cmake_minimum_required(VERSION 3.16)

project(hello)

add_executable(hello main.cpp)
```

Specifies the minimum required version of CMake. Use **cmake --version** in Vscode terminal window to check the cmake version in your computer.

Defines the project name.

Adds the hello executable target which will be built from main.cpp.

The first parameter indicates the filename of executable file.

The second parameter indicates the source file.

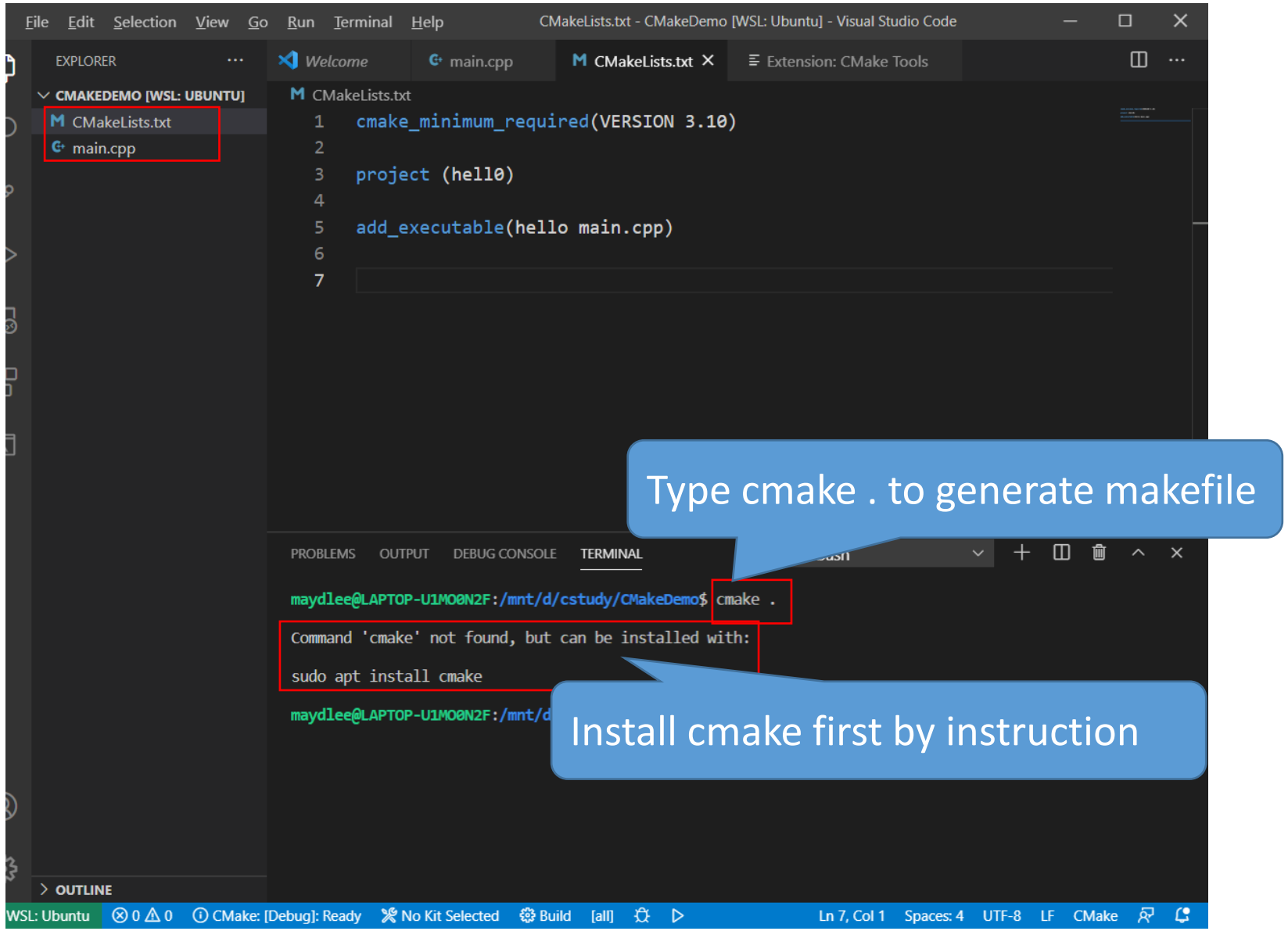
Suppose we have a main.cpp file

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Hello World!" << endl;

    return 0;
}
```

Store the CMakeLists.txt file in the same directory as the main.cpp.



Type cmake . to generate makefile

Install cmake first by instruction

File Edit Selection View Go Run Terminal Help CMakeLists.txt - CMakeDemo [WSL: Ubuntu] - Visual Studio Code

EXPLORER

CMAKEDEMO [WSL: UBUNTU]

- CMakeLists.txt
- main.cpp

Welcome | main.cpp | CMakeLists.txt | Extension: CMake Tools

CMakeLists.txt

```
cmake_minimum_required(VERSION 3.10)

project (hell0)

add_executable(hello main.cpp)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: sudo

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/cstudy/CMakeDemo$ sudo apt install cmake
[sudo] password for maydlee:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  cmake-data libjsoncpp1 librhash0
Suggested packages:
  cmake-doc ninja-build
The following NEW packages will be installed:
  cmake cmake-data libjsoncpp1 librhash0
0 upgraded, 4 newly installed, 0 to remove and 151 not upgraded.
Need to get 5470 kB of archives.
After this operation, 28.3 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
```

OUTLINE

WSL: Ubuntu | 0 0 | CMake: [Debug]: Ready | No Kit Selected | Build [all] | Ln 7, Col 1 | Spaces: 4 | UTF-8 | LF | CMake

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/cstudy/CMakeDemo$ cmake .
-- The C compiler identification is GNU 9.3.0
-- The CXX compiler identification is GNU 9.3.0#
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc --
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Configuring done
-- Generating done
-- Build files have been written to: /mnt/d/cstudy/
```

Run cmake to generate makefle, . indicates the makefile is stored in the current directory.

makefile file is created automatically after running cmake in the current directory.

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/cstudy/CMakeDemo$ ls
CMakeCache.txt  CMakeFiles  CMakeLists.txt  Makefile  cmake_install.cmake  main.cpp
```

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/cstudy/CMakeDemo$ make
Scanning dependencies of target hello
[ 50%] Building CXX object CMakeFiles/hello.dir/main.cpp.o
[100%] Linking CXX executable hello
[100%] Built target hello
```

Execute make to compile the program.

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/cstudy/CMakeDemo$ ./hello
Hello World!
```

Run the program

2. Multi-source files in a project

./CmakeDemo2

```
|
+--- main.cpp
|
+--- function.cpp
|
+--- function.h
```

There are three files in the same directory.

```
cmake_minimum_required(VERSION 3.10)

project(CmakeDemo2)

add_executable(CmakeDemo2 main.cpp function.cpp)
```

Add the function.cpp to the add_executable command.

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/cstudy/CMakeDemo2$ cmake .
-- The C compiler identification is GNU 9.3.0
-- The CXX compiler identification is GNU 9.3.0
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Configuring done
-- Generating done
-- Build files have been written to: /mnt/d/cstudy/CMakeDemo2
maydlee@LAPTOP-U1MO0N2F:/mnt/d/cstudy/CMakeDemo2$ make
Scanning dependencies of target CmakeDemo2
[ 33%] Building CXX object CMakeFiles/CmakeDemo2.dir/main.cpp.o
[ 66%] Building CXX object CMakeFiles/CmakeDemo2.dir/function.cpp.o
[100%] Linking CXX executable CmakeDemo2
[100%] Built target CmakeDemo2
```

2. Multi-source files in a project

If there are several files in directory, put each file into the add_executable command is not recommended. The better way is using **aux_source_directory** command.

aux_source_directory (<dir> <variable>)

The command finds all the source files in the specified directory indicated by <dir> and stores the results in the specified variable indicated by <variable>.

2. Multi-source files in a project

```
cmake_minimum_required(VERSION 3.10)

project(CmakeDemo2)

aux_source_directory(. DIR_SRCS)

add_executable(CmakeDemo2 ${DIR_SRCS})
```

Store all files in the current directory into DIR_SRCS.

Compile the source files in the variable by **${ }** into an executable file named CmakeDemo2

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/cstudy/CMakeDemo2$ cmake .
-- The C compiler identification is GNU 9.3.0
-- The CXX compiler identification is GNU 9.3.0
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Configuring done
-- Generating done
-- Build files have been written to: /mnt/d/cstudy/CMakeDemo2
```

3. Multi-source files in a project in different directories

```
./CMakeDemo3
    |
    +--- src/
    |     |
    |     +-- main.cpp
    |     +-- function.cpp
    |
    +--- include/
          |
          +--- function.h
```

We write CMakeLists.txt in CmakeDemo3 folder.

```cmake
# CMake minimum version
cmake_minimum_required(VERSION 3.10)

# project information
project(CMakeDemo3)

# Search the source files in the src directory
# and store them into the variable DIR_SRCS
aux_source_directory(./src DIR_SRCS)

# add the directory of include
include_directories(include)

# Specify the build target
add_executable(CMakeDemo3 ${DIR_SRCS})
```

All .cpp files are in the src directory

Include the header file which is stored in include directory.

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/cstudy/CMakeDemo3$ cmake .
-- The C compiler identification is GNU 9.3.0
-- The CXX compiler identification is GNU 9.3.0
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Configuring done
-- Generating done
-- Build files have been written to: /mnt/d/cstudy/CMakeDemo3
maydlee@LAPTOP-U1MO0N2F:/mnt/d/cstudy/CMakeDemo3$ make
Scanning dependencies of target CMakeDemo3
[ 33%] Building CXX object CMakeFiles/CMakeDemo3.dir/src/function.cpp.o
[ 66%] Building CXX object CMakeFiles/CMakeDemo3.dir/src/main.cpp.o
[100%] Linking CXX executable CMakeDemo3
[100%] Built target CMakeDemo3
```

For more about Cmake(cmake tutorial):
https://cmake.org/cmake/help/latest/guide/tutorial/index.html

https://riptutorial.com/cmake

Exercises 1 (in-class)

Declare a structure named **stuinfo** and four function prototypes below in a **stuinfo.hpp**. Implement the four functions in a **stufun.cpp**. Write a **main.cpp** which contains main() and demonstrate all the features of the prototyped functions.

Write a **MakeLists.txt** for cmake to create Makefile automatically. Run cmake and make, and then run your program at last.

```
struct stuinfo
{
    char name[20];
    double score[3];
    double ave;
};
```

Function prototypes:

- **void inputstu(stuinfo stu[] , int n),** asks the user to enter each of the preceding items of information to set the corresponding members of the structure.
- **void showstu(stuinfo stu[] , int n)** ,displays the contents of the structure, one student one line.
- **void sortstu(stuinfo stu[] , int n),** sorts in descending order of average of three scores.
- **bool findstu(stuinfo stu[] , int n, char ch[]),** finds if given characters is the student's name.

```
-- Configuring done
-- Generating done
-- Build files have been written to: /mnt/d/csourcecode/2021Fall/lab04/exercise
maydlee@LAPTOP-U1MO0N2F:/mnt/d/csourcecode/2021Fall/lab04/exercise$ make
Scanning dependencies of target Stuinfo
[ 33%] Building CXX object CMakeFiles/Stuinfo.dir/main.cpp.o
[ 66%] Building CXX object CMakeFiles/Stuinfo.dir/stufun.cpp.o
[100%] Linking CXX executable Stuinfo
[100%] Built target Stuinfo
maydlee@LAPTOP-U1MO0N2F:/mnt/d/csourcecode/2021Fall/lab04/exercise$ ./Stuinfo
Please input information of 5 students:
Student 0's name:Zhang Xiaodong
Student 0's scores:67 87 55
Student 1's name:Wang xin
Student 1's scores:89 90 83
Student 2's name:Wu xu
Student 2's scores:90 93 91
Student 3's name:Hu zhulong
Student 3's scores:78 65 62
Student 4's name:Bai yuting
Student 4's scores:87 76 66
The information of 5students you input are:
Student 0 name: Zhang Xiaodong,scores: 67 87 55
Student 1 name: Wang xin,scores: 89 90 83
Student 2 name: Wu xu,scores: 90 93 91
Student 3 name: Hu zhulong,scores: 78 65 62
Student 4 name: Bai yuting,scores: 87 76 66

The descending order of the students:
Student 0 name: Wu xu,scores: 90 93 91 average: 91.3333
Student 1 name: Wang xin,scores: 89 90 83 average: 87.3333
Student 2 name: Bai yuting,scores: 87 76 66 average: 76.3333
Student 3 name: Zhang Xiaodong,scores: 67 87 55 average: 69.6667
Student 4 name: Hu zhulong,scores: 78 65 62 average: 68.3333

Please input the name you want to find:Bai yuti
Bai yuti is not in the students list.
```

Exercise 2 (Homework, by Sunday)

- Design a struct “DayInfo” which contains two enumeration types as its member. The first is an enum “Day” for (Sunday, Monday, ...), and the second is an enum “Weather” for (Sunny, Rainy, ...).
- Define a boolean function “bool canTravel(DayInfo)” . It will return true if the day is at weekend and the weather is good.
- Call function canTravel() in main().