南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

C/C++ Program Design

Lab 5, CMake

廖琪梅，王大兴

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 '#'.

Steps for generating a makefile and compiling on Linux using CMake:

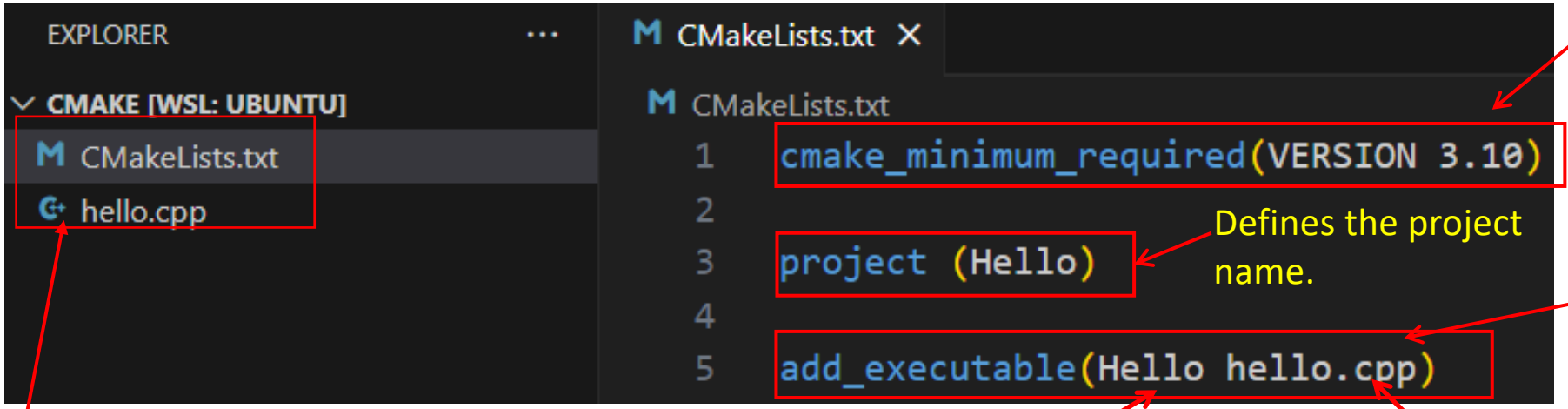
Step1: Writes the CMake configuration file **CMakeLists.txt**.

Step2: Executes the command **cmake PATH** to generate the **Makefile**. (PATH is the directory where the CMakeLists.txt resides.)

Step3: Compiles using the **make** command.

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.



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 hello.cpp.

The first parameter indicates the filename of executable file.

The second parameter indicates the source file.

Stores the CMakeLists.txt file in the same directory as the hello.cpp.

Suppose there is a hello.cpp

cmake > hello.cpp > ...

```cpp
#include <iostream>

int main()
{
    std::cout << "Hello World!" << std::endl;
}
```

In current directory, type **cmake .** to generate makefile. If cmake does not be installed, follow the instruction to install cmake.

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/CMake$ cmake .
```

```
Command 'cmake' not found, but can be installed with:

sudo apt install cmake
```

Install cmake first by instruction

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/CMake$ cmake .
-- The C compiler identification is GNU 9.4.0
-- The CXX compiler identification is GNU 9.4.0
-- 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/CMake
```

Run cmake to generate makefle, . indicates the CMakeList.txt is in the current directory.

Makefile file is created automatically after running cmake in the current directory. Except Makefile, there are other new files and folders.

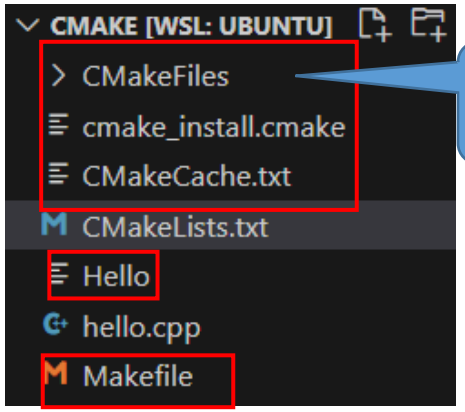
```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/CMake$ ls
CMakeCache.txt  CMakeFiles  CMakeLists.txt  Makefile  cmake_install.cmake  hello.cpp
```

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

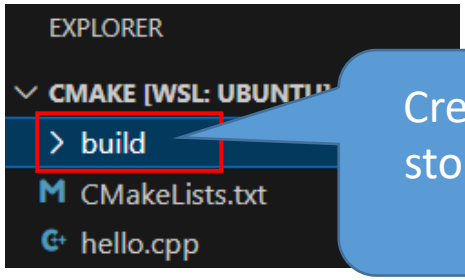
Execute make to compile the program.

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/CMake$ ./Hello
Hello World!
```

Run the program



Deletes all the building files and directory by CMake.



Creates an empty folder to store the building files and directory by CMake.

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/CMake$ cd build
maydlee@LAPTOP-U1MO0N2F:/mnt/d/CMake/build$ cmake ..
-- The C compiler identification is GNU 9.4.0
-- The CXX compiler identification is GNU 9.4.0
-- 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/CMake/build
```

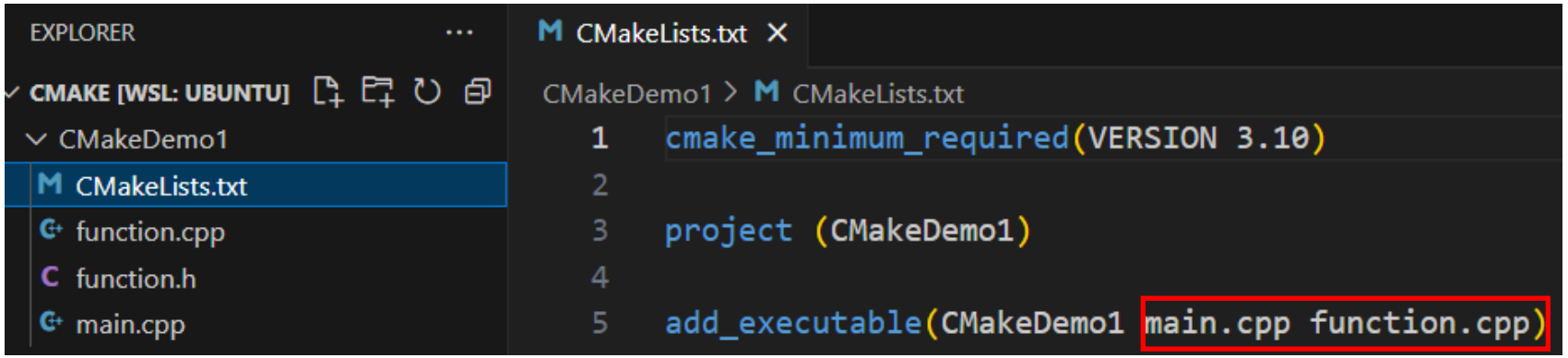
```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/CMake/build$ ls
CMakeCache.txt  CMakeFiles  Makefile  cmake_install.cmake
```

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/CMake/build$ make
Scanning dependencies of target Hello
[ 50%] Building CXX object CMakeFiles/Hello.dir/hello.cpp.o
[100%] Linking CXX executable Hello
[100%] Built target Hello
maydlee@LAPTOP-U1MO0N2F:/mnt/d/CMake/build$ ls
CMakeCache.txt  CMakeFiles  Hello  Makefile  cmake_install.cmake
maydlee@LAPTOP-U1MO0N2F:/mnt/d/CMake/build$ ./Hello
Hello World!
```

2. Multi-source files in a project

There are three files in the same directory.

```
./CmakeDemo1
      |
      +--- main.cpp
      |
      +--- function.cpp
      |
      +--- function.h
```



List all the source files using space as the separator.


```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/CMake$ cd CMakeDemo1
maydlee@LAPTOP-U1MO0N2F:/mnt/d/CMake/CMakeDemo1$ mkdir build
maydlee@LAPTOP-U1MO0N2F:/mnt/d/CMake/CMakeDemo1$ cd build
maydlee@LAPTOP-U1MO0N2F:/mnt/d/CMake/CMakeDemo1/build$ cmake ..
-- The C compiler identification is GNU 9.4.0
-- The CXX compiler identification is GNU 9.4.0
-- 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/CMake/CMakeDemo1/build
maydlee@LAPTOP-U1MO0N2F:/mnt/d/CMake/CMakeDemo1/build$ make
Scanning dependencies of target CMakeDemo1
[ 33%] Building CXX object CMakeFiles/CMakeDemo1.dir/main.cpp.o
[ 66%] Building CXX object CMakeFiles/CMakeDemo1.dir/function.cpp.o
[100%] Linking CXX executable CMakeDemo1
[100%] Built target CMakeDemo1
```

Creates a folder

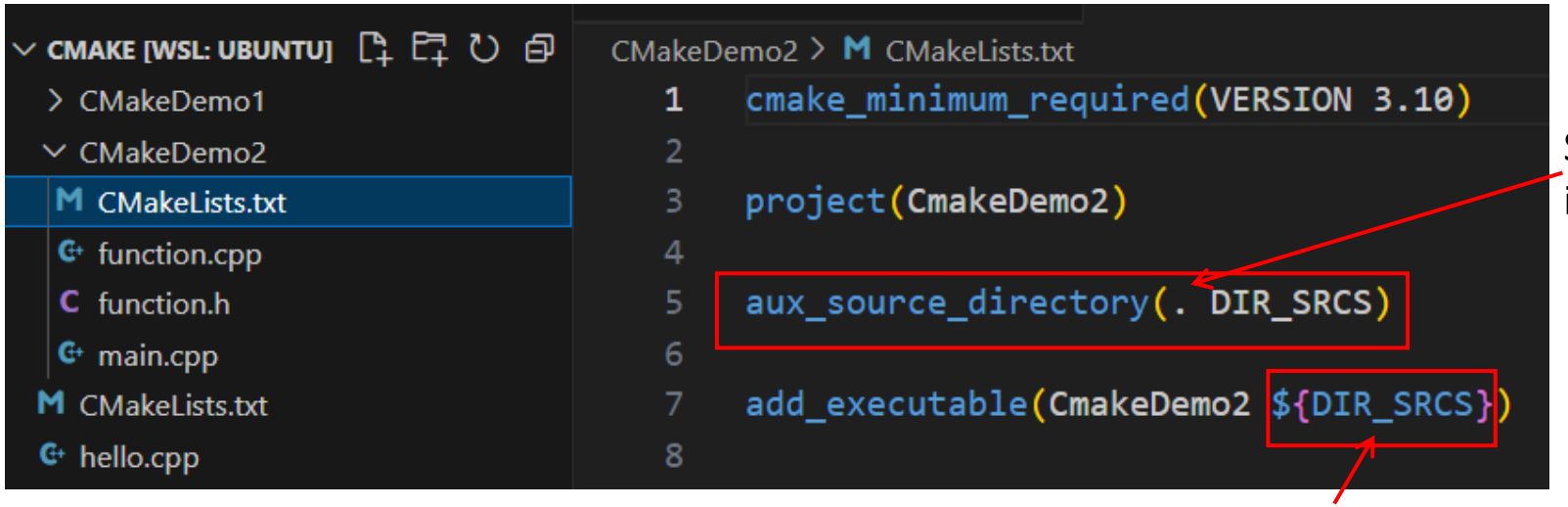
2. Multi-source files in a project

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



Stores all files in the current directory into DIR_SRCS variable.

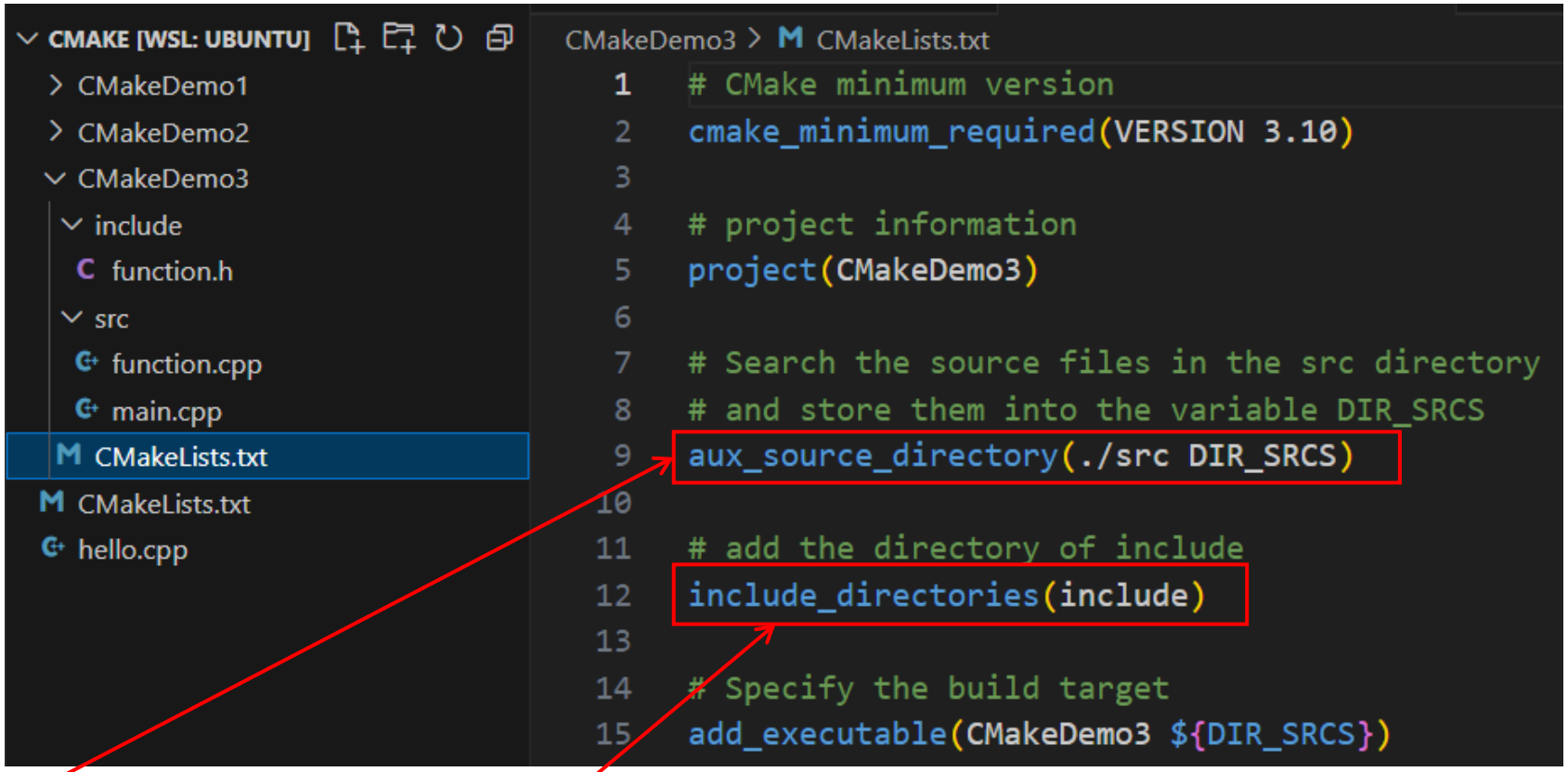
Compiles the source files in the variable by **${ }** into an executable file named CmakeDemo2


```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/CMake$ cd CMakeDemo2
maydlee@LAPTOP-U1MO0N2F:/mnt/d/CMake/CMakeDemo2$ mkdir build
maydlee@LAPTOP-U1MO0N2F:/mnt/d/CMake/CMakeDemo2$ cd build
maydlee@LAPTOP-U1MO0N2F:/mnt/d/CMake/CMakeDemo2/build$ cmake ..
-- The C compiler identification is GNU 9.4.0
-- The CXX compiler identification is GNU 9.4.0
-- 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/CMake/CMakeDemo2/build
maydlee@LAPTOP-U1MO0N2F:/mnt/d/CMake/CMakeDemo2/build$ make
Scanning dependencies of target CmakeDemo2
[ 33%] Building CXX object CMakeFiles/CmakeDemo2.dir/function.cpp.o
[ 66%] Building CXX object CMakeFiles/CmakeDemo2.dir/main.cpp.o
[100%] Linking CXX executable CmakeDemo2
[100%] Built target CmakeDemo2
```

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

We write CMakeLists.txt in CmakeDemo3 folder.

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



All .cpp files are in the **src** directory

Include the header file which is stored in **include** directory.


```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/CMake$ cd CMakeDemo3
maydlee@LAPTOP-U1MO0N2F:/mnt/d/CMake/CMakeDemo3$ mkdir build
maydlee@LAPTOP-U1MO0N2F:/mnt/d/CMake/CMakeDemo3$ cd build
maydlee@LAPTOP-U1MO0N2F:/mnt/d/CMake/CMakeDemo3/build$ cmake ..
-- The C compiler identification is GNU 9.4.0
-- The CXX compiler identification is GNU 9.4.0
-- 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/CMake/CMakeDemo3/build
maydlee@LAPTOP-U1MO0N2F:/mnt/d/CMake/CMakeDemo3/build$ 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 cmake tutorial:

https://cmake.org/cmake/help/latest/guide/tutorial/index.html

https://riptutorial.com/cmake

Exercise 1

```
#include<stdio.h>

int main()
{
   int a[]={2,4,6,8,10},y=1,*p;
   p=&a[1];

   printf("a = %p\np = %p\n",a, p);

   for(int i = 0; i < 3; i++)
      y += *(p+i);

   printf("y = %d\n\n",y);

   int b[5]={1,2,3,4,5};
   int *ptr=(int*)(&b+1);

   printf("b = %p\nb+4 = %p\nptr = %p\n",b,b+4,ptr);
   printf("%d,%d\n",*(b+1),*(ptr-1));

   return 0;

}
```

Run the program and explain the result to SA.

Exercise 2

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

int main()
{
   int a[][4]={1,3,5,7,9,11,13,15,17,19};
   int *p=*(a+1);
   p += 3;
   cout << "*p++ = " << *p++ << ",*p = " << *p << endl;

   const char *pc = "Welcome to programming.", *r;
   long *q = (long *)pc;
   q++;
   r = (char *)q;

   cout << r << endl;

   unsigned int m = 0x3E56AF67;
   unsigned short *pm = (unsigned short *) &m;

   cout << "*pm = " <<  hex << *pm << endl;

   return 0;
}
```

Run the program and explain the result to SA.

Exercises 3

Declare a structure named **stuinfo** and two function prototypes below in a **stuinfo.hpp**. Implement the two 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 the 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 and compute the average score for each student.
- **void showstu(stuinfo stu[] , int n)** ,displays the contents of the structure, one student one line.