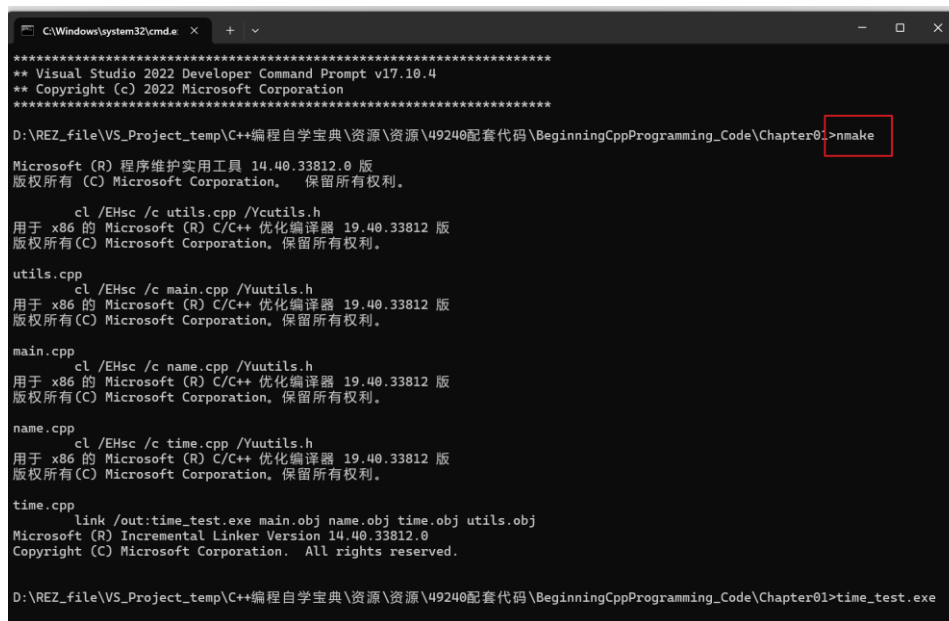


## 第一个交互式 C++ 项目小结

1. 以下命令行代码的主要功能是演示了如何使用 C++ 编写一个简单的交互式程序，**通过编译和链接多个源文件来组织代码**，并展示了如何获取和处理系统时间。



```
C:\Windows\system32\cmd.exe
*****
** Visual Studio 2022 Developer Command Prompt v17.10.4
** Copyright (c) 2022 Microsoft Corporation
*****

D:\REZ_file\VS_Project_temp\C++编程自学宝典\资源\资源\49240配套代码\BeginningCppProgramming_Code\Chapter01>nmake

Microsoft (R) 程序维护实用工具 14.40.33812.0 版
版权所有 (C) Microsoft Corporation. 保留所有权利。

    cl /EHsc /c utils.cpp /Yuutils.h
用于 x86 的 Microsoft (R) C/C++ 优化编译器 19.40.33812 版
版权所有 (C) Microsoft Corporation. 保留所有权利。

utils.cpp
    cl /EHsc /c main.cpp /Yuutils.h
用于 x86 的 Microsoft (R) C/C++ 优化编译器 19.40.33812 版
版权所有 (C) Microsoft Corporation. 保留所有权利。

main.cpp
    cl /EHsc /c name.cpp /Yuutils.h
用于 x86 的 Microsoft (R) C/C++ 优化编译器 19.40.33812 版
版权所有 (C) Microsoft Corporation. 保留所有权利。

name.cpp
    cl /EHsc /c time.cpp /Yuutils.h
用于 x86 的 Microsoft (R) C/C++ 优化编译器 19.40.33812 版
版权所有 (C) Microsoft Corporation. 保留所有权利。

time.cpp
    link /out:time_test.exe main.obj name.obj time.obj utils.obj
Microsoft (R) Incremental Linker Version 14.40.33812.0
Copyright (C) Microsoft Corporation. All rights reserved.

D:\REZ_file\VS_Project_temp\C++编程自学宝典\资源\资源\49240配套代码\BeginningCppProgramming_Code\Chapter01>time_test.exe
```

具体步骤和功能解释如下：

### 1. 编译过程：

- 使用 `cl` 命令编译 `utils.cpp`、`main.cpp`、`name.cpp`、`time.cpp` 这四个源文件。
- 编译选项 `/EHsc` 表示启用异常处理，`/c` 表示编译但不链接，生成各自的目标文件（.obj 文件）。

### 2. 链接过程：

- 使用 `link` 命令将 `main.obj`、`name.obj`、`time.obj`、`utils.obj` 这四个目标文件链接在一起，生成可执行文件 `time\_test.exe`。
- `/out:time\_test.exe` 指定生成的可执行文件名。

### 3. 程序执行：

- 运行生成的 `time\_test.exe` 可执行文件。
- 程序首先要求用户输入他们的名字。
- 输入名字后，程序会调用相应的功能函数，获取当前系统时间，并将用户输入的名字和时间一起输出到控制台。

### 4. 功能实现：

- `utils.cpp` 中可能包含一些通用的函数或者类，供其他文件调用。
- `main.cpp` 包含程序的入口点 `main` 函数，负责与用户的交互和调用其他功能。
- `name.cpp` 和 `time.cpp` 可能包含根据需要定义的函数或者类，这些文件编译后生成对应的目标文件。
- `time.cpp` 中可能包含获取系统时间的函数实现，例如通过 `std::time` 和相关库函数来获取当前时间，并格式化输出。



D:\REZ\_file\VS\_Project\_temp\C++编程自学宝典\资源\资源\49240  
配套代码\BeginningCppProgramming\_Code\Chapter01>nmake  
这句的nmake有什么作用



在您提供的命令中,`nmake`是一个命令,用于调用 Microsoft Visual Studio 自带的 nmake 工具,用于执行 Makefile 文件或者 Makefile 相似的规则文件。具体来说,对于您的情况:

#### 1. 作用:

- `nmake` 命令会执行当前目录下的 Makefile 文件,这个文件描述了如何编译和链接项目中的源代码文件。
- Makefile 中通常包含了一系列规则,规定了每个源文件如何编译,如何链接生成最终的可执行文件,以及可能的其他操作,比如清理临时文件等。

#### 2. 执行过程:

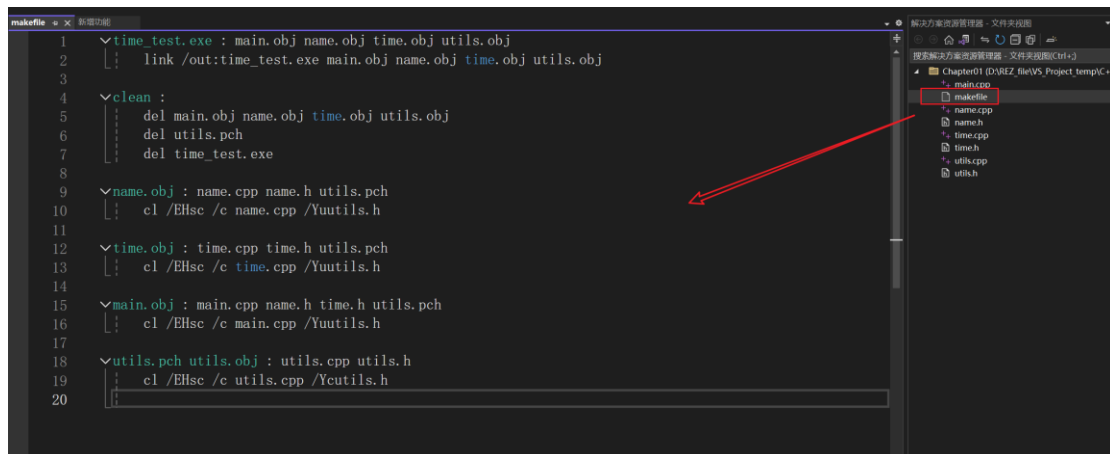
- 当您运行 `nmake` 命令时,它会依据当前目录下的 Makefile 中的规则,逐一执行编译、链接、以及其他必要的操作。
- 在您提供的命令行输出中,可以看到 `nmake` 执行了编译(使用 `cl` 命令编译每个源文件)、链接(使用 `link` 命令链接生成可执行文件)的操作。

#### 3. 适用环境:

- `nmake` 是 Microsoft Visual Studio 提供的一个版本控制工具,通常在 Windows 平台上使用。
- 它主要用于管理大型项目的编译和构建过程,通过自动化编译过程可以显著简化和加快开发人员的工作流程。

总结来说,`nmake` 的作用是根据 Makefile 文件定义的规则,自动化地执行编译和链接操作,从而构建和生成您的项目的可执行文件或者其他目标文件。

2. Makefile 里的代码实现了一个简单的编译和链接过程,通过命令行工具调用编译器和链接器,从源文件生成目标文件,最后链接成可执行文件 time\_test.exe。同时定义了清理操作,用于删除中间文件和可执行文件。



#### 3. time.cpp

获取伦敦格林威治时间:

```
#include "utils.h"  
#include "time.h"
```

```

void PrintTime()
{
    std::time_t now = std::time(nullptr);
    std::cout << ", the time and date are " << std::asctime(gmtime(&now)) << std::endl;
}

```

修改后，获取当地的北京时间：

```

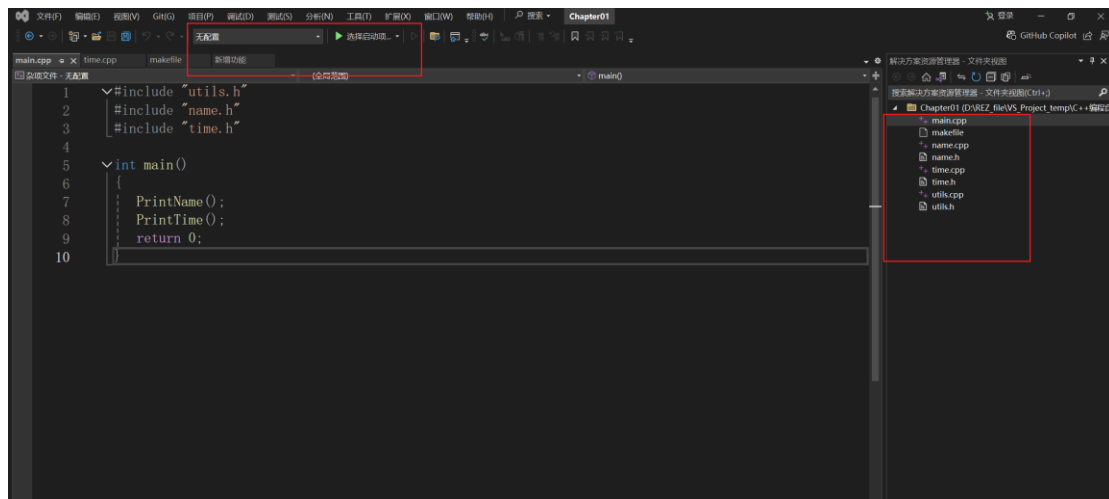
#include "utils.h"
#include "time.h"

void PrintTime()
{
    std::time_t now = std::time(nullptr);
    std::tm* localTime = std::localtime(&now); // 获取本地时间结构体指针

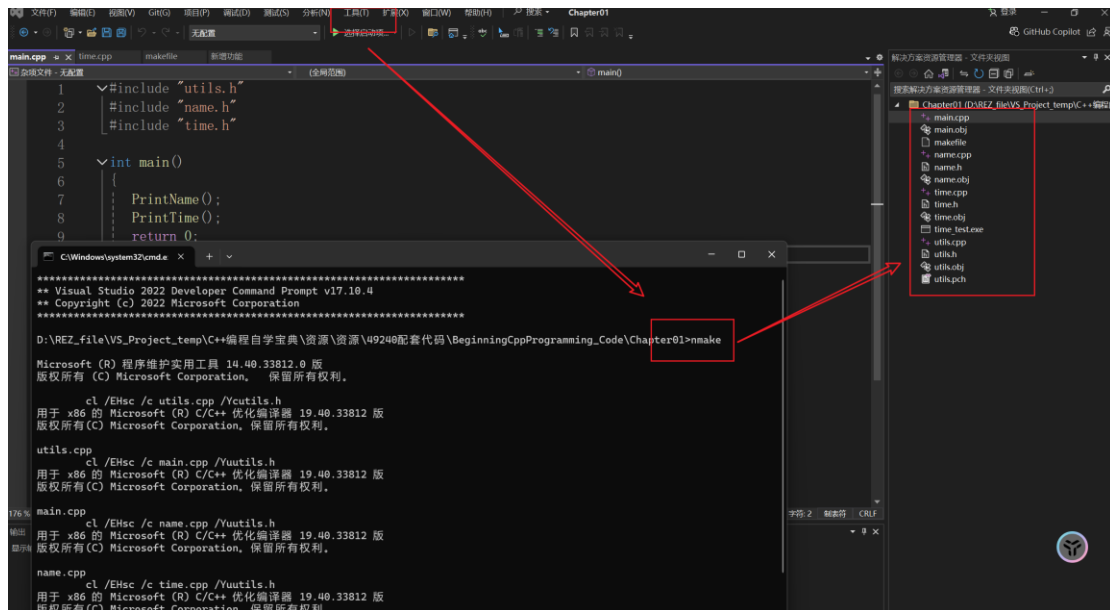
    if (localTime)
    {
        // 输出本地时间
        std::cout << ", the time and date are " << std::asctime(localTime) << std::endl;
    }
    else
    {
        std::cerr << "Failed to get local time!" << std::endl;
    }
}

```

4.整个项目执行前，界面如下：



命令行通过 nmake 运行 makefile 文件后：



直接可运行 exe 程序:

