# 实验3-3：词法分析之NFA转换成DFA

## 一、实验目的

通过上机实习，利用子集法实现NFA到DFA的转换，掌握这一过程中涉及的确定有穷自动机的概念，以及子集法的算法原理及实现技术，培养分析问题和解决问题的能力。

## 二、实验要求

在实验3-2的基础上，将实验3-2创建的NFA转换成DFA。

## 三、参考设计思路

从 = 构造等价的 = 的基本方法是：首先将从状态 出发经过任意条 弧所能到达的状态所组成的集合作为 的初态 ，然后从 出发，经过对输入符号 的状态转移所能到达的状态（包括读输入符号 之前或之后所有可能的 转移所能到达的状态）所组成的集合作为 的新状态，如此重复，直到不再有新的状态出现为止。下面给出构造 及 的算法描述如下:

（1）置 中的状态集 和 为 集。

（2）给出 的初态，并把 置为未标记状态后加入到 中(未标记状态即新状态)。  
 （3）如果 中存在未标记的状态 ，则进行如下变换（即求的后继状态 ）。

1）对于每个，置，。如果 不在 中，则将 置为无标记的状态添加到 中，且把状态转移 添加到 中，如果 中至少含有一个元素是 的终态，则把 置为 的终态，即把 添加到 中。

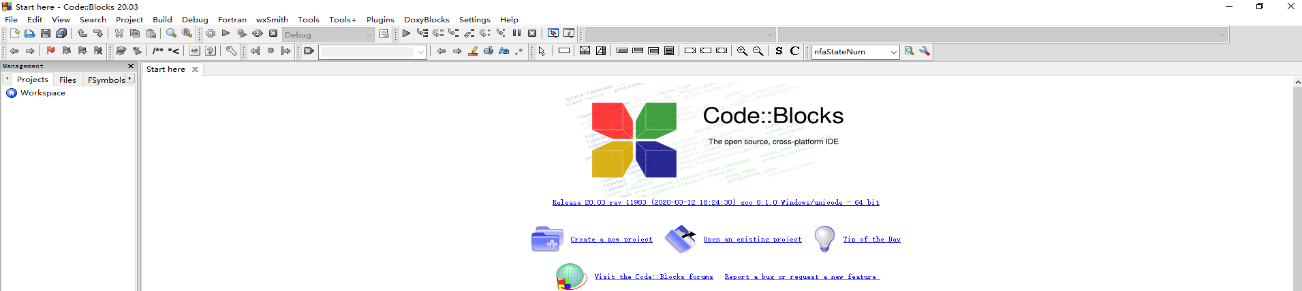
2)对 置标记（表示 不再是新加入 中的状态）。

（4）重复进行步骤（3），直到 中不再含有未标记的状态为止。

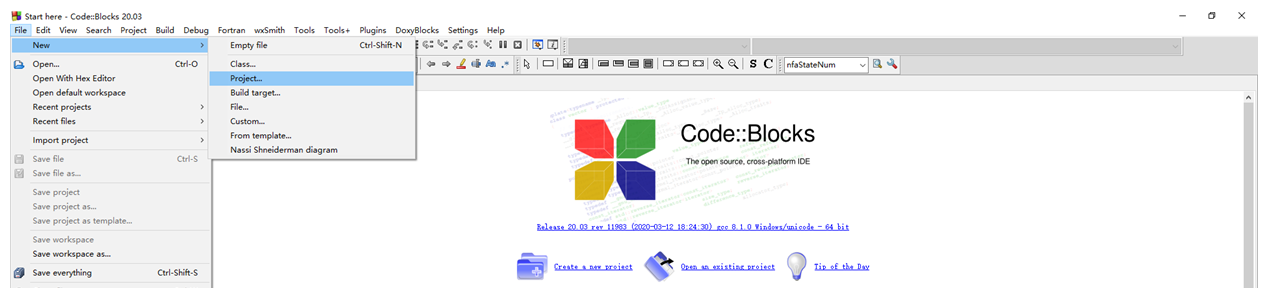
（5）重命名 中的状态，最后获得等价的 。

实验具体步骤如下：

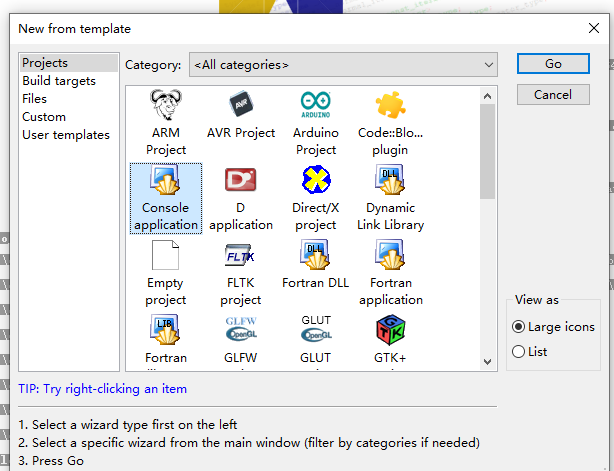
在虚拟机中，双击桌面上的“Code::Blocks”应用程序图标，启动“Code::Blocks”集成开发环境，如下图所示：



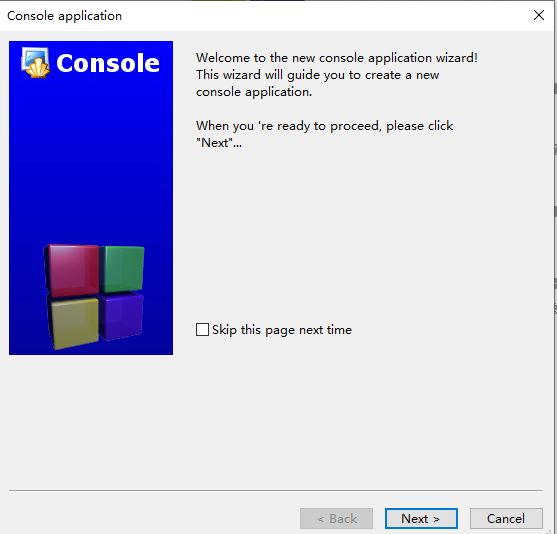
点击“File”-“New”-“project”，新建一个项目，



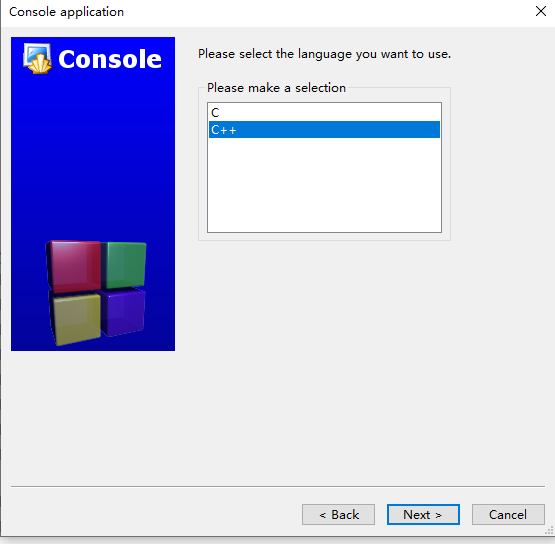
在新建项目类型中选择“Console application”，点击“Go”按钮，



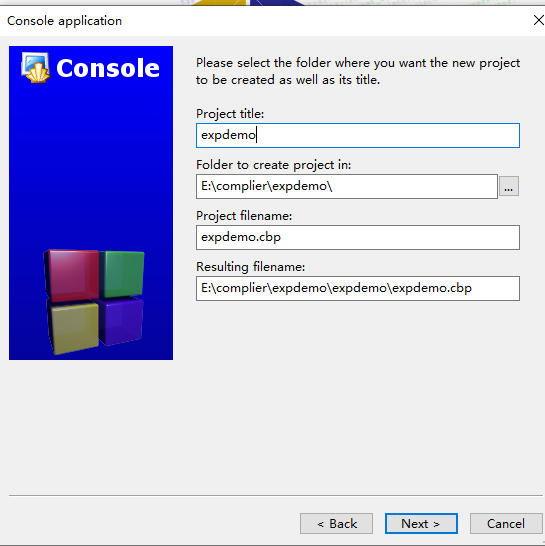
进入下面页面：



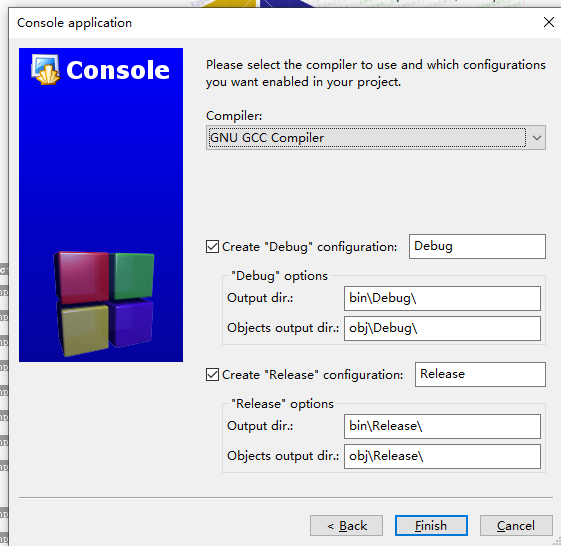
点击“Next”按钮，进入下面页面：



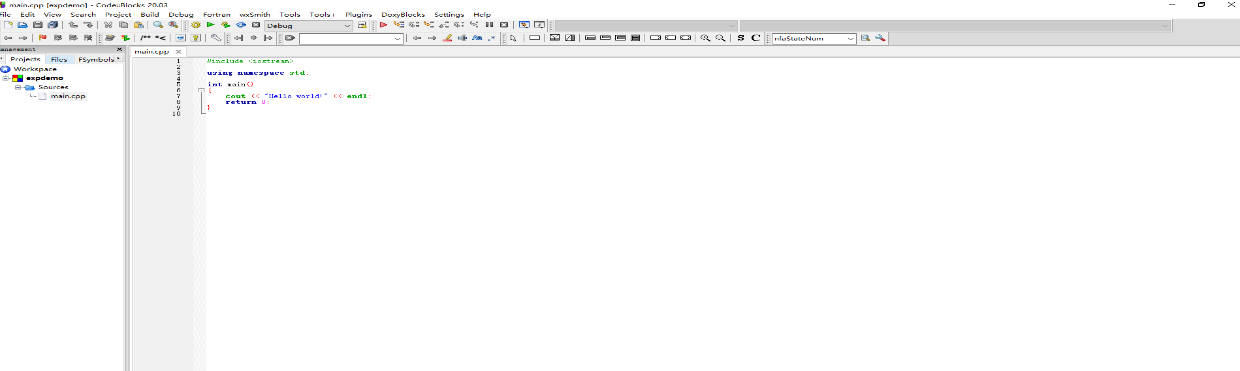
选择“C++”选项后，点击“Next”进入下一页面，在该页面中填入项目名“project title”，选择项目所在的位置，



填完项目名，选择好项目所在位置后，点击“Next”按钮，进入下面页面，不需要修改选项，点击“Finish”按钮，完成项目创建：

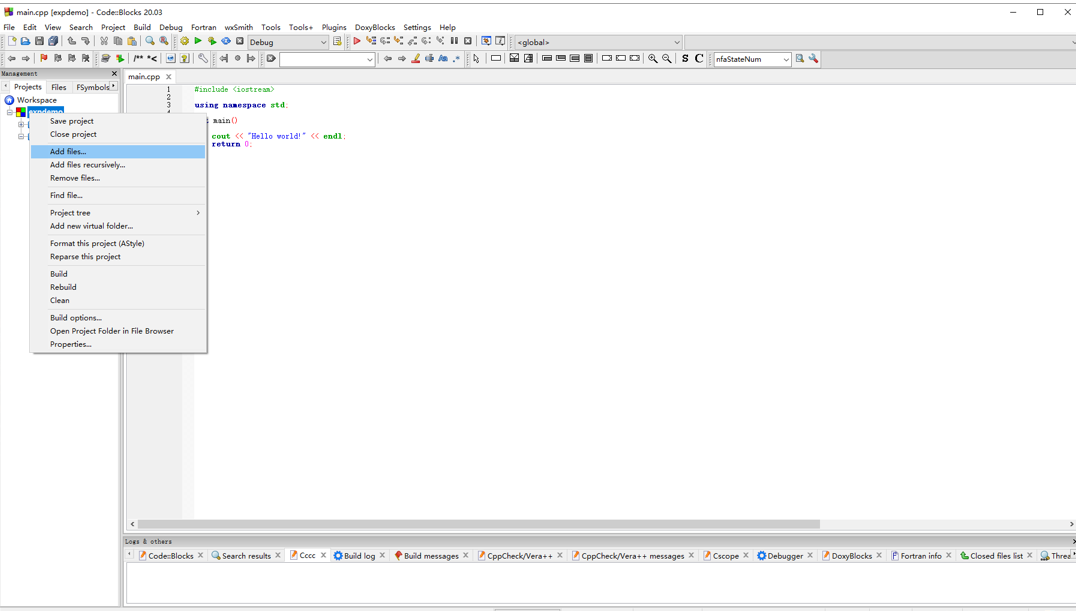


项目创建成功后，页面如下图所示，

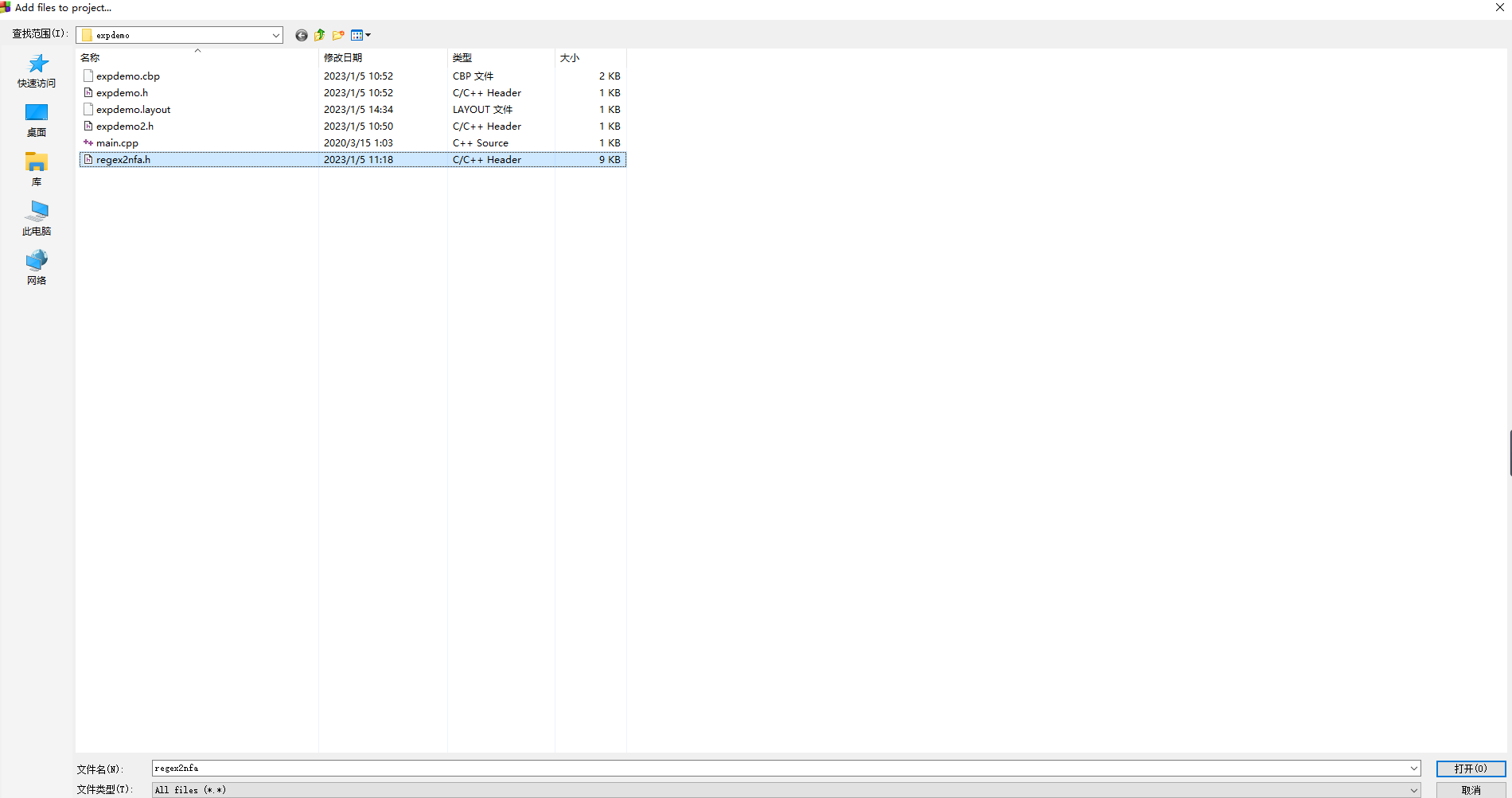


因为本次实验的NFA可由实验3-2方法构建，因此需要将实验3-2中编写的regex2nfa.h导入到本项目中，具体步骤如下：

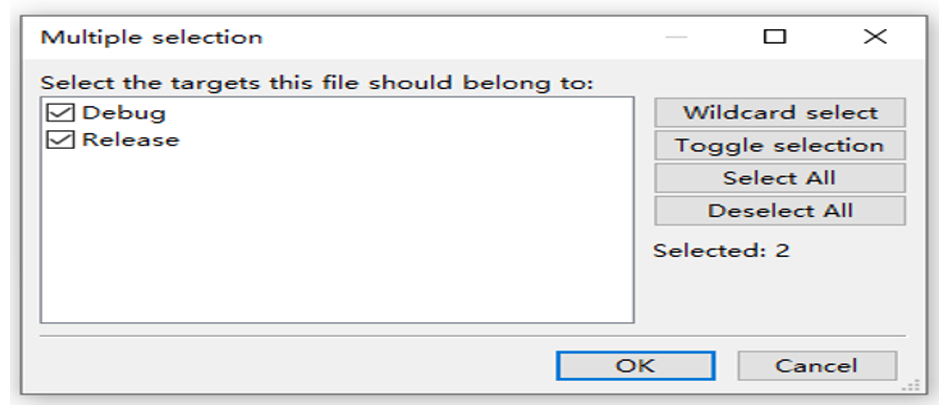
将regex2nfa.h文件拷贝到本次实验新建项目所在文件夹，打开项目之后，右键点击项目名，出现如下界面，



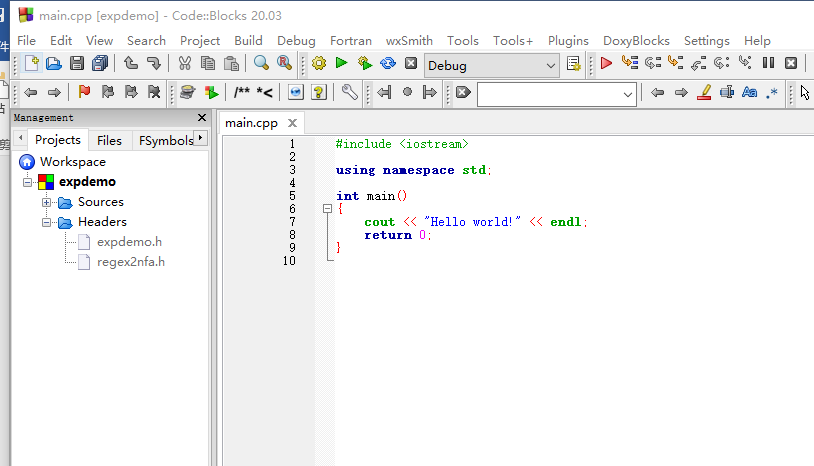
左键点击“Add files…”，出现如下对话框，



选择regex2nfa.h文件后，左键点击“打开”按钮，出现如下对话框：

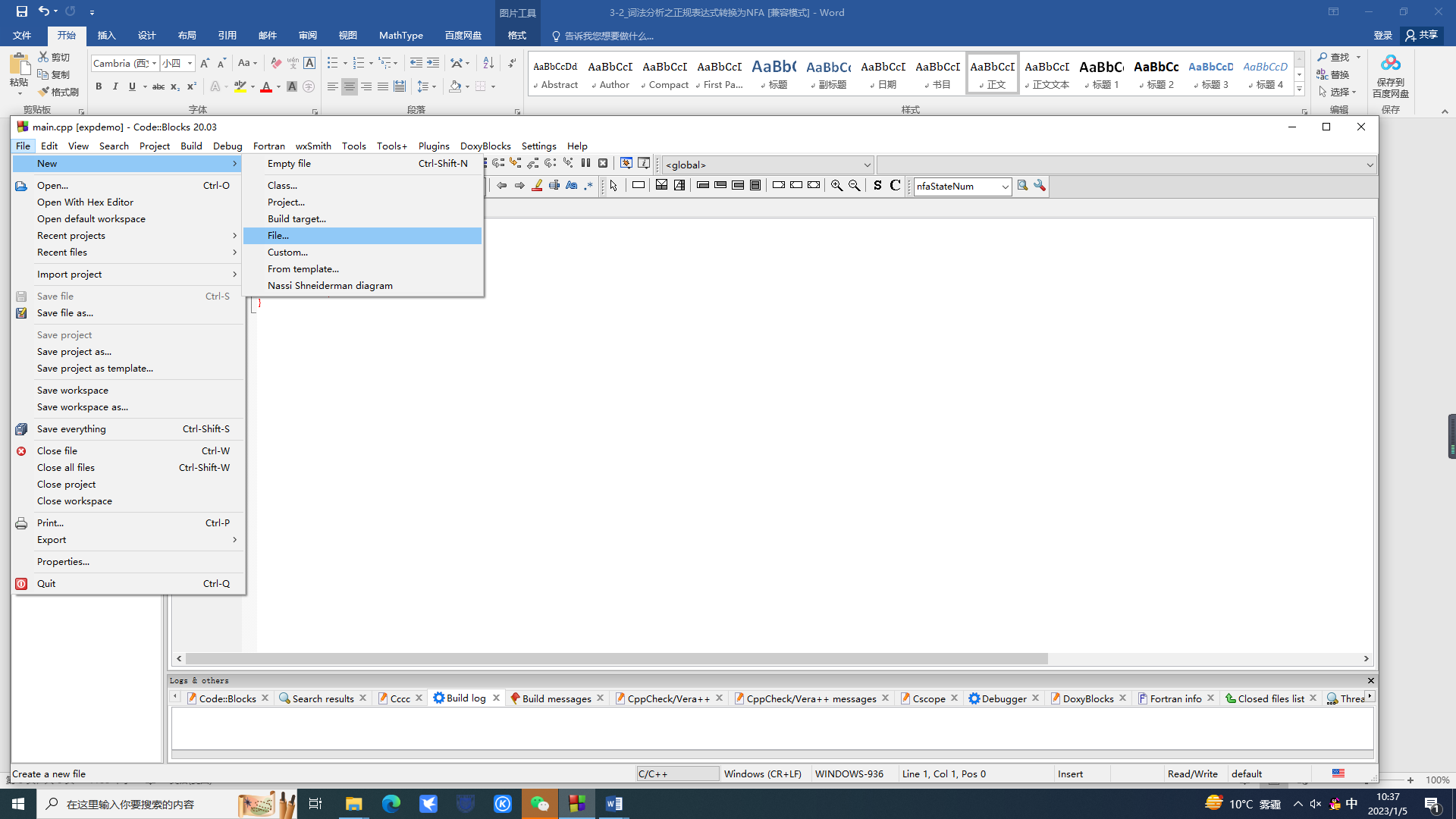


左键点击“OK”按钮后，就可以将regex2nfa.h文件加入到本实验所建的工程中了。

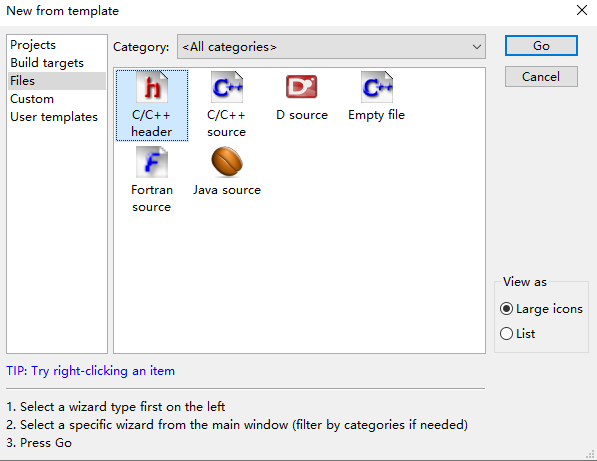


因为本次实验也是3-4实验的基础，所以将核心代码放在nfa2dfa.h文件中，方便后续实验调用，新建\*.h文件步骤如下：

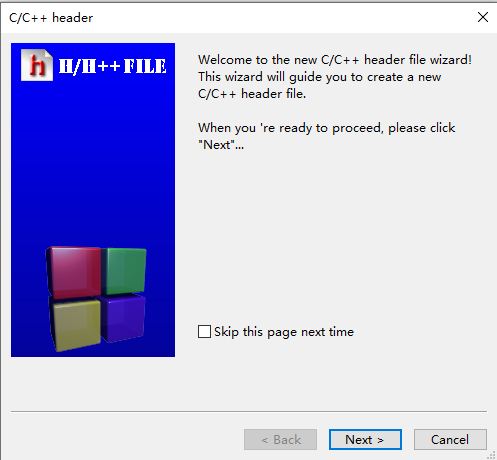
点击“File”-“New”-“file”，新建一个文件，



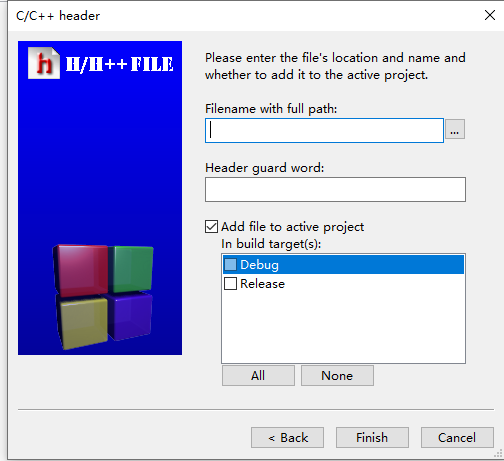
在新建文件类型中选择“C/C++ header”类型，点击“Go”按钮，



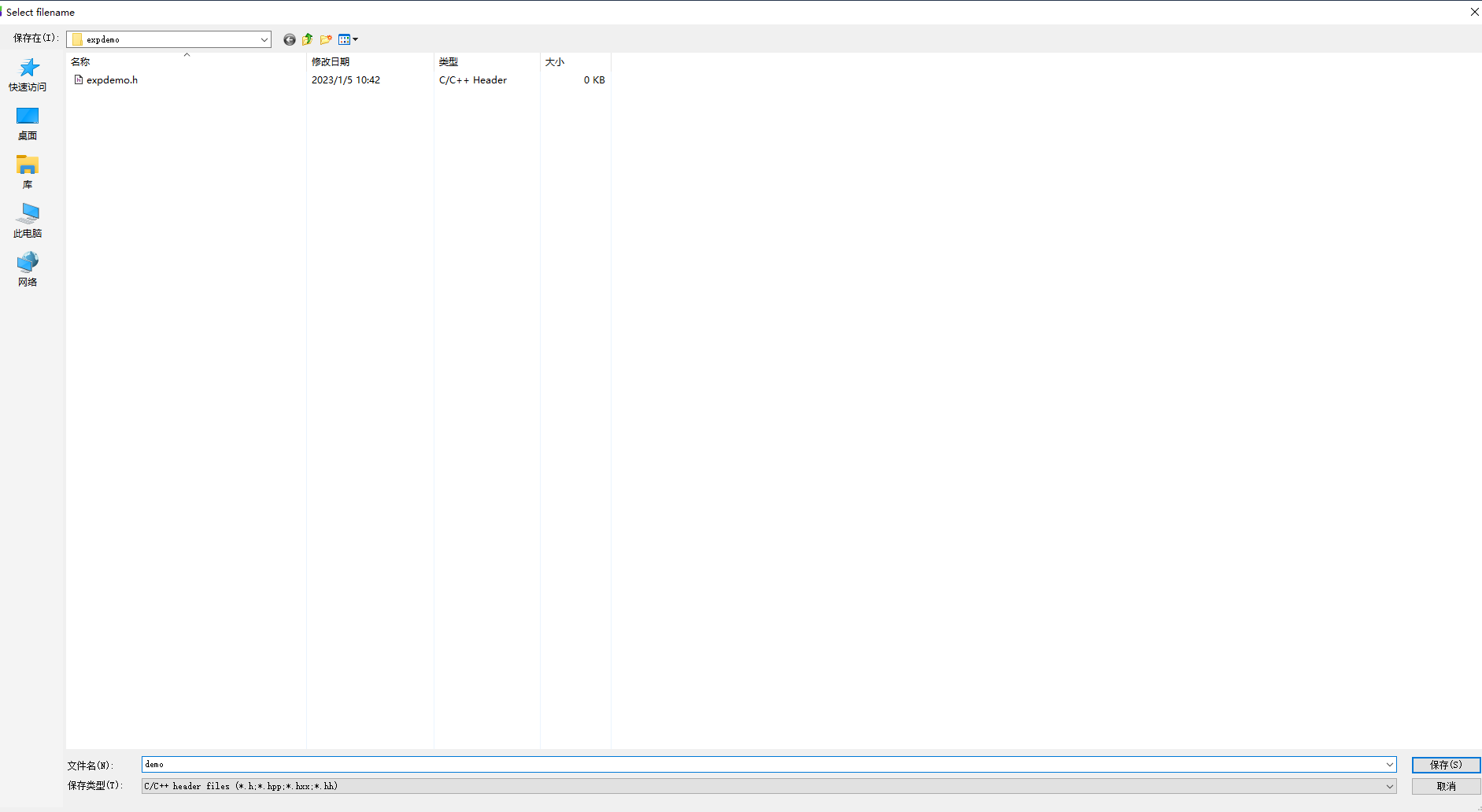
进入下面页面，



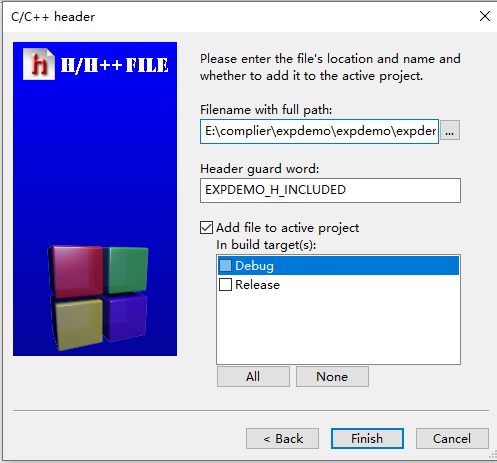
点击“Next”按钮，进入下面页面，



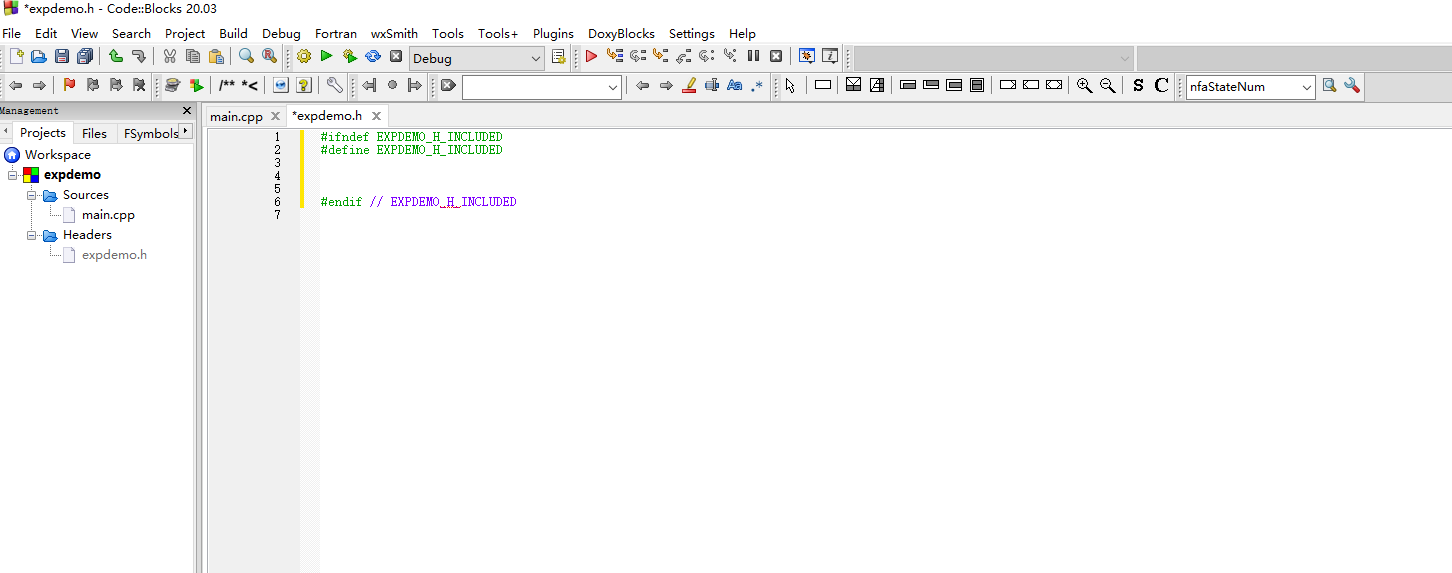
在“Filename with full path”下，通过点击“…”进入下面页面，选择文件所在的位置（建议和主函数放在同一个文件夹下），



填入文件名，点击确认后进入下面页面：

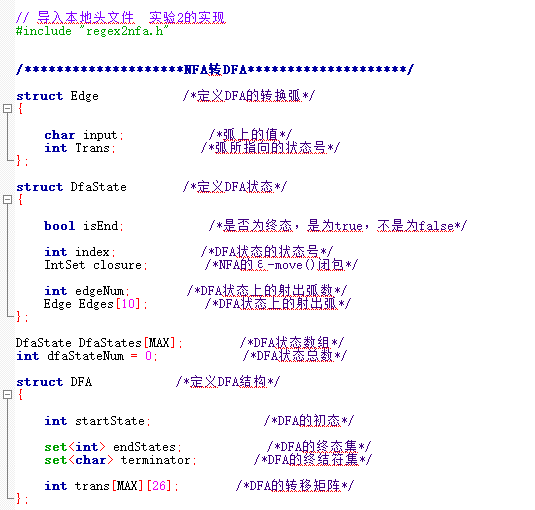


点击“Finish”按钮后，生成\*.h文件，如下图所示：

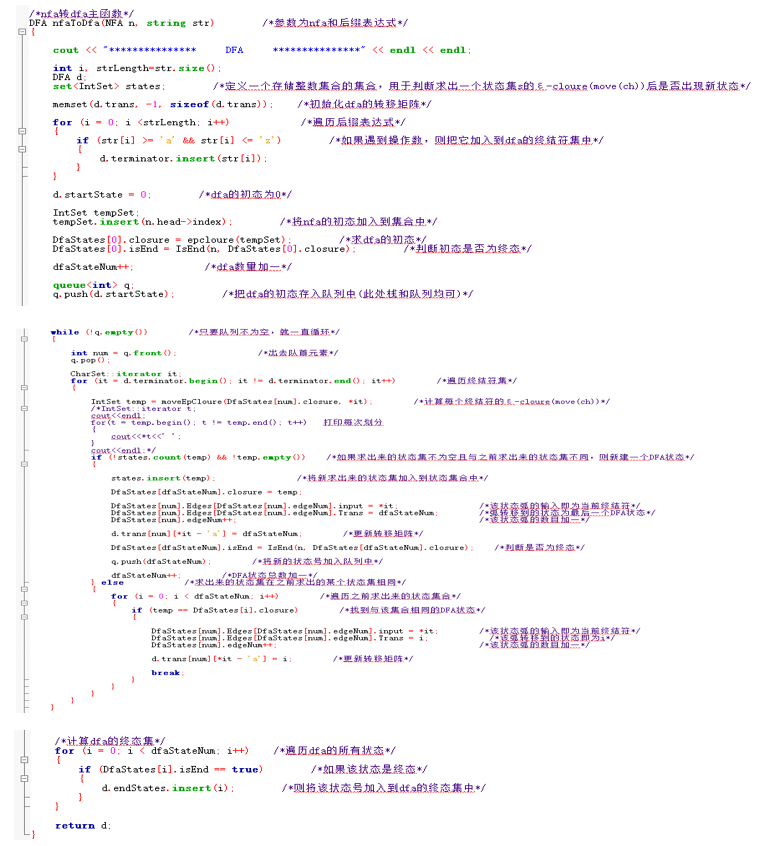


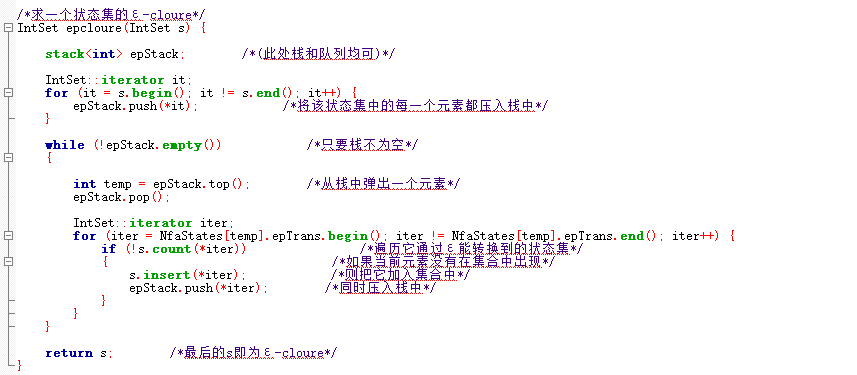
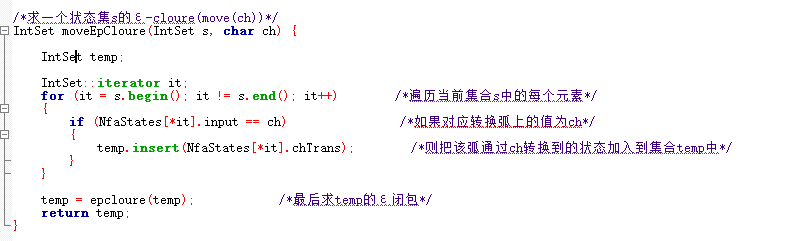
可以在新建的头文件中（下面以nfa2dfa.h文件名），按下面步骤完成程序编写：

首先进行初始化定义如下：

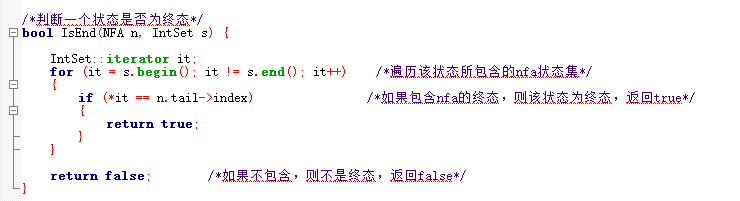


编写nfaToDfa()函数如下：

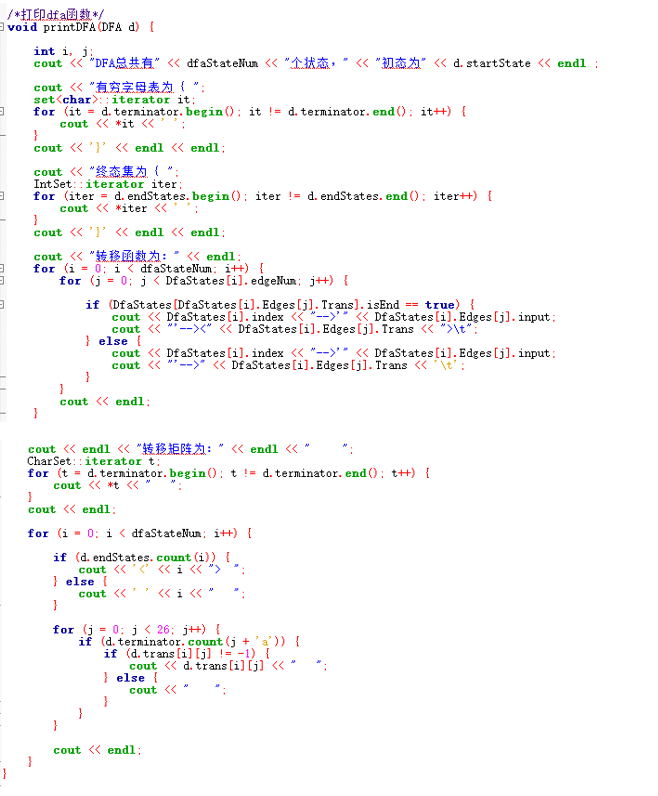
nfaToDfa()函数中，所涉及的epcloure()函数需要自行编写如下：

moveEpCloure()函数编写如下： 

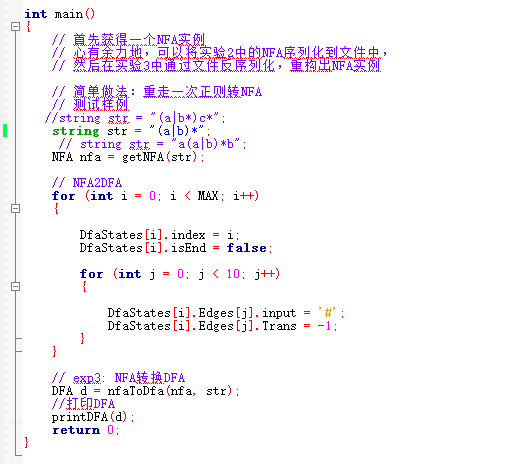
IsEnd()函数编写如下：



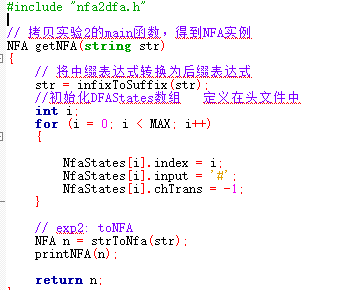
为了将生成的DFA打印出来，可以编写printDFA()函数如下：



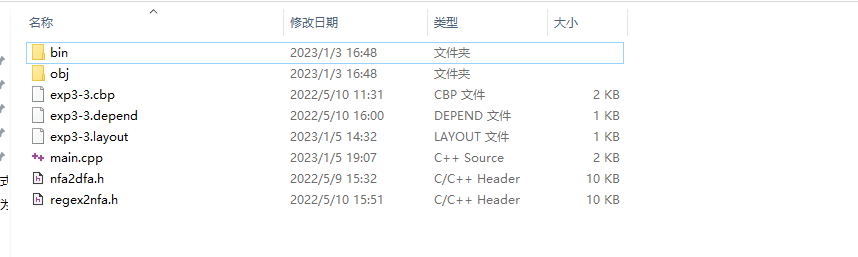
在完成nfa2dfa.h文件编写后，在main.cpp中按照下面步骤完成程序编写：



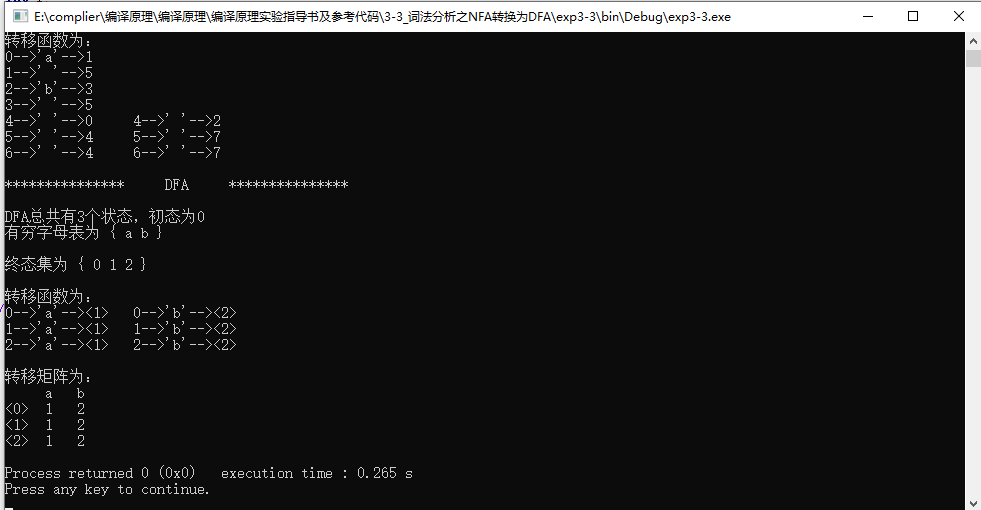
在main()函数中，需要自行编写getNFA()函数如下：



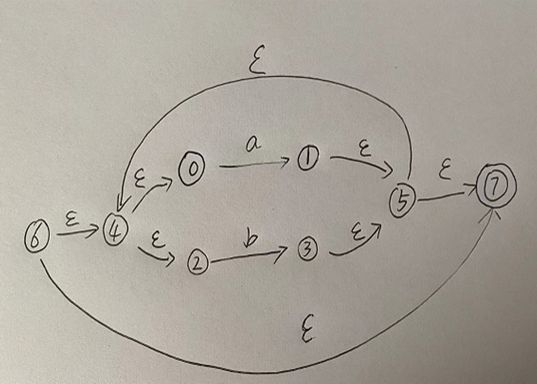
本次实验代码在“exp3\_3”文件夹下，双击“exp3-3.cbp”即可打开项目。



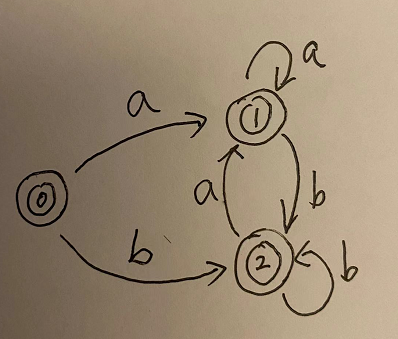
点击“Run”运行结果下图所示：



其中NFA所对应的状态转换图为：



转换后的DFA所对应的状态转换图为：



**四、总结**

通过本次实验学习，要求学员掌握编写NFA转换成DFA的方法。