****

**计算机组成原理实验报告**

**题目：**从⾼级语⾔到机器语⾔

**姓 名： 段欣然**

**专 业： 计算机科学与技术**

**年 级： 2020级**

**学 号： 202011081033**

**任课教师： 王志春**

**完成日期： 2021年3月8日**

**人工智能学院**

1. 实验要求

对下面的c语言代码进行编译、汇编、链接（省略预编译阶段），最终生成可执行文件。对生成的可执行文件进行反汇编，观察汇编文件和可执行文件。

#include <stdio.h>

int main(){

int a = 1, b = 2;

int c = a + b;

printf("%d\n", c);

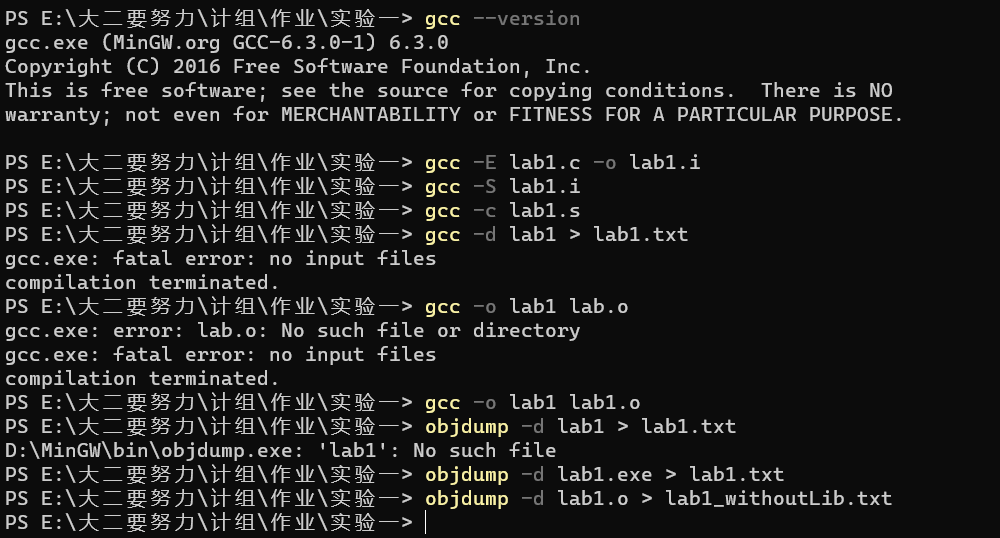
return 0;

}

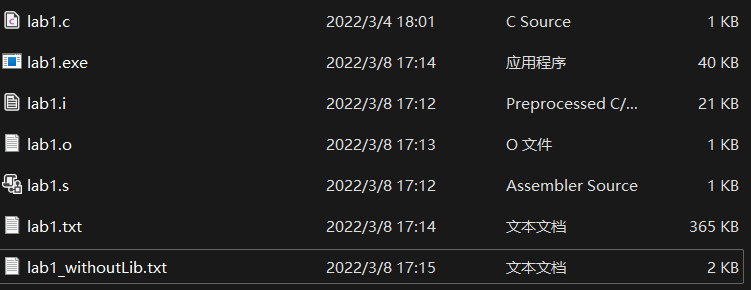
1. 实验结果与分析

## 实验步骤

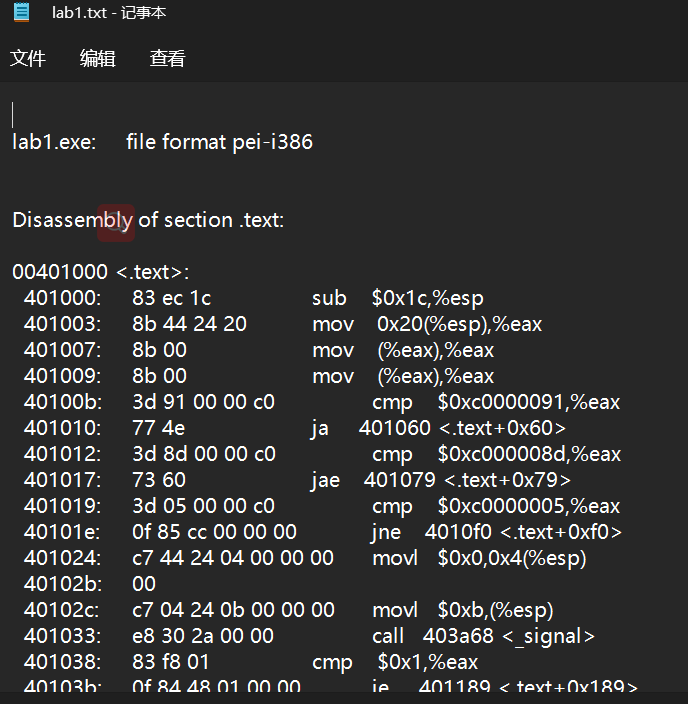
1. gcc预编译lab1.c并输出文件lab1.i
2. 对lab1.i进行编译
3. 对lab1.s进行汇编
4. 生成可执行文件lab1
5. 对可执行文件lab1反汇编生成目标文件lab1.txt
6. 为便于观察lab1.c程序内容，对lab1.o反汇编生成文件lab1\_withoutLib.txt，详细实验过程见下图。



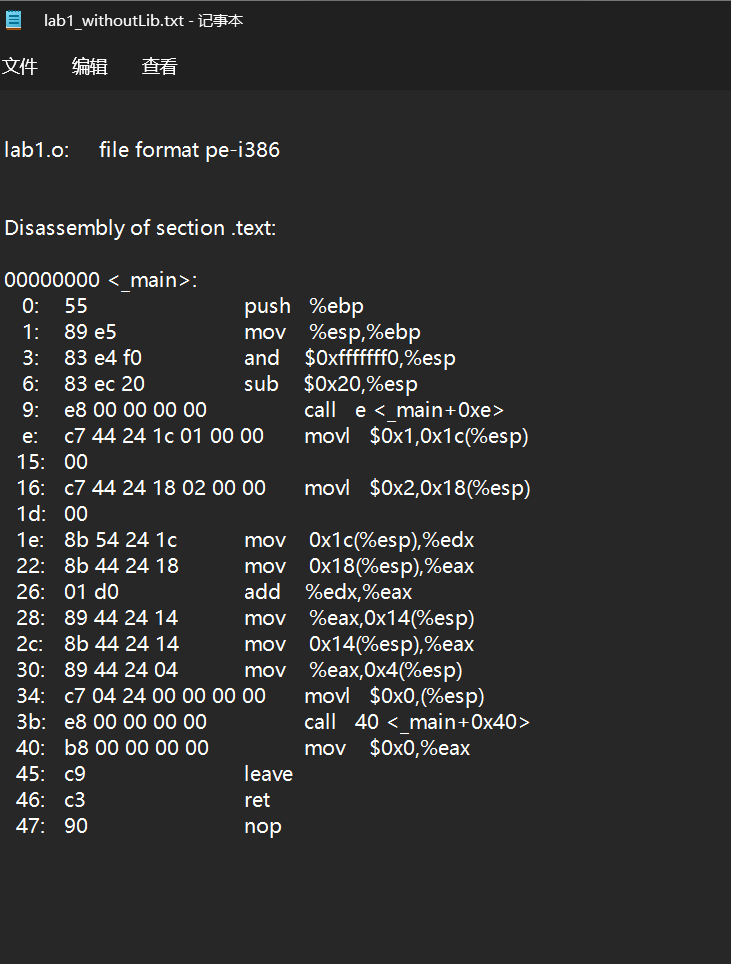
图表 1具体实验过程



图表 2编译、汇编、反汇编后生成文件截图

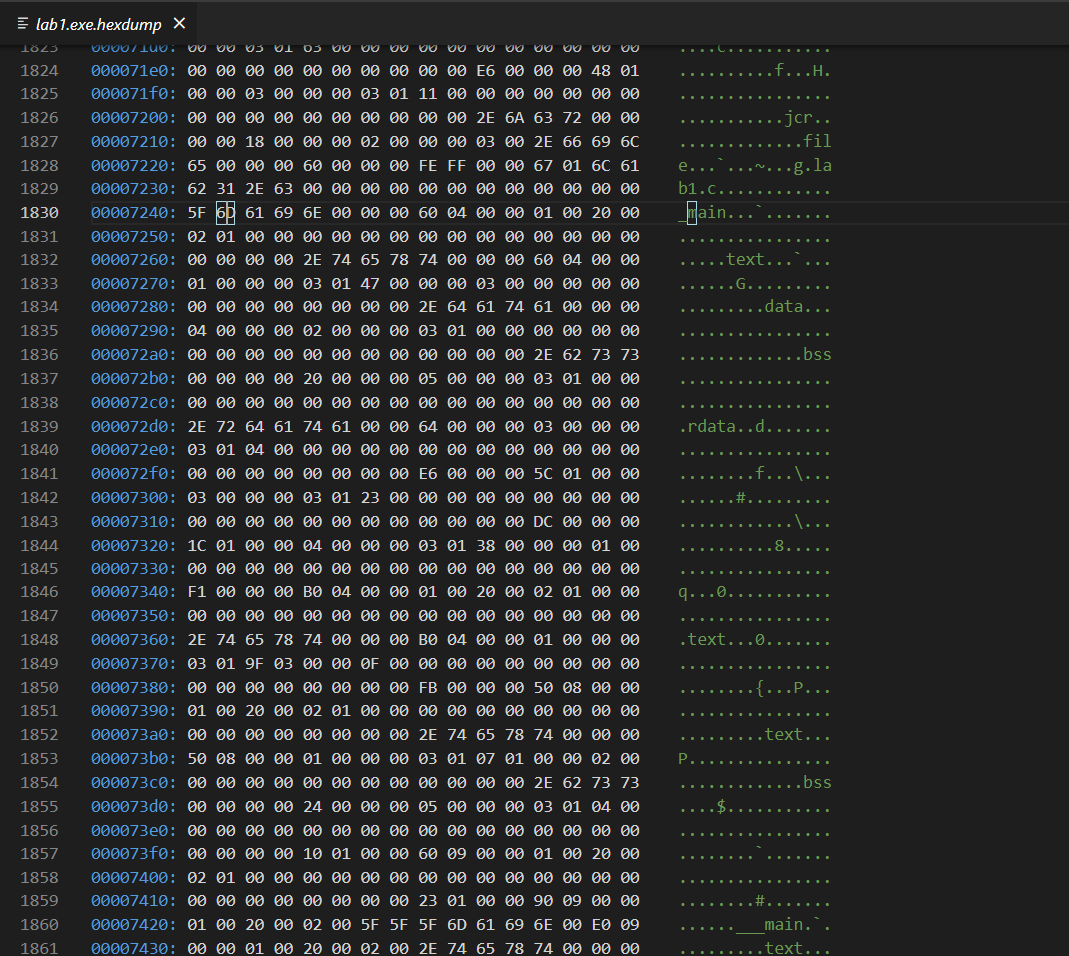


图表 3lab1.txt部分内容截图



图表 4lab1\_withoutLib.txt全部内容截图

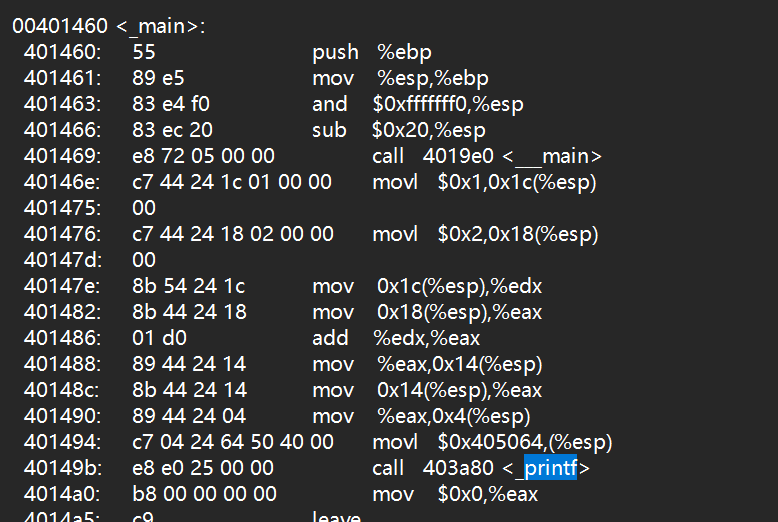
## 观察可执⾏⽬标⽂件的内容



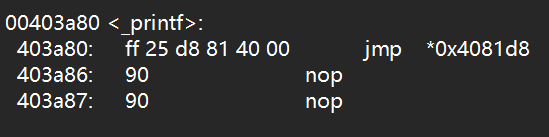
图表 5可执行目标文件内容（十六进制表示）

## 实验结果分析

1. 函数printf()对应的机器代码段



图表 6先定位到主函数调用printf函数

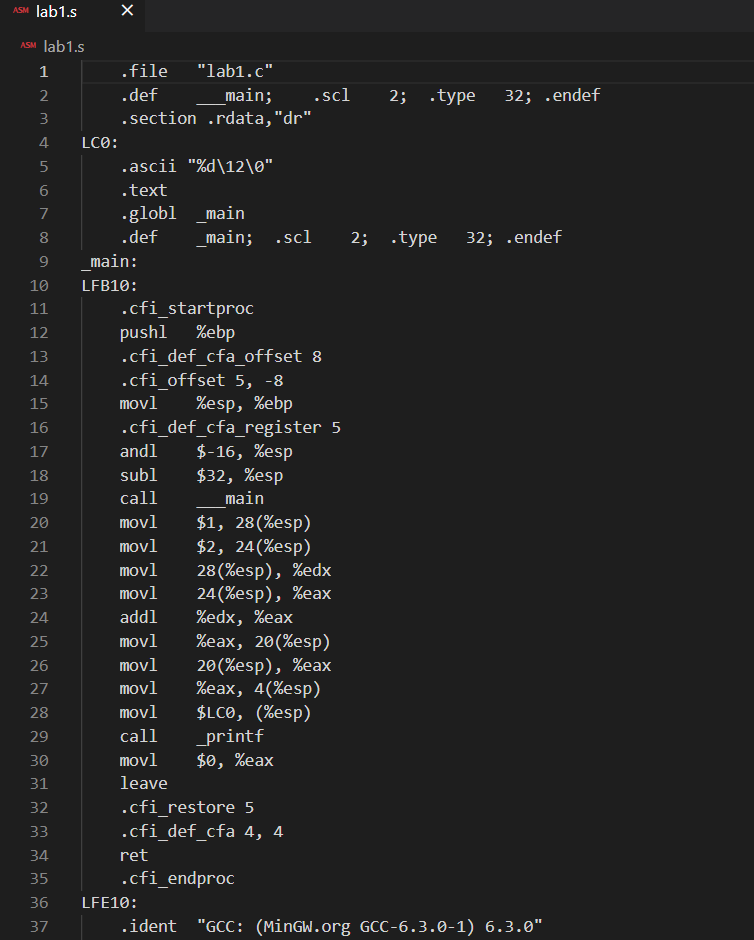


图表 7printf函数对应机器代码段

1. 源程序⽂件的内容和可执⾏⽬标⽂件的内容完全不同

预处理阶段，预处理器根据“#”字符开头的命令修改源程序，得到拓展名为.i的文件。

编译阶段，编译器将文本文件翻译成lab1.s，该文件为汇编语言程序，每条语句都描述了一条低级机器语言指令。



图表 8lab1.s部分示例

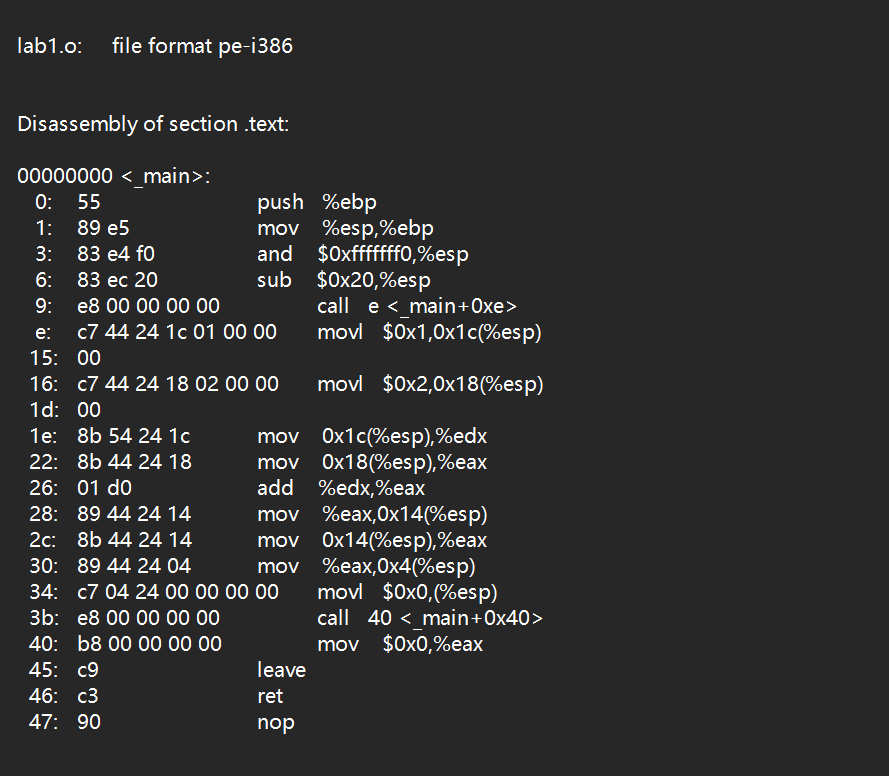
汇编阶段，汇编器将汇编语言翻译成机器语言指令，并将结果保存在二进制文件lab1.o中。

链接阶段，把标准C库中的函数（例如printf）并入文件中，生成可执行文件。

1. 不同的编译器或操作系统⽣成可执⾏⽬标⽂件

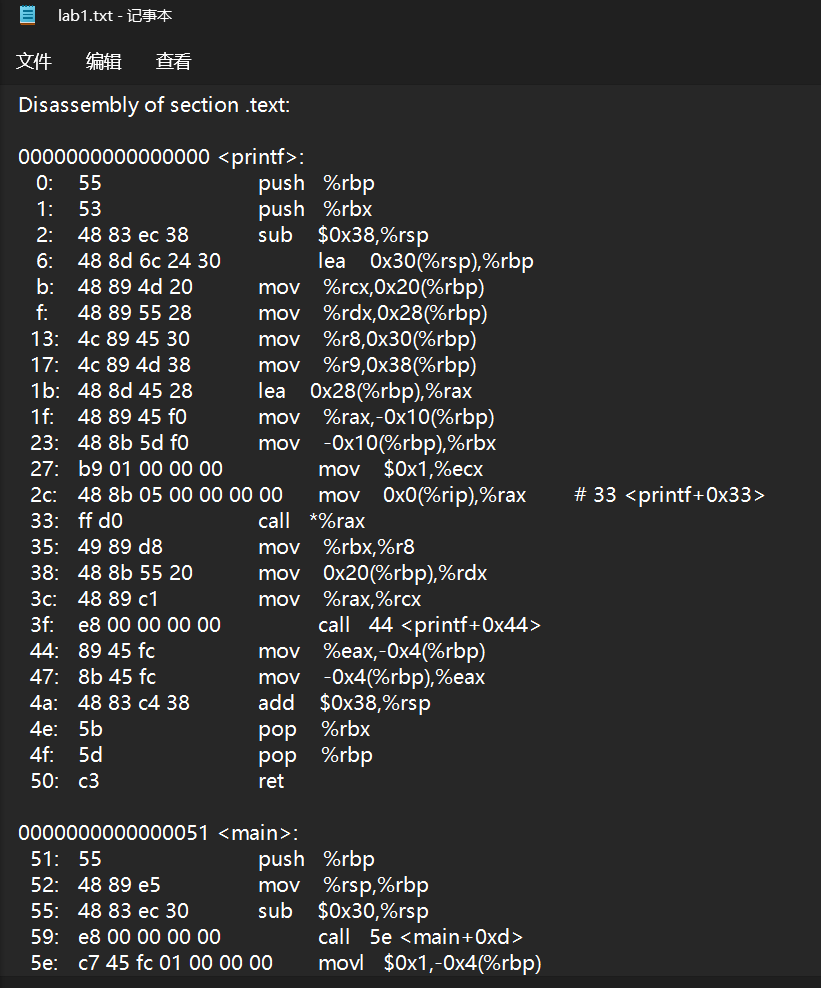
为便于比较，在此仅展示用lab1.o反汇编的主函数部分。

* 1. Win10+ GCC: (MinGW.org GCC-6.3.0-1) 6.3.0



图表 9Win10系统使用GCC: (MinGW.org GCC-6.3.0-1) 6.3.0编译

* 1. Win10+gcc (tdm64-1) 10.3.0



图表 10Win10系统使用gcc (tdm64-1) 10.3.0编译

* 1. Mac+Apple clang version 13.0.0（clang-1300.0.29.30）



图表 11MAC系统下使用Apple clang version 13.0.0（clang-1300.0.29.30）编译

1. 实验小结

高级语言编写程序后运行需要经过四个阶段：预处理阶段、编译阶段、预处理阶段，汇编阶段和链接阶段。以下用lab1.c为例分析四个阶段的作用。

预处理器根据“#”字符开头的命令修改源程序，得到拓展名为.i的文件。例如lab1.c中的#include<stdio.h>指令告诉预处理器读取系统呕吐文件内容，并插入程序文本中，得到lab1.i。

编译阶段，编译器将文本文件翻译成lab1.s，该文件为汇编语言程序，每条语句都描述了一条低级机器语言指令。不同的编译器汇编语句有略微差异，但是汇编语言是通用的计算机语言，所以不会出现较大差别，描述的指令均为相同的操作。

汇编阶段，汇编器将汇编语言翻译成机器语言指令，并将结果保存在二进制文件lab1.o中。lab1.o的字节编码是机器语言指令而不是字符，所以打开时呈现乱码。

链接阶段，把标准C库中的函数（例如printf）并入文件中，生成可执行目标文件。可执行文件加载到存储器后，由系统负责执行。