# 实训2

## 实训2题目：

## 试述用gcc对C语言程序的编译过程？

当使用GCC编译一个C语言程序时，整个编译过程大致可以分为四个主要阶段：预处理、编译、汇编和链接。

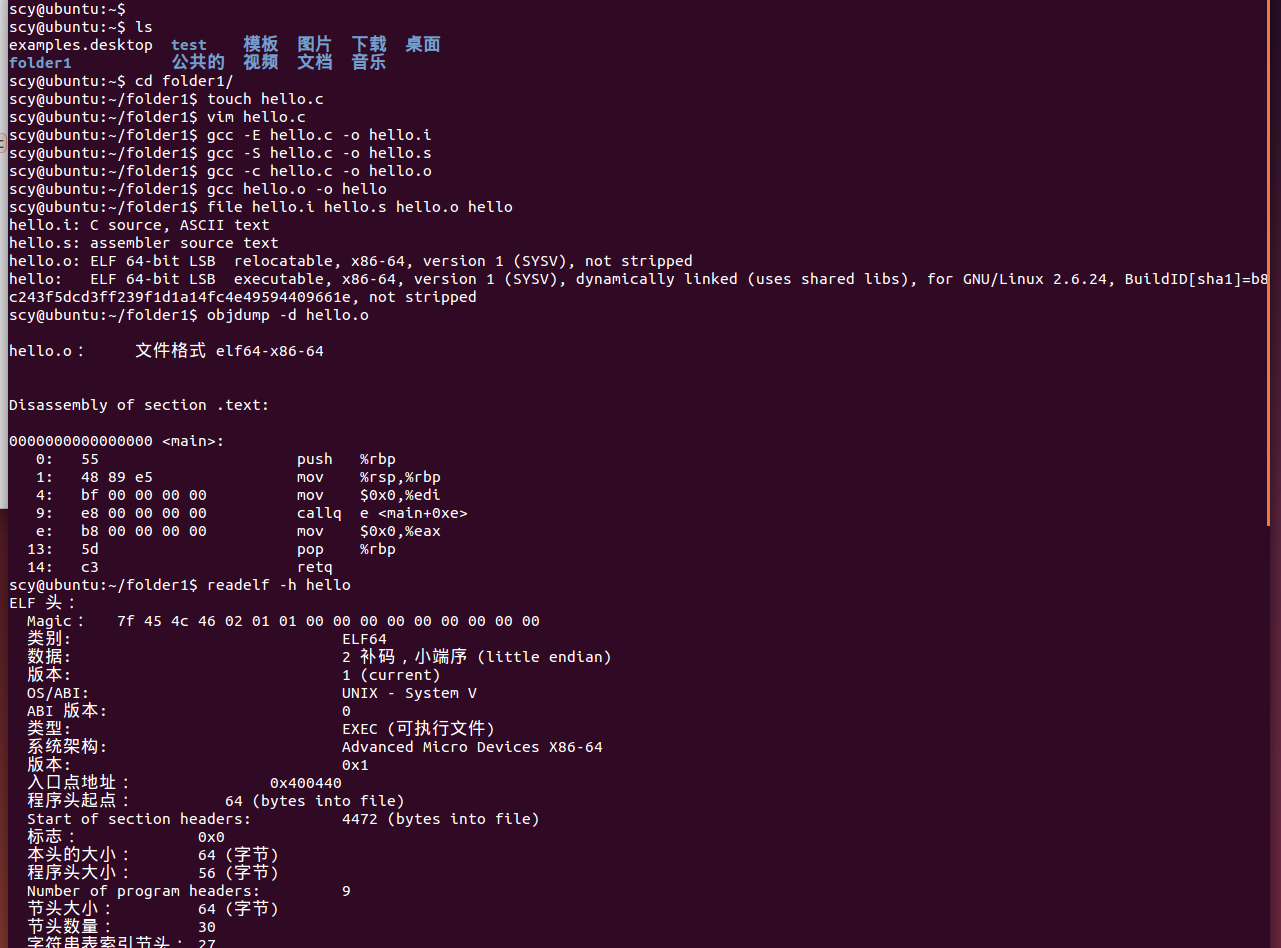
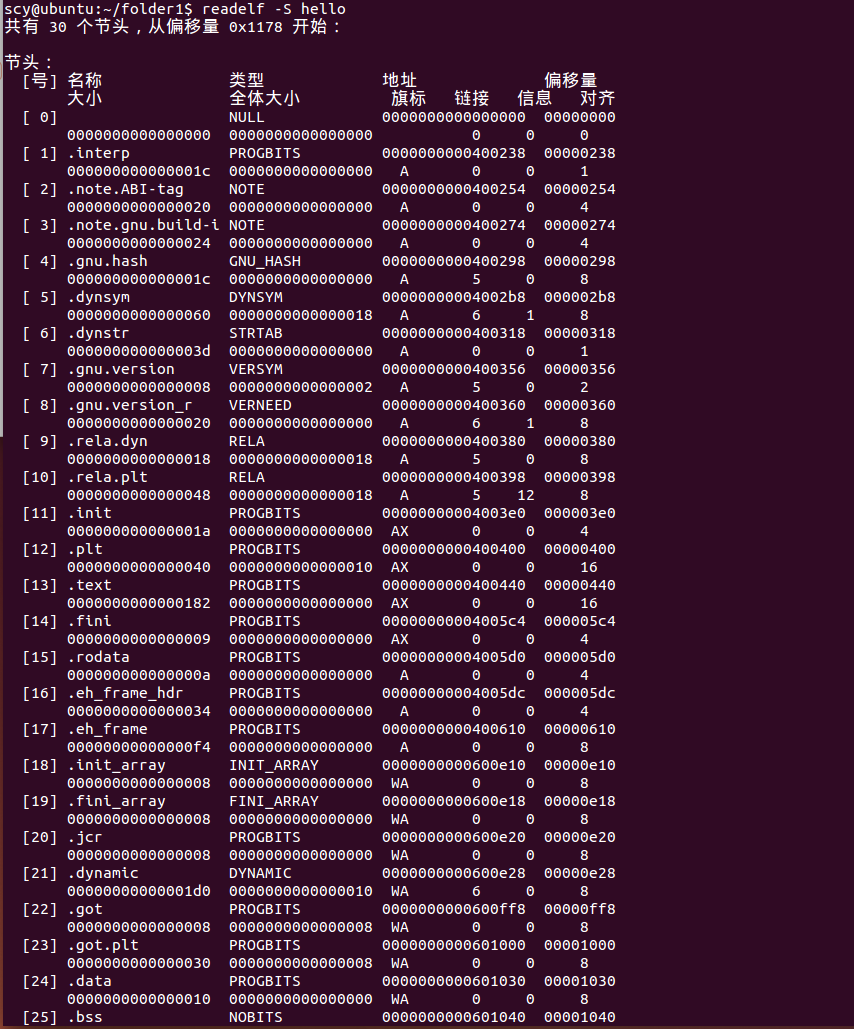
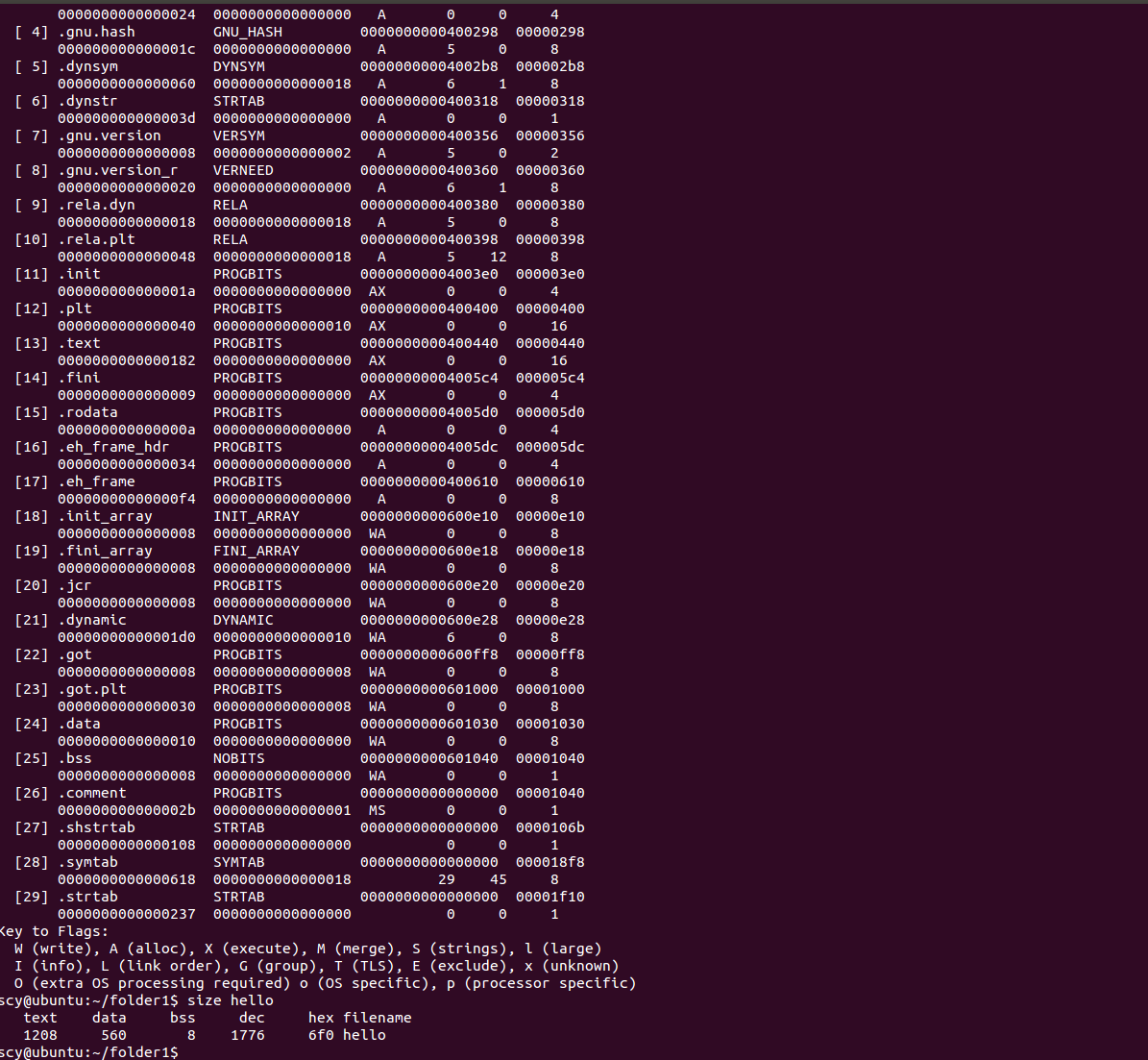
开始时，源代码经过预处理阶段，GCC会处理诸如#include这样的指令，将指定的头文件内容合并到当前源代码中；同时，它会解析并应用#define定义的宏，替换掉源代码中的宏引用，以及移除所有的注释行。这样做的结果是生成一个扩展名为.i的预处理文件，它不再包含任何宏定义或#include指令，为下一步的编译做好准备。

接下来，GCC进入编译阶段，分析预处理后的代码，将其转换为汇编语言。在这个过程中，GCC不仅检查代码的语法和语义，还进行类型检查、范围检查等静态分析，确保代码的正确性和安全性。此外，编译器会执行多种优化策略，比如常量折叠和死代码消除，以提升生成代码的执行效率。编译结束后，得到的是一个扩展名为.s的汇编语言文件。

随后，在汇编阶段，由汇编器接手，将编译产生的汇编代码转换成目标平台上的机器语言指令。这个步骤将汇编语言翻译成二进制形式的指令集和数据段，形成对象文件，其格式可能为ELF（Executable and Linkable Format）或其他平台特有的格式。

最后，链接阶段将各个独立的对象文件以及所需的库文件合并，构建出完整的可执行程序。链接器解析并解决对象文件间的符号引用，分配内存地址，生成最终的可执行文件。如果程序依赖于动态链接库，链接器还会记录这些库的信息，确保在程序运行时能够正确加载。

## 试写简单c程序，输出“hello”，在Linux下使用gcc工具分别生成.i文件，.s文件，.o文件和elf可执行文件，并对这些文件分别用file、objdump、readelf、size命令进行处理。

1. 
2. 
3. 

## 3.用gdb调试工具对如下代码进行调试：

调试函数的一系列命令，源代码如下main.c  
**#include <stdio.h>**

**int add\_range(int low, int high)  
{  
int i,sum;  
for(i=low;i<=high;i++)  
sum=sum+i;  
return sum;  
}  
int main(void)  
{  
int result[100];  
result[0]=add\_range(1,10);  
result[1]=add\_range(1,100);  
printf("result[0]=%d\nresult[1]=%d\n",result[0],result[1]);  
return 0;  
}**

**变量sum没有声明**

## 4.用gdb调试工具对如下程序进行调试，并改正该程序，gdb调试的步骤，给出截图。

**#include <stdio.h>**

**static char buff [256];**

**static char\* string;**

**int main ()**

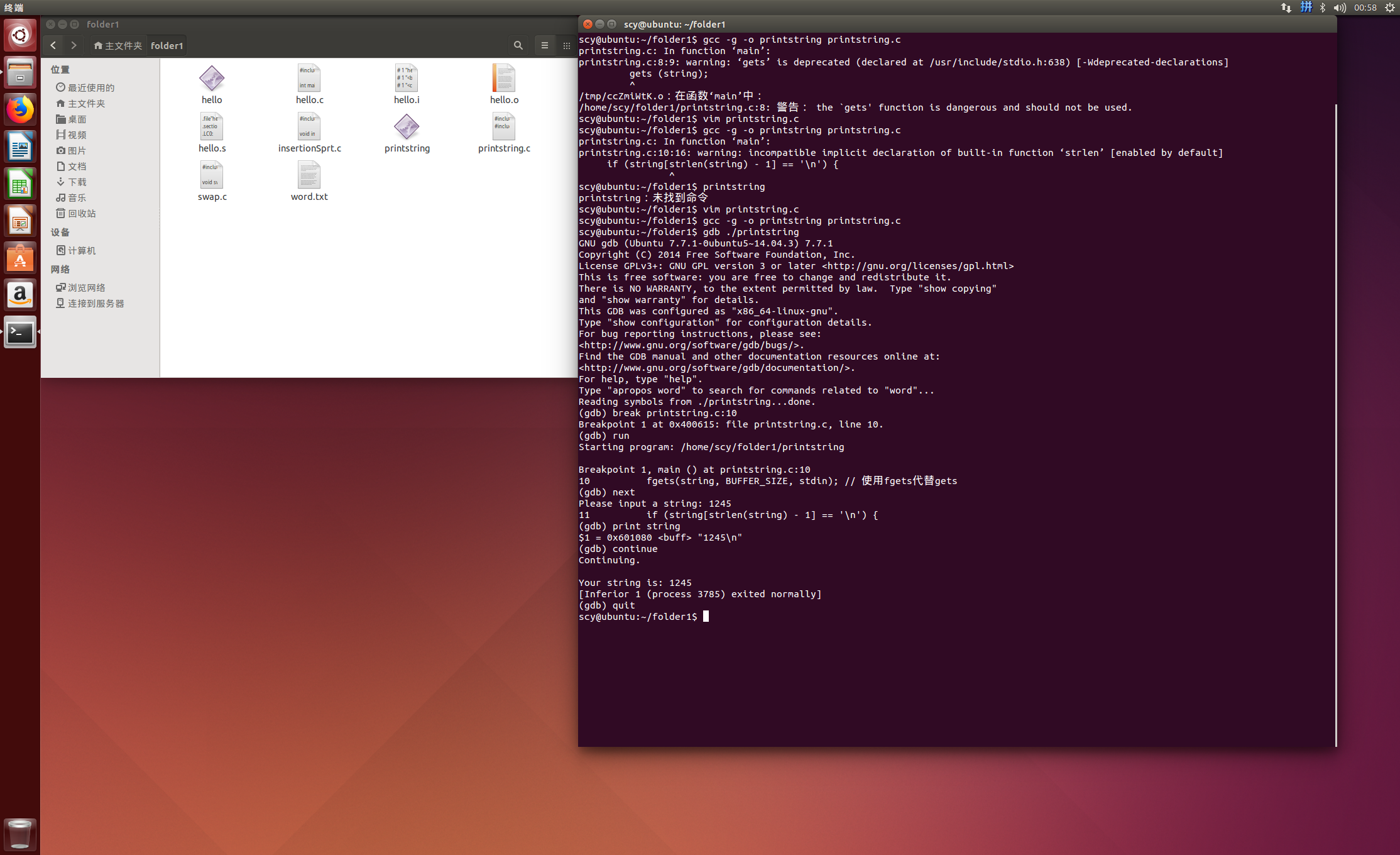
**{**

**printf ("Please input a string: ");**

**gets (string);**

**printf ("\nYour string is: %s\n", string);**

**}**



修改代码：#include <stdio.h>

#include <string.h>

#define BUFFER\_SIZE 256

static char buff[BUFFER\_SIZE];

static char\* string = buff;

int main () {

printf("Please input a string: ");

fgets(string, BUFFER\_SIZE, stdin); // 使用fgets代替gets

if (string[strlen(string) - 1] == '\n') {

string[strlen(string) - 1] = '\0'; // 移除换行符

}

printf("\nYour string is: %s\n", string);

return 0;

}