# 实训2

## 实训2题目：

## 试述用gcc对C语言程序的编译过程？

**预处理：预处理通过对宏定义(像#define）进行展开，对头文件（像 stdio.h）进行展开，对条件进行（像ifdef）编译，展开所有宏，删除所有注释（像"//"）。预处理cpp把源代码以及头文件预编成一个.i文件**

**编译：编译也就是检查语法是否错误，将预处理过的文件编译成汇编（.s）文件**

**汇编：汇编也就是将汇编（.s）文件生成目标文件（二进制文件）。通过汇编，文本代码变成了二进制代码（二进制代码文件以.o为后缀名）**

**链接：链接过程就是找到依赖的库文件（静态与动态），将目标文件链接为可执行程序**

## 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。

gcc -E hello.c -o hello.i

gcc -S hello.i -o hello.s

gcc -c hello.s -o hello.o

gcc hello.o -o hello

file hello.i

file hello.s

file hello.o

file hello

objdump -s -d main.o > main.o.txt

objdump -s -d main > main.txt

readelf -h hello.i

readelf -h hello.o

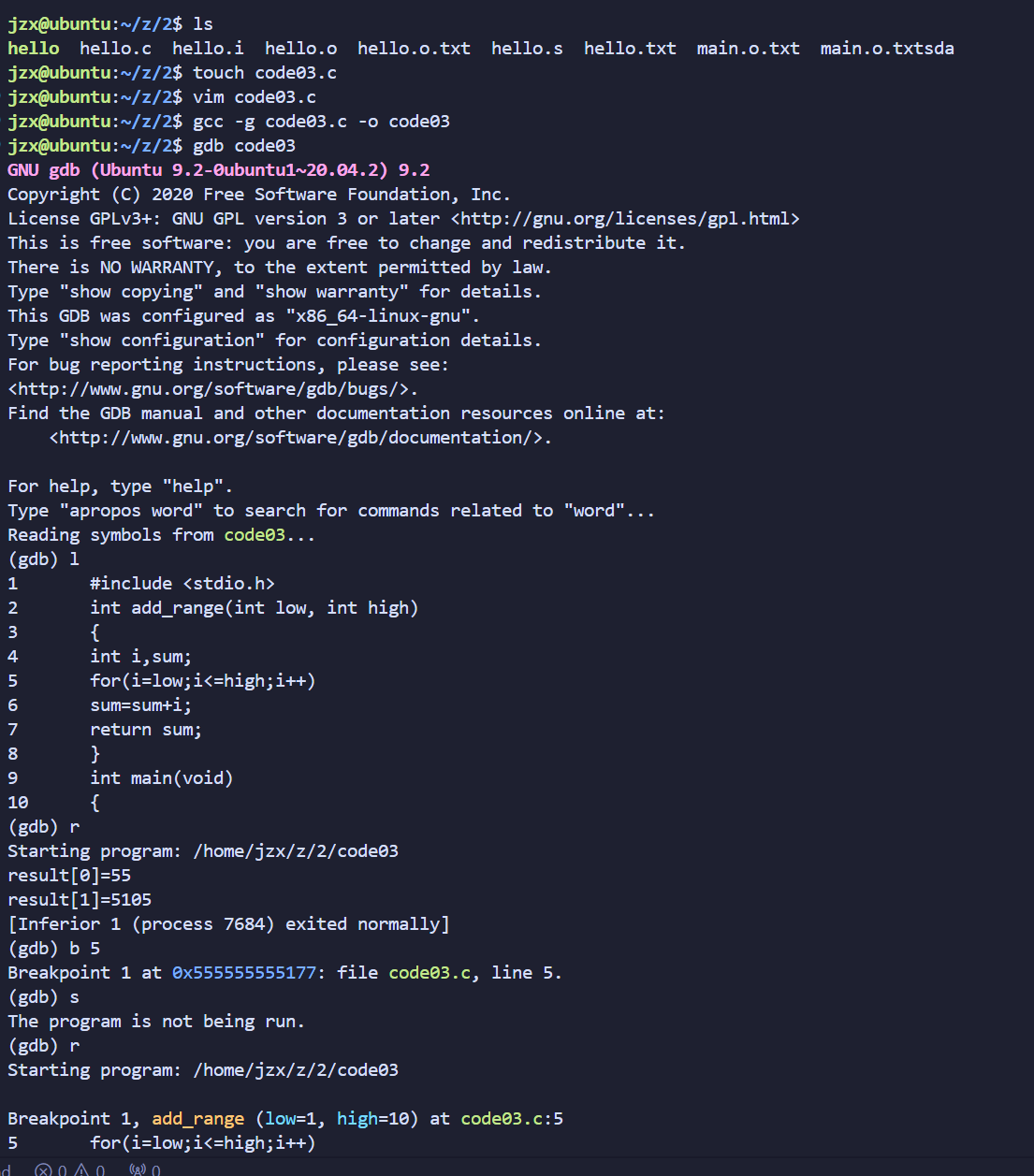
size hello.o

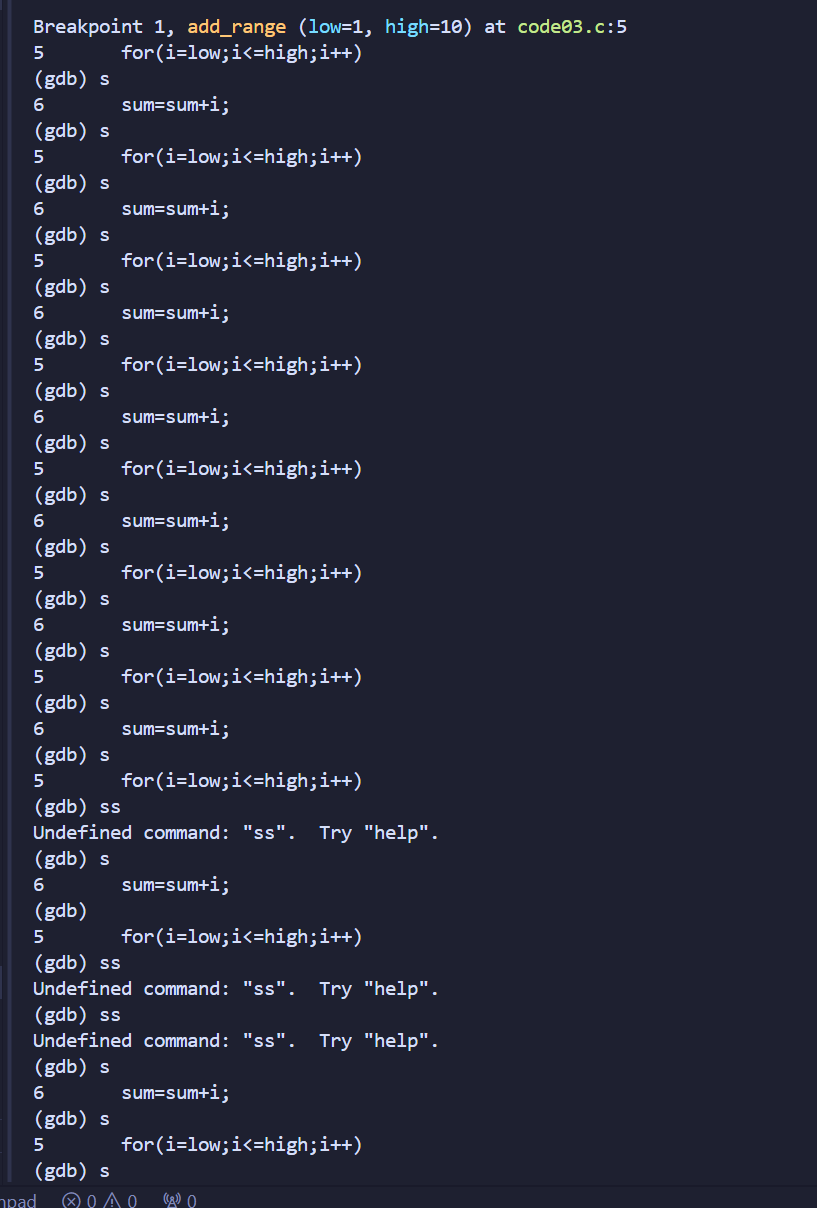
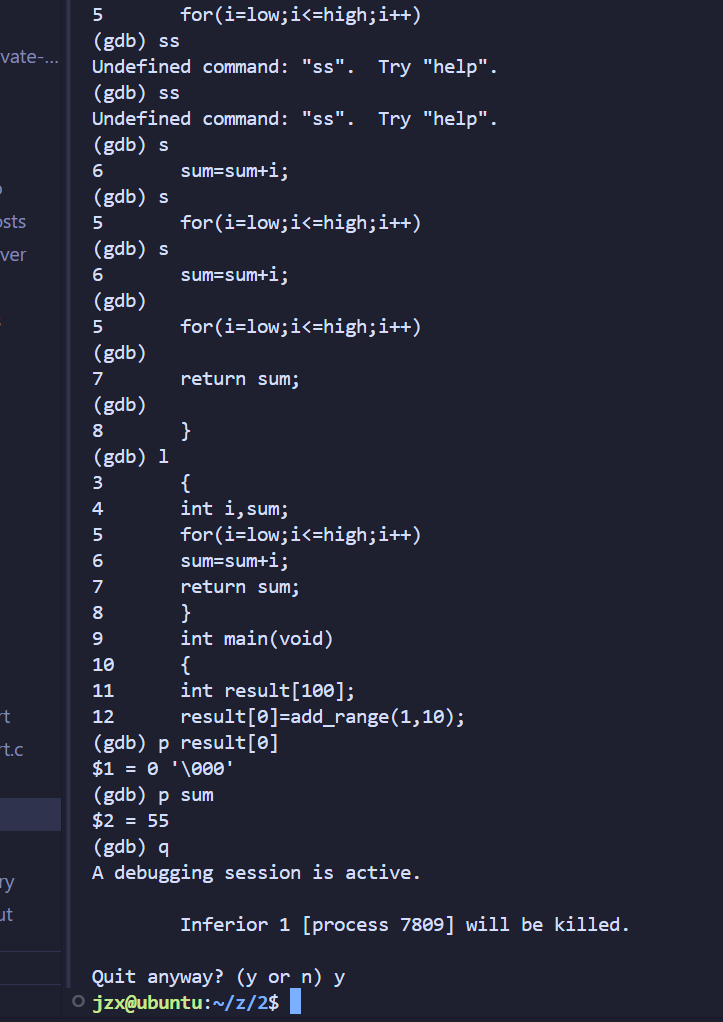
size hello

## 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;  
}**





## 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 ()**

**{**

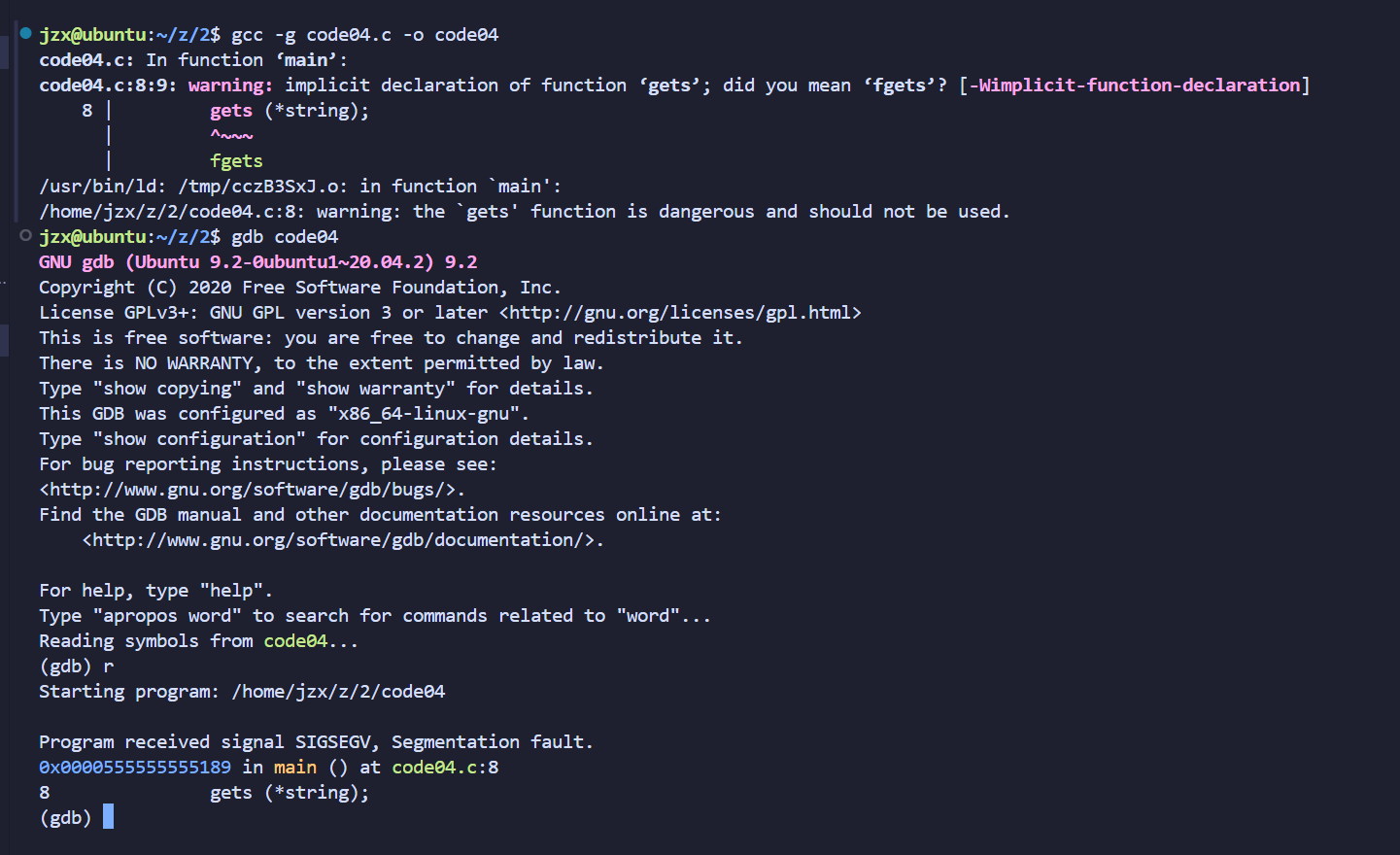
**printf ("Please input a string: ");**

**gets (string);**

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

**}**

先gcc编译有上述代码的code04.c文件，然后gdb run发现读入有问题，string是一个未初始化的指针，读的话很不安全，可能发生缓冲区溢出，（1）所以把gets替换，用安全的fgets读入，又因为并且指针指向的位置未定义，擅自读入可能会导致段错误，代码定义了一个buff数组，（2）所以改为读入buff数组



改正后的代码：

**#include <stdio.h>**

**static char buff[256];**

**static char\* string = buff;**

**int main() {**

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

**fgets(string, sizeof(buff), stdin);**

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

**return 0;**

**}**