实验题目：lab0，操作系统的编程基础

实验环境

<http://www.shiyanlou.com/courses/221>

安装linux环境，并安装gcc和gdb。

1. 了解汇编

尝试理解下面的命令

$gcc -S -m32 lab0\_ex1.c

接着我们将得到lab0\_ex1.s文件，请写出汇编代码与c代码之间的关系。

**分析:1.-S表示仅仅编译，不进行链接或者汇编**

**2.-m32表示生成32位机器的汇编代码**

**3.汇编代码与c代码之间的关系：**

**1：**[**汇编语言**](https://www.baidu.com/s?wd=%E6%B1%87%E7%BC%96%E8%AF%AD%E8%A8%80&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao)**是面向具体CPU及其硬件结构的语言**[**C语言**](https://www.baidu.com/s?wd=C%E8%AF%AD%E8%A8%80&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao)**可以做到硬件无关性  
2：**[**汇编语言**](https://www.baidu.com/s?wd=%E6%B1%87%E7%BC%96%E8%AF%AD%E8%A8%80&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao)**效率较高**[**C语言**](https://www.baidu.com/s?wd=C%E8%AF%AD%E8%A8%80&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao)**效率相对较低  
3：**[**汇编语言**](https://www.baidu.com/s?wd=%E6%B1%87%E7%BC%96%E8%AF%AD%E8%A8%80&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao)**是非**[**结构语言**](https://www.baidu.com/s?wd=%E7%BB%93%E6%9E%84%E8%AF%AD%E8%A8%80&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao)**，可读性较弱**[**C语言**](https://www.baidu.com/s?wd=C%E8%AF%AD%E8%A8%80&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao)**是**[**结构语言**](https://www.baidu.com/s?wd=%E7%BB%93%E6%9E%84%E8%AF%AD%E8%A8%80&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao)**，可读性较强  
4：  
汇编语言基本没有**[**可移植性**](https://www.baidu.com/s?wd=%E5%8F%AF%E7%A7%BB%E6%A4%8D%E6%80%A7&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao) **C言**[**可移植性**](https://www.baidu.com/s?wd=%E5%8F%AF%E7%A7%BB%E6%A4%8D%E6%80%A7&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao)**很强  
5：  
汇编语言没有函数库，所有代码需要自己组织  
C语言有丰富的函数库可供调用。**

1. 用gdb调试

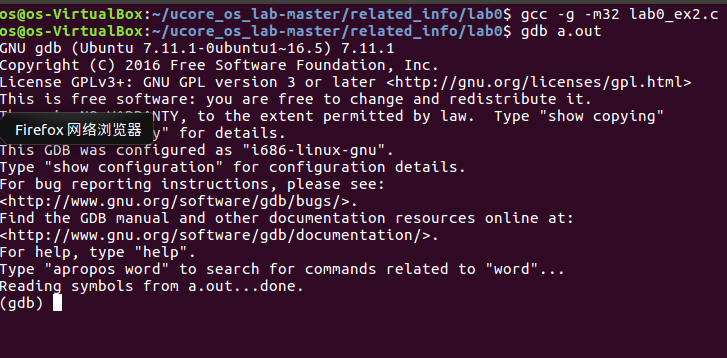
尝试下面的命令，

$gcc -g -m32 lab0\_ex2.c

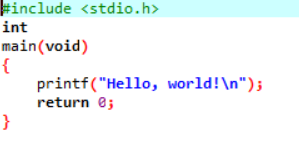
接着我们会得到a.out文件，请用gdb调试，并写出设置断点、单步执行及查看变量的过程。

**分析:1.-g表示调试**

**2.-m32表示生成32位机器的汇编代码**



**3.先用gdb 1查看代码，是Helloworld；**



**给每一行都加入断点，输入info breakpoints查看，虽然每行都有断点，但是从地址上看实际上只有三处断点：**

**按r（run）从头运行，停止在第一个断点。然后：**

**按c(continue)继续运行，直到遇见下一个断点**

**按s(step)表示执行下一个语句，会进入函数内部**

**按n(next)表示执行下一个语句，但是不会进入函数内部**

**按下s后，屏幕输出Hello,world!停止在了断点6，再按s停止在断点7反花括号那里，最后一步程序结束运行。然后按q退出。**

1. 掌握指针和类型转换相关的Ｃ编程

分析如下代码段，写出gintr和intr的结果，试着编译这段代码，如果遇到错误进行改正，并分析错误原因。

**分析：输出结果为intr is 0x10002**

**Gintr is 0xee0000010002**

**编译以后出现错误，报错以后把强制类型转换成如下形式：\*（long long unsigned \*）&gintr 即可。**

**解释：题目意思应该是输出更改后的gintr这个结构体变量的内容，所以首先取gintr这个变量的地址，强制转换为指向llu型变量的指针，再引用这个地址就得到了llu型的变量。**

4. 掌握通用链表结构相关的Ｃ编程

查看list.h和lab0\_ex4.c，编写一个程序，利用list.h中的链表结构，将26个英文字母存入链表中，并逆序打印出来。

将1-8换成26个字母即可