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。

它们的指令集，[寻址方式](https://m.sogou.com/web/searchList.jsp?keyword=%E5%AF%BB%E5%9D%80%E6%96%B9%E5%BC%8F&insite=wenwen.sogou.com&pid=sogou-waps-6954f0b6918c1bcf&rcer=g9PEAOFXxfOnjdRMR)不同，相当于两个机器各有各的操作方式 汇编语言是低级语言，在编写程序的时候会把根据不同的情况指定使用不同的寻址方式，能够对内存和CPU里的[通用寄存器](https://m.sogou.com/web/searchList.jsp?keyword=%E9%80%9A%E7%94%A8%E5%AF%84%E5%AD%98%E5%99%A8&insite=wenwen.sogou.com&pid=sogou-waps-6954f0b6918c1bcf&rcer=g9PEAOFXxfOnjdRMR)直接操纵。不同的计算机系列会有不同的汇编语言 而C语言是[高级语言](https://m.sogou.com/web/searchList.jsp?keyword=%E9%AB%98%E7%BA%A7%E8%AF%AD%E8%A8%80&insite=wenwen.sogou.com&pid=sogou-waps-6954f0b6918c1bcf&rcer=g9PEAOFXxfOnjdRMR)，可以实现跨平台。编写程序无需指定CPU的指令执行情况，一段程序可以在不同体系结构的计算机上使用

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。

设置断点：（gdb）file a.out

（gdb）b 3

Break point 1 at 0x8048426: file lab0\_ex2.c,line 3

单步执行:（gdb）r

（gdb）n

Hello,world!

1. return 0;

查看变量:（gdb）watch printf

（gdb）c

Hello,world!

1. 掌握指针和类型转换相关的Ｃ编程

分析如下代码段，

#include <stdio.h>

#define STS\_IG32 0xE // 32-bit Interrupt Gate

#define STS\_TG32 0xF // 32-bit Trap Gate

typedef unsigned uint32\_t;

#define SETGATE(gate, istrap, sel, off, dpl) { \

(gate).gd\_off\_15\_0 = (uint32\_t)(off) & 0xffff; \

(gate).gd\_ss = (sel); \

(gate).gd\_args = 0; \

(gate).gd\_rsv1 = 0; \

(gate).gd\_type = (istrap) ? STS\_TG32 : STS\_IG32; \

(gate).gd\_s = 0; \

(gate).gd\_dpl = (dpl); \

(gate).gd\_p = 1; \

(gate).gd\_off\_31\_16 = (uint32\_t)(off) >> 16; \

}

/\* Gate descriptors for interrupts and traps \*/

struct gatedesc {

unsigned gd\_off\_15\_0 : 16; // low 16 bits of offset in segment

unsigned gd\_ss : 16; // segment selector

unsigned gd\_args : 5; // # args, 0 for interrupt/trap gates

unsigned gd\_rsv1 : 3; // reserved(should be zero I guess)

unsigned gd\_type : 4; // type(STS\_{TG,IG32,TG32})

unsigned gd\_s : 1; // must be 0 (system)

unsigned gd\_dpl : 2; // descriptor(meaning new) privilege level

unsigned gd\_p : 1; // Present

unsigned gd\_off\_31\_16 : 16; // high bits of offset in segment

};

int

main(void)

{

unsigned before;

unsigned intr;

unsigned after;

struct gatedesc gintr;

intr=8;

before=after=0;

gintr=\*((struct gatedesc \*)&intr);

SETGATE(gintr, 0,1,2,3);

intr=\*(unsigned \*)&(gintr);

printf("intr is 0x%x\n",intr);

printf("intr is 0x%llx\n", gintr);

return 0;

}

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。

intr is 0x10002

gintr is 0xee0000010002

因为gintr不是long long unsigned int型的，所以printf语句中的gintr应该改为

\*（long long unsigned int \*）&（gintr）

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。

