实验 2.1 Linux 下 C 编程

一、 实验目的

1、练习并掌握 Linux 提供的 vi 编辑器来编译 C 程序。

2、了解 Linux C 库函数的使用方法，掌握常用函数来解决实际问题。

3、学会利用 gcc、gdb 编译、调试 C 程序。

4、了解静态链接、动态链接的过程及原理。

二、实验内容

1、 设计一个程序，生成 50 个 100-1000 之间的随机数。

2、 设计一个猜数游戏的程序，先产生一个随机数，要求被试输入一个数，计算机会提示猜大了，猜小了或恭喜您猜中了，直到猜中，退出程序。

3、 设计一个程序，要求输入两个整数，求和输出。通过使用 gcc 的参数，控制 gcc 的编译过程，了解 gcc 的编译过程，进一步认识 gcc 的灵活性。

4、 设计一个程序, 要求把输入的字符串原样输出, 程序中的头文件自己定义，源程序文件为“1-4.c”，自定义的头文件为“my.h”。

5、 设计一个程序，要求把输入的数字作为 X 轴坐标，算出它的 sin 值。

6、 编写程序 main.c，实现格式为 a+b=c 格式的输出。

（1） 使用 printf 函数实现输出，如 printf（“%d+%d=%d”，a，b，a+b）。

（2） 实现 main 命令行参数的获得，其中第一个参数是 a，第二个参数是 b；定义 add 函数 int add（int p1，int p2）。

（3） 实现 add 函数，并于 printf 中采用 printf（“%d+%d=%d”, a, b, add(a+b)）。基于以上代码，分别进行全静态和全动态的编译。

（4） 执行 objdump -x 命令，查看包含 NEEDED 的字段，分析生成的两种执行

文件的依赖库的不同。

（5） 执行 file 和 ld 命令，查看两个执行文件的不同。

7、 分别编写 add.h、add.c、main.c 文件，其中 add.h 中包含加法函数的定义 int add（int p1，int p2），add.c 文件中包含加法函数的实现。

（1） 对 add.c 进行编译，生成目标文件 add.o。

（2） 执行 ar 命令，生成 libadd.a。

（3） 编写 main.c 文件，通过 include add.h 头文件和 gcc 编译，实现对 libadd.a 中加法函数的调用。

8、 分别编写 add.h、add.c、main.c 文件，其中 add.h 中包含加法函数的定义 int add（int p1，int p2），add.c 文件中包含加法函数的实现，对 add.c 进行编译生成共享库 libadd.so。

编写 main.c 文件，通过 include add.h 头文件和 gcc 编译，实现对 libadd.so 中加法函数的调用。

9、 针对题 8 的共享库，以至少三种方式实现共享库的加载运行，针对每种方式编写一个 run.sh 来实现运行。

10、 基于题 8 的共享库重新编写 add 的实现，完成针对“按位与”操作，运行题 8 生成的可执行文件 main，使其加载修改成的 libadd.so，显示“按位与”的结果，如 1&0=0，1&4=0。