dup() 和 dup2() 是两个非常有用的系统调用，都是用来复制一个文件的描述符

# **所需头文件**

#include <unistd.h>

# int dup(int oldfd);

功能：

通过 oldfd 复制出一个新的文件描述符

参数：

oldfd: 需要复制的文件描述符 oldfd

返回值：

成功：新文件描述符，该文件描述符是系统自动分配的最小可用的文件描述符  
失败：-1

# dup实例:

下面的例子为，打开一个文件得到文件描述符，并复制该文件描述符，通过这 2 个描述符分别对此文件进行写操作：

#include <stdio.h>

#include <unistd.h>

#include <stdlib.h>

#include <sys/types.h>

#include <sys/stat.h>

#include <fcntl.h>

#include <string.h>

int main(int argc, char \*argv[])

{

    int fd1 = 0;

int fd2 = 0;

    // 打开文件

    fd1 = open("test.txt", O\_CREAT | O\_WRONLY, 0666);

    if (fd1 < 0)

    {

        perror("open");

        exit(-1);

    }

printf("fd1 ============== %d\n", fd1);

    // 通过 fd1 复制出 fd2, 最终 fd1, fd2 都指向 test.txt

    fd2 = dup(fd1);

printf("fd2 ============== %d\n", fd2);

    char \*buf1 = "this is a test for fd1\n";

    // 操作 fd1 文件描述符

write(fd1, buf1, strlen(buf1));

    char \*buf2 = "this is a test for fd2\n";

    // 操作 fd2 文件描述符

write(fd2, buf2, strlen(buf2));

    // 关闭文件描述符，两个都得关

    close(fd1);

close(fd2);

    return 0;

}

接下来，我们再写一个例子，把本来通过 printf() 显示到屏幕上的内容，不显示在屏幕上，而让这些内容写入一个文件里：

#include <stdio.h>

#include <unistd.h>

#include <stdlib.h>

#include <sys/types.h>

#include <sys/stat.h>

#include <fcntl.h>

int main(int argc, char \*argv[])

{

    int fd1;

int fd2;

    // 打开文件

    fd1 = open("test.txt", O\_CREAT | O\_WRONLY, 0666);

    if (fd1 < 0)

    {

        perror("open");

        exit(-1);

    }

printf("fd1 ============== %d\n", fd1);

    // 1 本来指向标准输出设备（如，显示屏）

    // 把 1 文件描述符关闭，就是说 1 不再指向标准输出设备

    // 这样的话，1 文件描述符就空闲了，它就成为最小可用的文件描述符

close(1);

    // 通过 fd1 复制出 fd2, 最终 fd1, fd2 都指向 “1.txt”

    // 系统会给 fd2 分配一个最小可用的文件描述符 1，即 fd2 = 1

fd2 = dup(fd1);

    // 下面这句话的内容不会打印到屏幕上，而会写到文件 “1.txt” 里

    // printf()是标准库函数，最终还是会调用系统调用函数write()

    // 相当于这样，write(1，)，往 1 文件描述符写内容，

    // 默认的情况下, 1 文件描述符指向标准输出设备（如，显示屏）

    // 所以， printf()的内容先显示到屏幕上

    // 但是现在 1 文件描述符指向文件 “1.txt”

    // 所以，printf()的内容会写入文件 “1.txt”

printf("fd2 ============== %d\n", fd2);

    close(fd1);

close(fd2);

    return 0;

}

# int dup2(int oldfd, int newfd);

功能：

通过 oldfd 复制出一个新的文件描述符 newfd

参数：

oldfd: 需要复制的文件描述符  
newfd: 新的文件描述符，这个描述符可以人为指定一个合法数字（0-1023），如果指定的数子已经被占用（和某个文件有关联），此函数会自动关闭 close() 断开这个数字和某个文件的关联，再来使用这个合法数字。

返回值：

成功：返回 newfd  
失败：返回 -1