电子科技大学

University of Electronic Science and Technology of China

Linux操作系统编程

实验四 | 进程控制实验

主讲老师：杨珊

实验四 | 进程控制实验目的

目的一：掌握Linux系统创建进程的方法

目的二：掌握在代码中如何区别父子进程的方法

目的三：掌握父子进程之间的资源共享与异同

目的四：掌握等待子进程执行结束的方法

目的五：掌握在进程中执行另外一个可执行文件的方法

实验四 | 实验原理 | 创建子进程

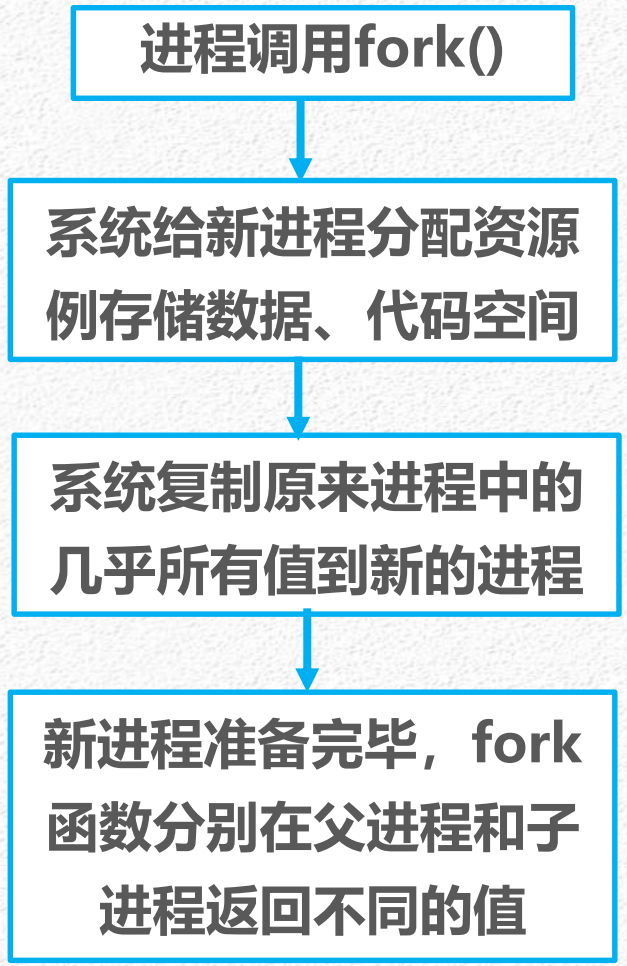
头文件： unistd.h

函数： pid_t fork(void)

作用： 创建一个子进程

返回值：

- ✓该函数会在父进程和子进程**同时返回**
- ✓返回0，当前在子进程中（0非子进程的ID）
- ✓返回子进程ID，当前在父进程中。
- ✓出错返回-1



实验四｜创建子进程示例

```
int main(void){
    pid_t pid;
    pid=fork();
    if(pid==-1)
        printf("fork error\n");
    else if(pid==0){
        printf("the returned value is %d\n",pid);
        printf("in child process!!\n");
        printf("My PID is %d\n",getpid();}
    else{
        printf("the returned value is %d\n",pid);
        printf("in father process!!\n");
        printf("My PID is %d\n",getpid();}
    return 0;
}
```

将左边代码编译为progress

progress运行结果

实验四｜fork创建父子进程比较

父子进程相同之处
真实用户ID， 真实组ID
有效用户ID， 有效组ID
当前工作目录
文件模式掩码
信号量掩码
环境变量
堆
栈
文件表（打开的文件）

父子进程差异之处
Fork的返回值
进程ID
父进程ID
子进程的 tms_utime, tms_stime , tms_cutime, tms_ustime值被设置为 0
文件锁

实验四 | 实验原理 | fork用法

情况一：父进程希望复制自己（共享代码，拷贝数据空间），但由父子进程执行代码中不同的分支

- 网络服务程序中，父进程等待客户端的服务请求，当请求达到时，父进程调用fork，使子进程处理该次请求，而父进程继续等待下一个服务请求到达

情况二：父子进程执行不同的可执行文件（具有完全不同的代码段和数据空间）。子进程使用exec系列函数执行另外的可执行文件。

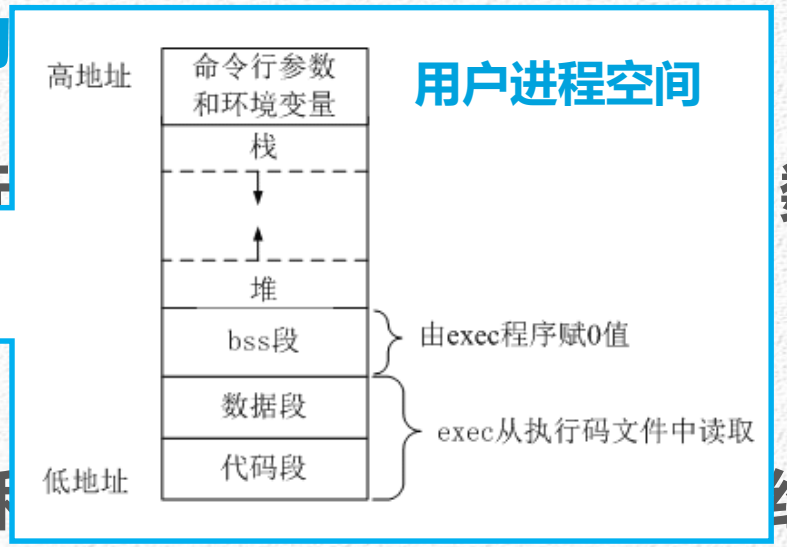
实验四｜实验原理｜exec系列函数

子进程使用exec系列函数执行另外的

exec系列函数会替换当前进程（执行

数据段、

堆和栈（来源于加载的可执行文件）



- ✓ 只替换不新建，调用exec的进程

经打开的

文件描述符都不改变

- ✓ 从新开始，exec让进程从所加载的可执行文件的main开始执行

实验四｜实验原理｜创建子进程

头文件： unistd.h

函数： int execl(const char *pathname,
const char *arg0, .../*(char*)0*/);

作用： 替换当前进程的资源，并从加载文件的main函

返回值：

- ✓ 成功该函数不返回
- ✓ 出错返回-1

可变参数：
此处是要执行程序的
命令行参数
以(char *)0结束

实验四｜实验原理｜演示程序

```
int main(void)
{
    printf("entering main process---\n");
    if(fork()==0){
    execl("/bin/ls","ls","-l",NULL);
    printf("exiting main process ----\n");}
    return 0;
}
```

运行结果

```
[zxy@test unixenv_c]$ cc execl.c
[zxy@test unixenv_c]$ ./a.out
entering main process---
total 104
-rwxrwxr-x. 1 zxy zxy       5976 Jul 12 22:54 a.out
-rw-r--r--. 1 zxy zxy        527 Jul 12 15:48 atexit02.c
-rw-r--r--. 1 zxy zxy        426 Jul 12 15:59 atexit03.c
-rw-r--r--. 1 zxy zxy        287 Jul 12 15:44 atexit.c
-rw-r--r--. 1 zxy zxy        472 Jul 10 12:39 creathole.c
```

实验四 | 实验原理 | 进程终止

程序正常终止：

- ✓ 从main返回
- ✓ 调用exit函数
- ✓ 调用_exit函数
- ✓ 最后一个线程从其启动例程中返回
- ✓ 最后一个线程调用pthread_exit

程序异常终止：

- ✓ 调用abort，产生SIGABRT信号
- ✓ 接收到终止信号

实验四｜实验原理｜等待子进程

头文件： sys/wait.h

函数： pid_t waitpid(pid_t pid, int *statloc, int options)

作用： 等待某个特定子进程

返回值：

- ✓ 成功返回子进程ID
- ✓ 出错返回-1

pid：等待进程ID，程执行终止

statloc：存放子进程终止

options：可以为0，也可以是WNOHANG（如果没有任何已经终止的子进程则马上返回，函数不等待，此时返回值为0）

实验四 | waitpid代码示例

```
pid_t pid_child, pid_return;
pid_child =fork();
if(pid_child <0) printf("Error occured
   on forking.\n");
else if(pid_child ==0)
{            sleep(10);
             exit(0);}
   do{ pid_return =waitpid(pid_child,
                 NULL, WNOHANG);
      if(pid_return ==0){
        printf("No child exited\n");
        sleep(1);}}
   while(pid_return ==0);
   if(pid_return == pid_child)
      printf("successfullyget child
              %d\n", pid_return);
   else  printf("some error
               occured\n");
```

运行结果

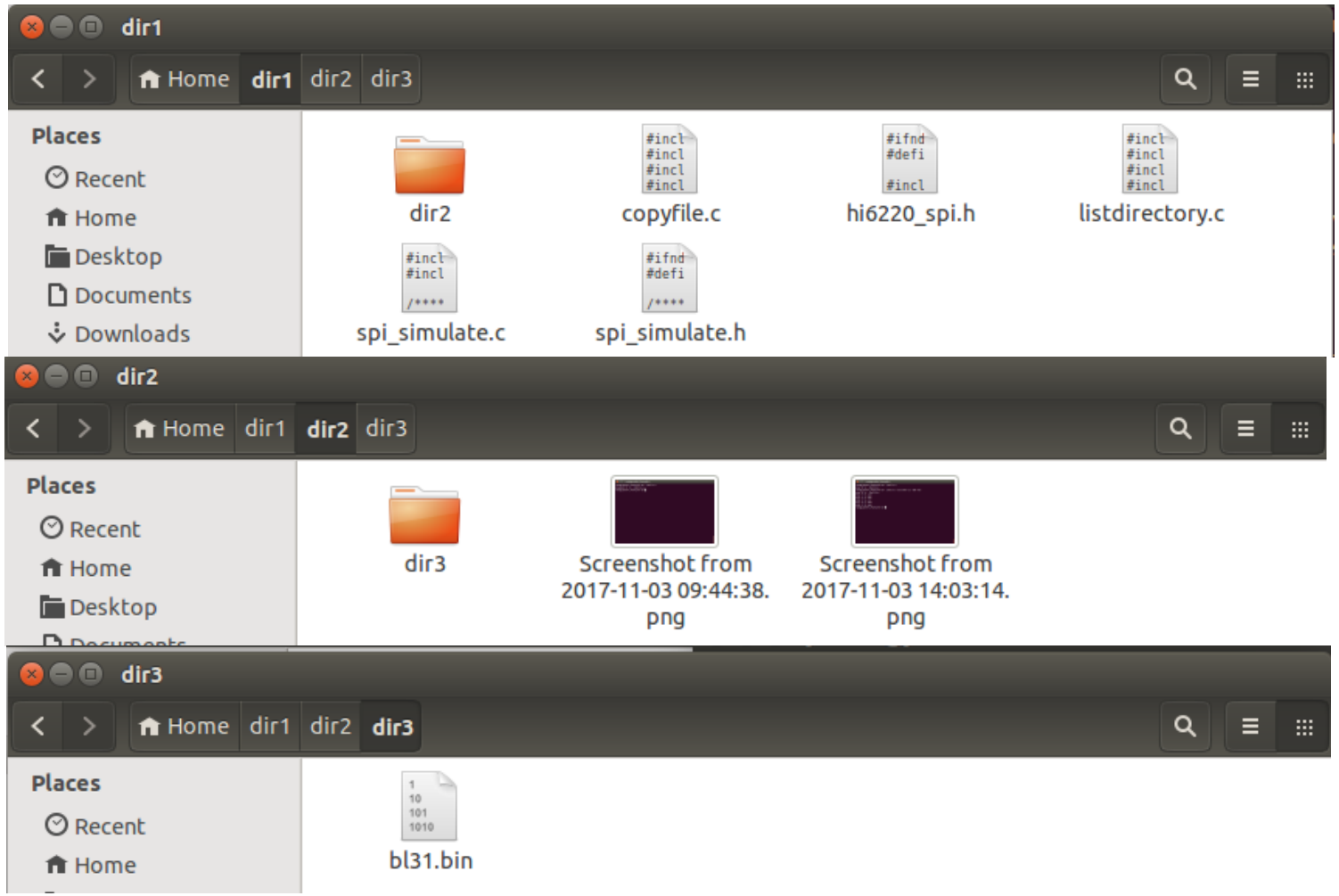
```
[devis@gdmss eight]$ ./waitpid
No child exited
No child exited
No child exited
No child exited
No child exited
No child exited
No child exited
No child exited
No child exited
No child exited
No child exited
No child exited
successfully get child 1851
```

实验四 | 进程控制实验内容

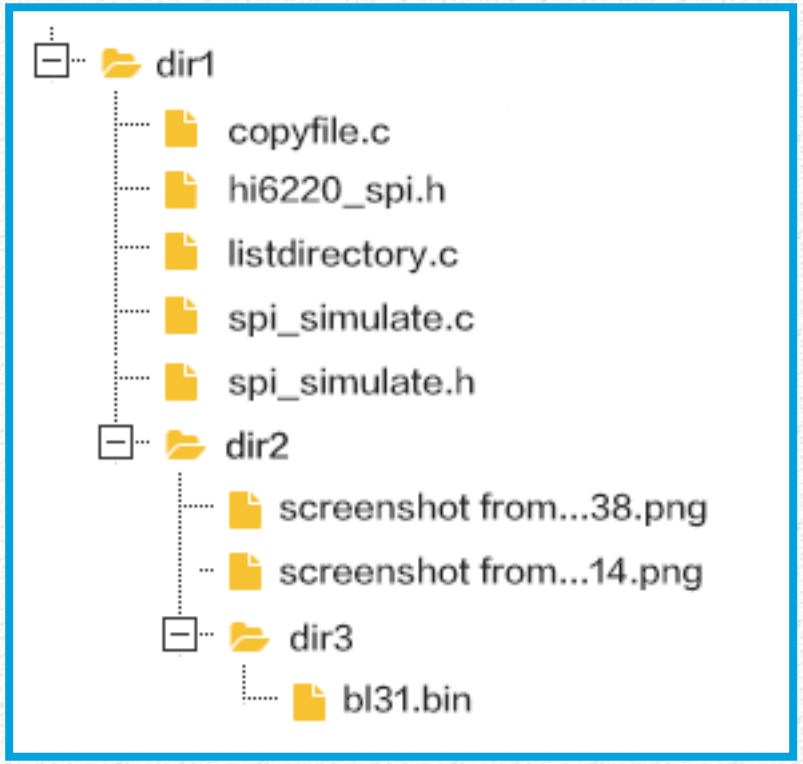
基于已经实现的实验二文件拷贝（mycp）以及实验三目录遍历（myls）的内容：

1. 改造myls程序作为父进程，其在遍历目录时，对非目录文件创建子进程运行mycp程序。
2. mycp源文件路径是父进程myls遍历所获取的文件的路径名（通过命令行参数传递给子进程mycp），并将源文件拷贝到指定目录下（在/home目录下以自己的名字的汉语拼音创建一个目录）。
3. 父进程myls等待子进程mycp运行结束，回收其内核空间资源，直到父进程遍历完成。

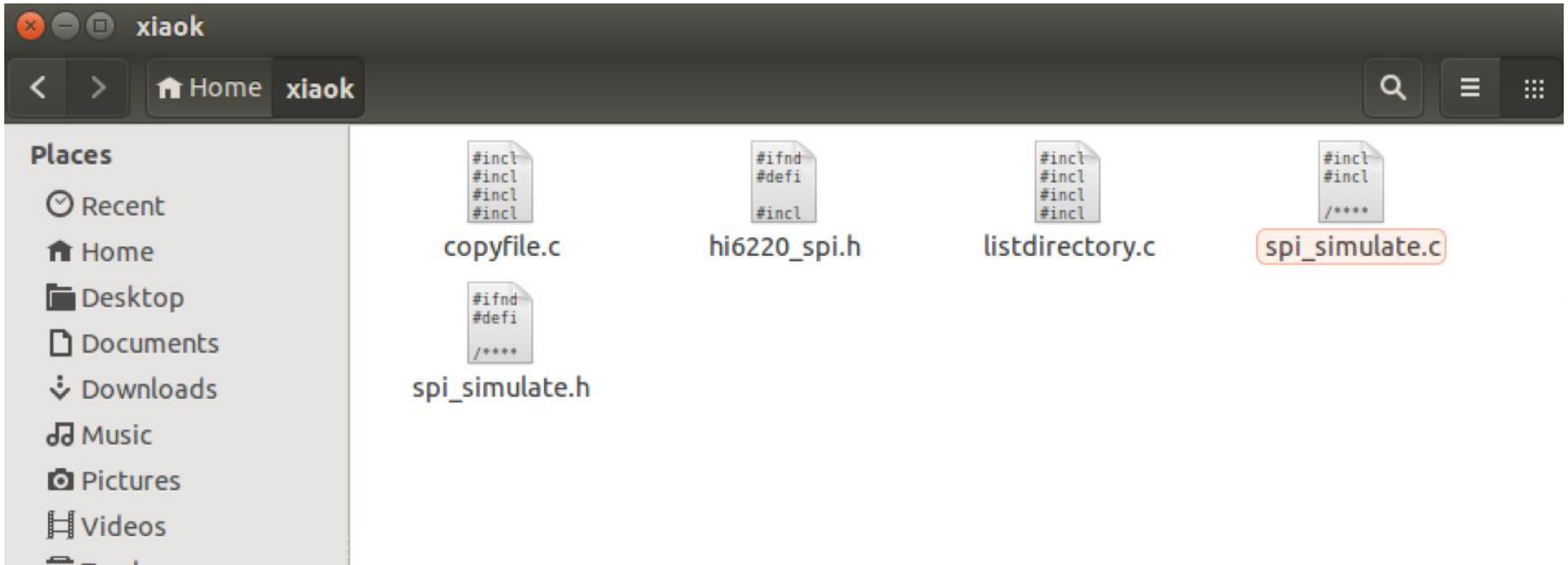
实验四 | 源文件目录结构



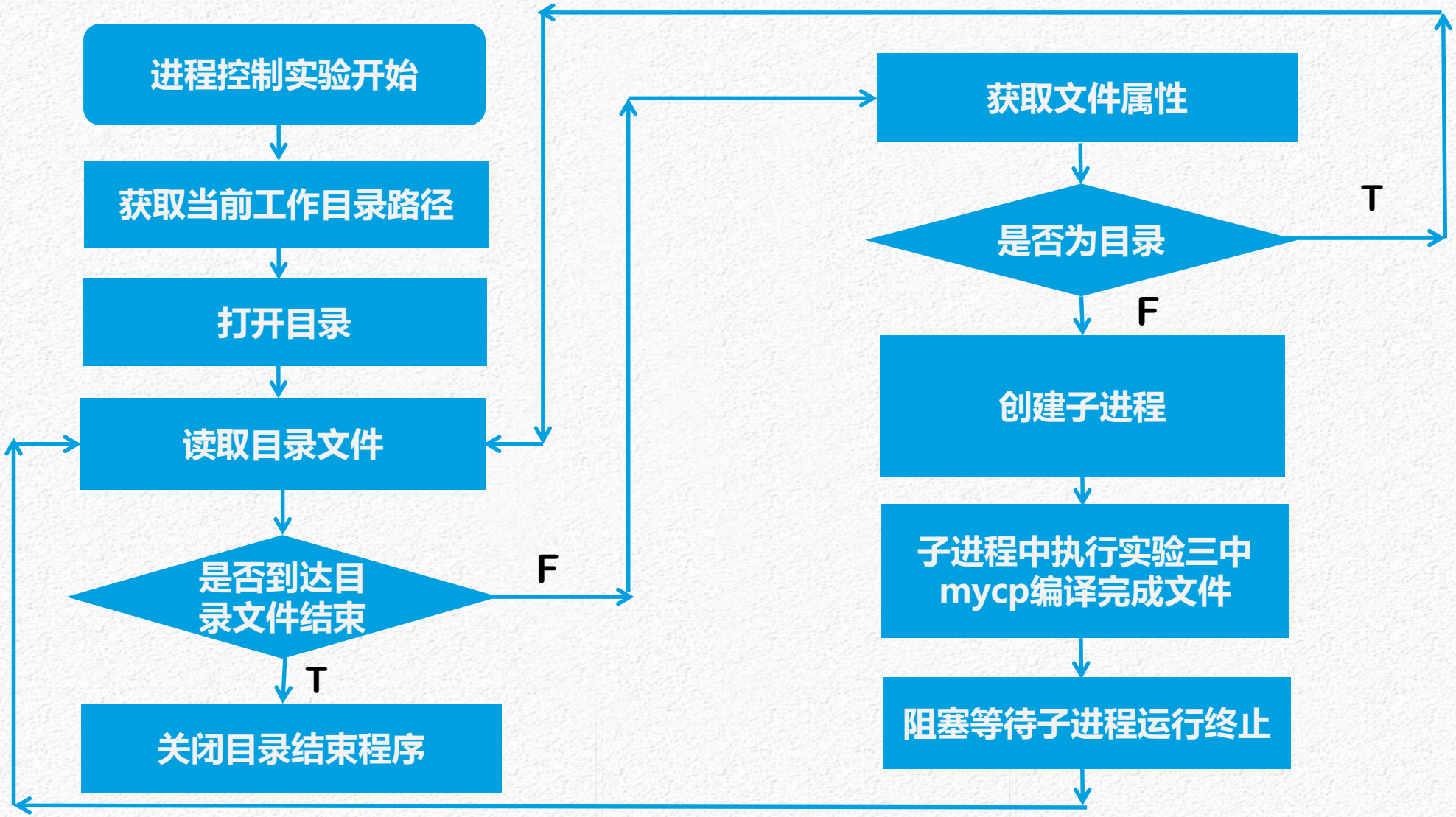
源目录结构如下所示：



实验四 | 实现对非目录文件拷贝



实验四 | 程序流程

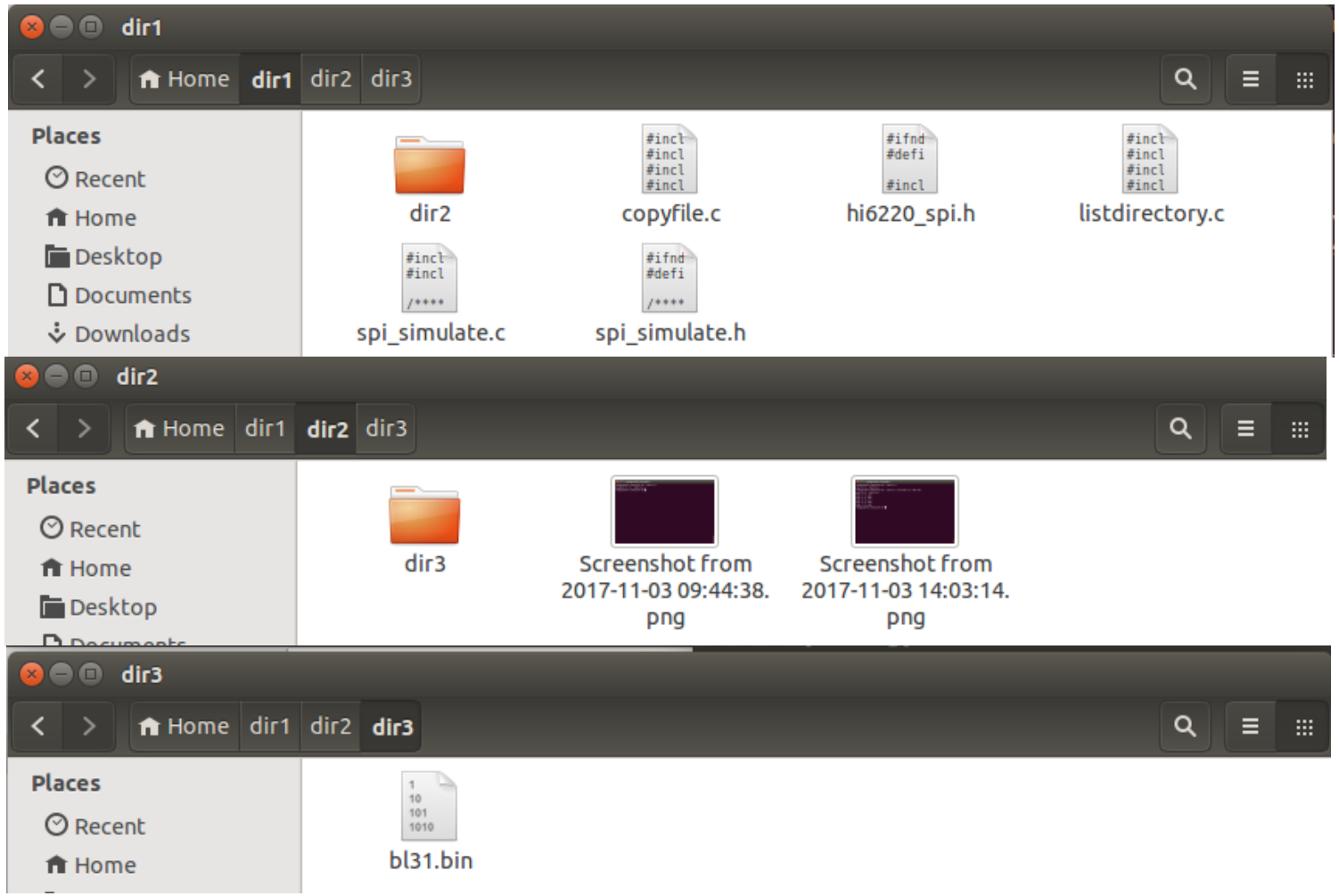

进程控制实验开始
获取当前工作目录路径
打开目录
读取目录文件
是否到达目录文件结束
F
T
关闭目录结束程序
获取文件属性
是否为目录
T
F
创建子进程
子进程中执行实验三中mycp编译完成文件
阻塞等待子进程运行终止

实验四｜进程控制实验内容扩展

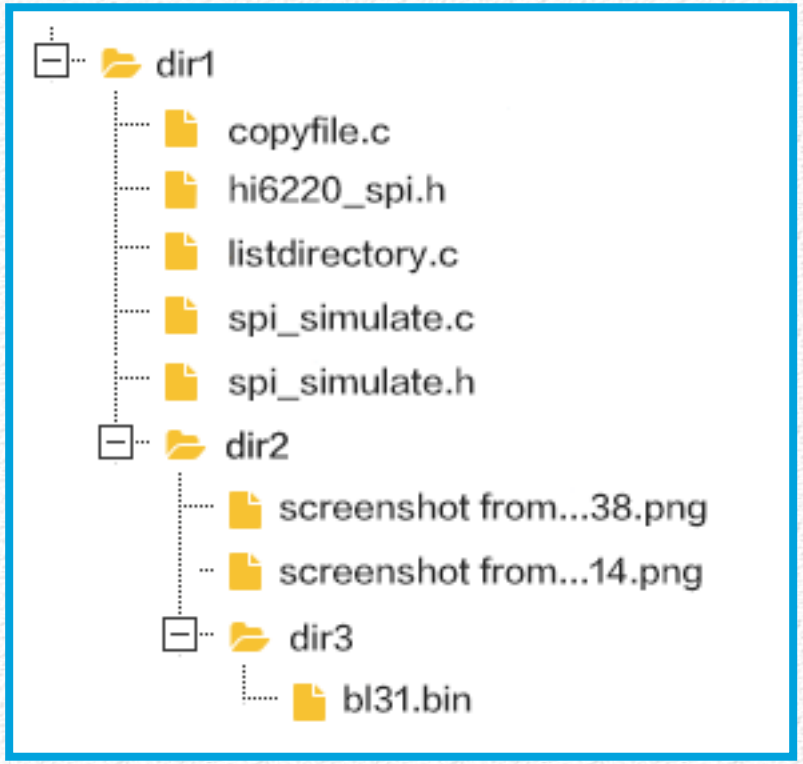
在实现基本功能实现的基础上，对程序功能进行扩展，以支持对目录的递归遍历并将**所有目录**及**子目录中的文件**拷贝到一个目录中

- ✓ 如果没有命令行参数则通过getcwd获取当前工作目录
- ✓ 如果包含一个命令行参数则通过该参数传递需要遍历的目录
- ✓ 如果有超过一个命令行参数则出错
- ✓ 拷贝文件及子目录下的文件

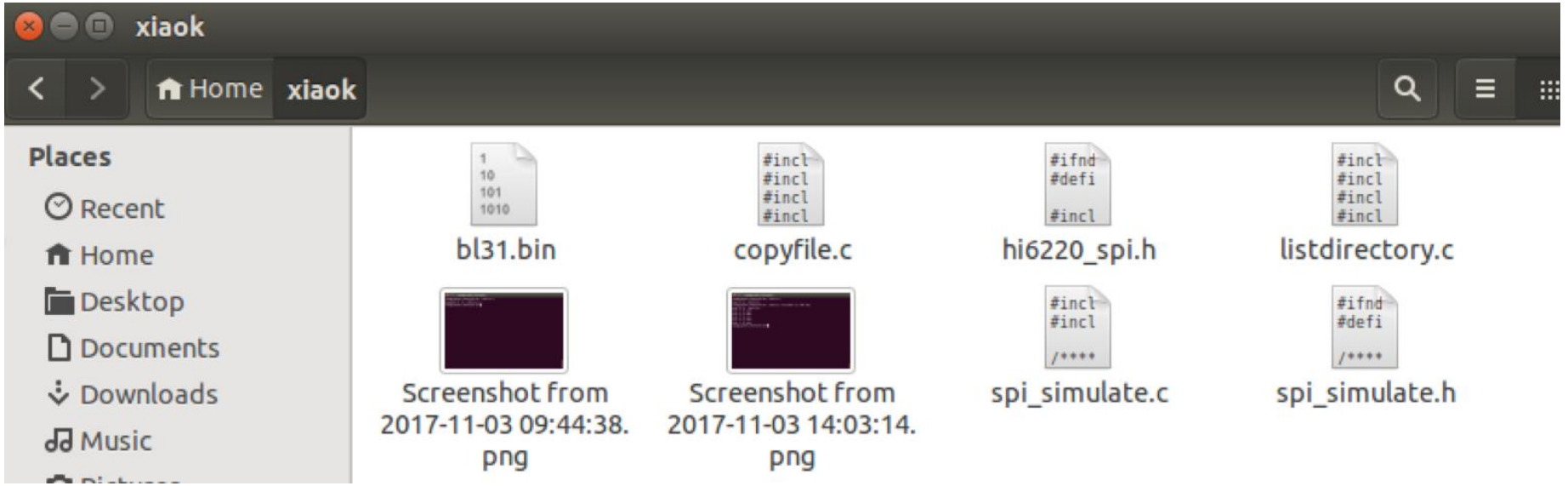
实验四 | 源文件目录结构



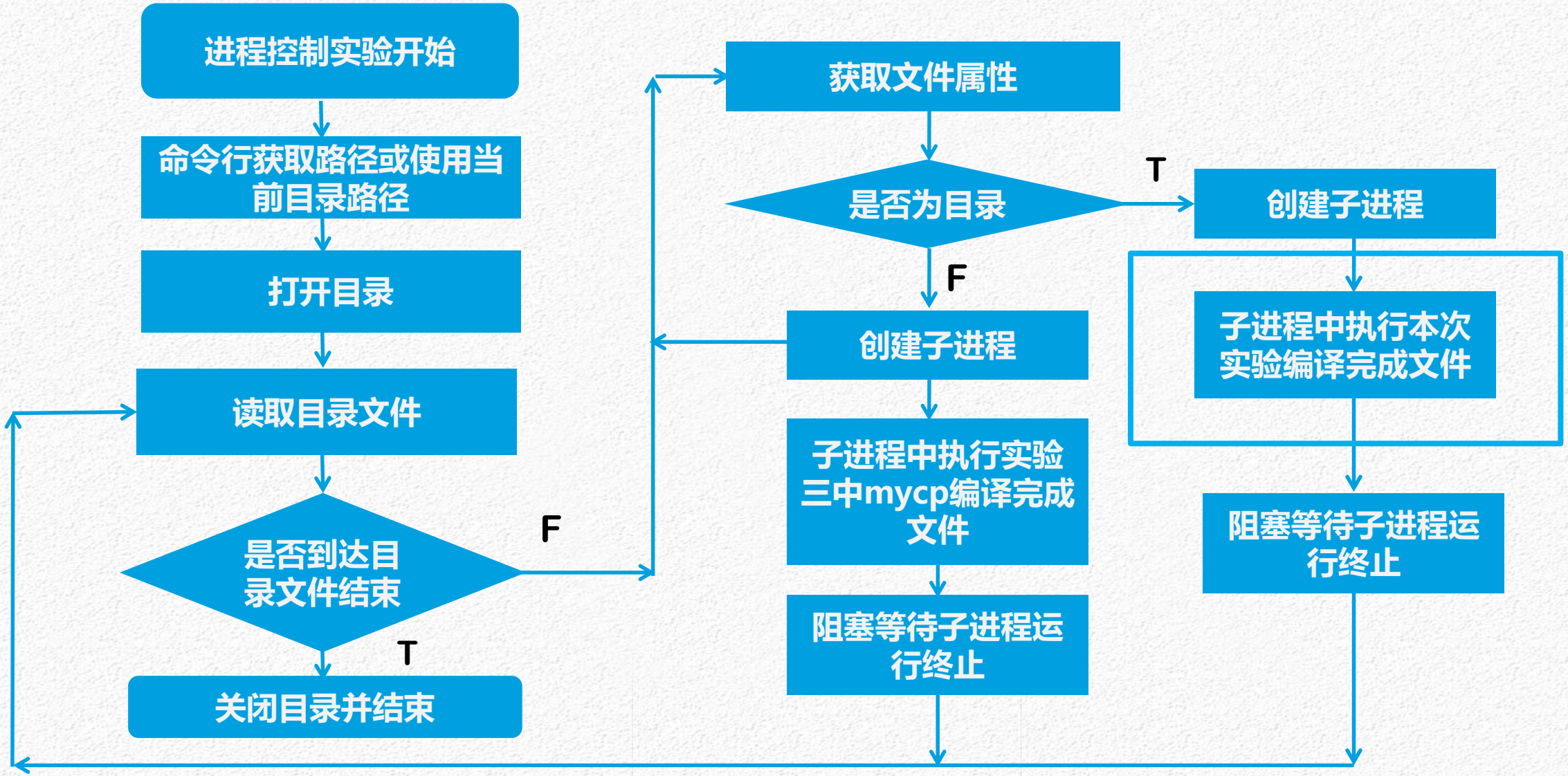
源目录结构如下所示：



实验四 | 实现对非目录文件拷贝

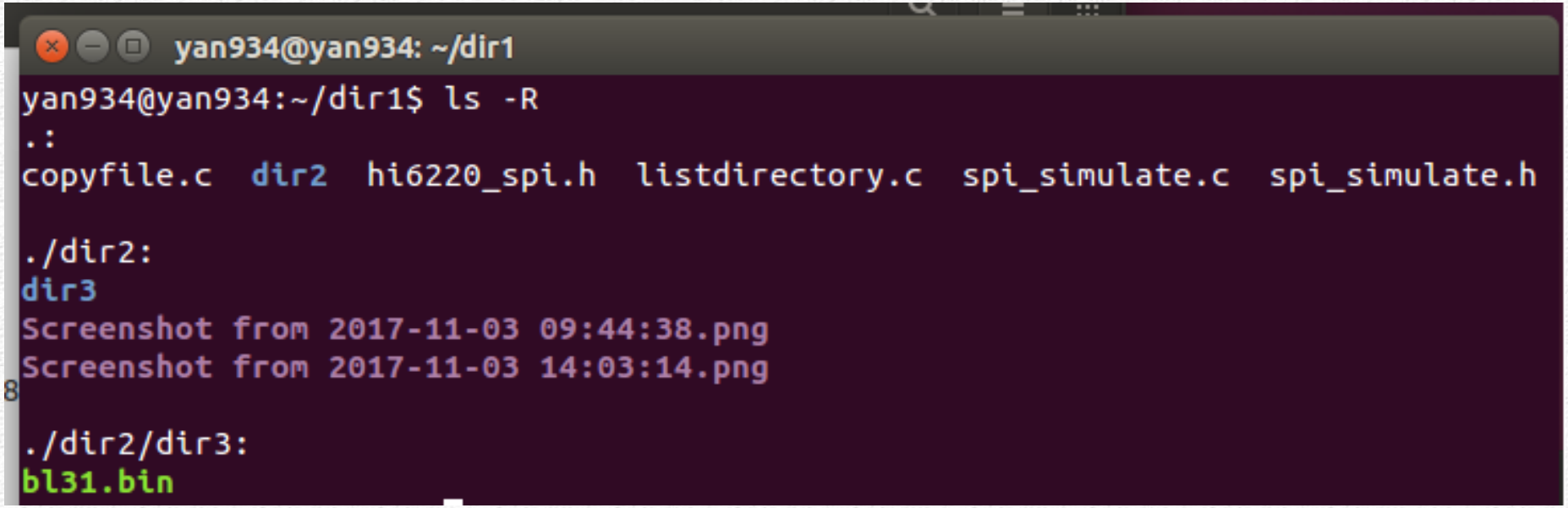


实验四｜扩展内容程序流程



递归：

- ✓若一个过程直接地或间接地调用自己，则称这个过程是递归的过程
- ✓将问题分解为同一个过程在不同层次上的多次执行



电子科技大学

University of Electronic Science and Technology of China

Linux操作系统编程

感谢观看

主讲老师：杨珊