

**实 验（实训）报 告**

**项 目 名 称**  进程管理

**所属课程名称**  操作系统

**项 目 类 型**  验证/设计型

**实验(实训)日期**  2024.11.20

**班 级**  22软件2

**学 号**  220110900219

**姓 名**  李伊能

**指导教师**  陈伟锋

浙江财经大学教务处制

|  |
| --- |
| **一、实验（实训）概述：（简单复述实验要求，1页以内，红字部分应删除）** |
| **【目的及要求】**        **【实施环境】**  **Ubuntu** |
| **二、实验（实训）内容：（记录实验内容，要求逻辑清晰、页面整洁、必要的图文并茂，核心的数据及程序代码，不能完全复制所有代码，完整代码若有必要，可以增加附录，尽量控制在6页以内，红字部分应删除）** |
| **【实验（实训）过程】（步骤、记录、数据、程序等）**  一、  1.安装gcc编译器  Sudo apt install gcc  2.  3.  4.  5.    二、    #include <stdio.h>  #include <stdlib.h>  #include <unistd.h>  #include <signal.h>  #include <sys/types.h>  #include <sys/wait.h>  // 信号处理器  void sig\_handler(int signum) {  switch(signum) {  case SIGINT:  printf("Received SIGINT\n");  break;  case SIGUSR1:  printf("Received SIGUSR1\n");  break;  default:  printf("Unknown signal received\n");  break;  }  }  int main() {  // 注册信号处理器  signal(SIGINT, sig\_handler);  signal(SIGUSR1, sig\_handler);  // 创建子进程  pid\_t pid = fork();  if (pid < 0) {  perror("fork");  exit(EXIT\_FAILURE);  } else if (pid == 0) {  // 子进程  printf("Child process PID: %d\n", getpid());  sleep(5); // 子进程休眠5秒  printf("Child process exiting...\n");  exit(EXIT\_SUCCESS);  } else {  // 父进程  printf("Parent process PID: %d\n", getpid());  printf("Child process PID: %d\n", pid);  // 发送SIGUSR1信号给子进程  kill(pid, SIGUSR1);  // 等待子进程结束  int status;  waitpid(pid, &status, 0);  if (WIFEXITED(status)) {  printf("Child process exited with status: %d\n", WEXITSTATUS(status));  } else {  printf("Child process did not exit normally\n");  }  printf("Parent process exiting...\n");  exit(EXIT\_SUCCESS);  }  }  **【结论与讨论】（结果、分析）** |
| **三、指导教师评语及成绩：** |
| **评语：**  **成绩： 指导教师签名：**  **批阅日期：** |