Lab05 - Assignment

> 姓名：周星达 学号：21373339

1. 进程

1.1 基础

请写这样一个程序（不是函数）：传入三个参数，传入该程序的第一个参数用以判断该程序进行有理数算术加运算还是减运算（0 表示将要进行加运算，1 表示将要进行减运算），第二个第三个参数分别是加（减）运算的第一第二个元素。（提示：`main(int argc, char* argv[])`，可获取命令行参数）。

```
int main(int argc, char* argv[])
{
    switch(*argv[1])
    {
        case '0':
            printf("%d\n",(*argv[2]-48)+(*argv[3]-48));
            break;

        case '1':
            printf("%d\n",(*argv[2]-48)-(*argv[3]-48));
            break;
    }
    return 0;
}
```

1.2 僵尸进程

僵尸进程有什么危害？编写一个会产生僵尸进程的程序并运行，在终端查看当前进程。然后利用终端杀死该进程。

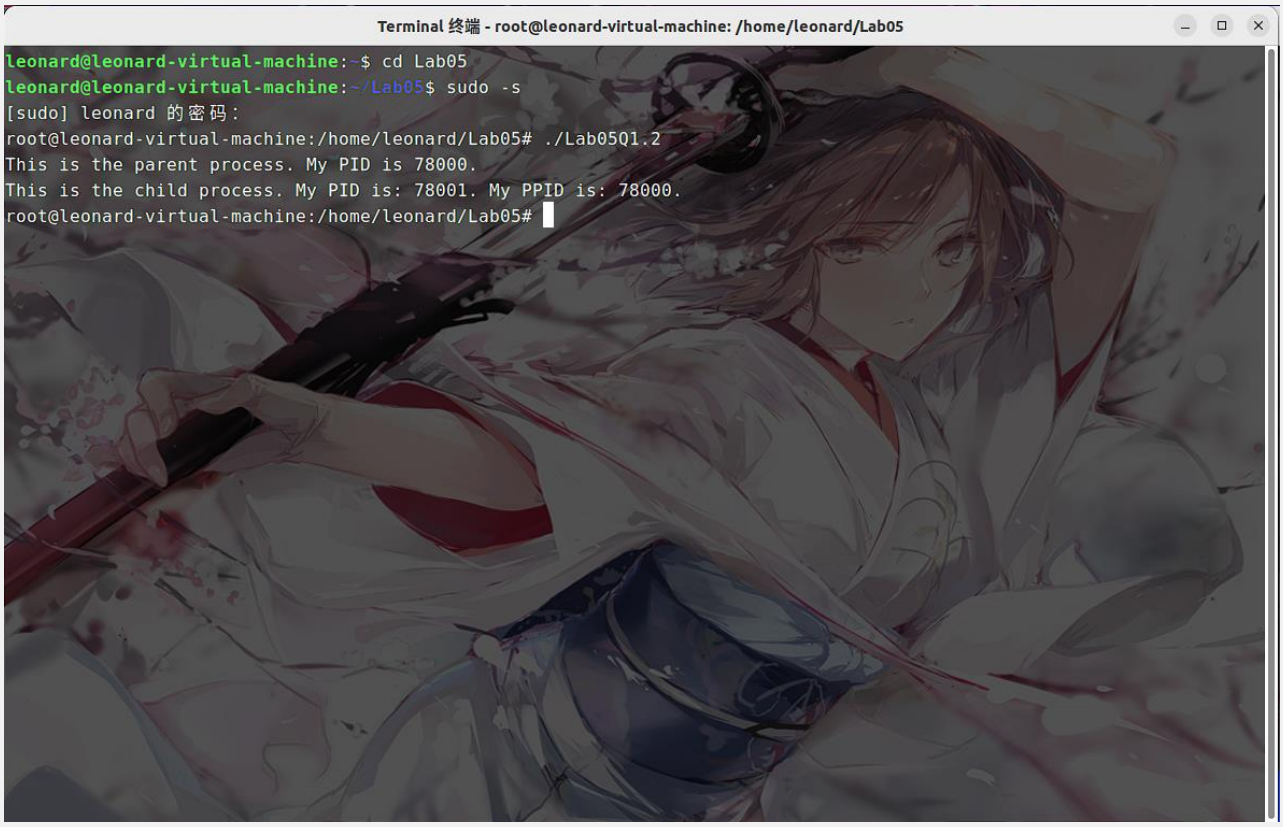
> 僵尸进程的危害：如果大量的僵尸进程存在，就可能因为没有可用的进程号而导致系统不能产生新的进程。

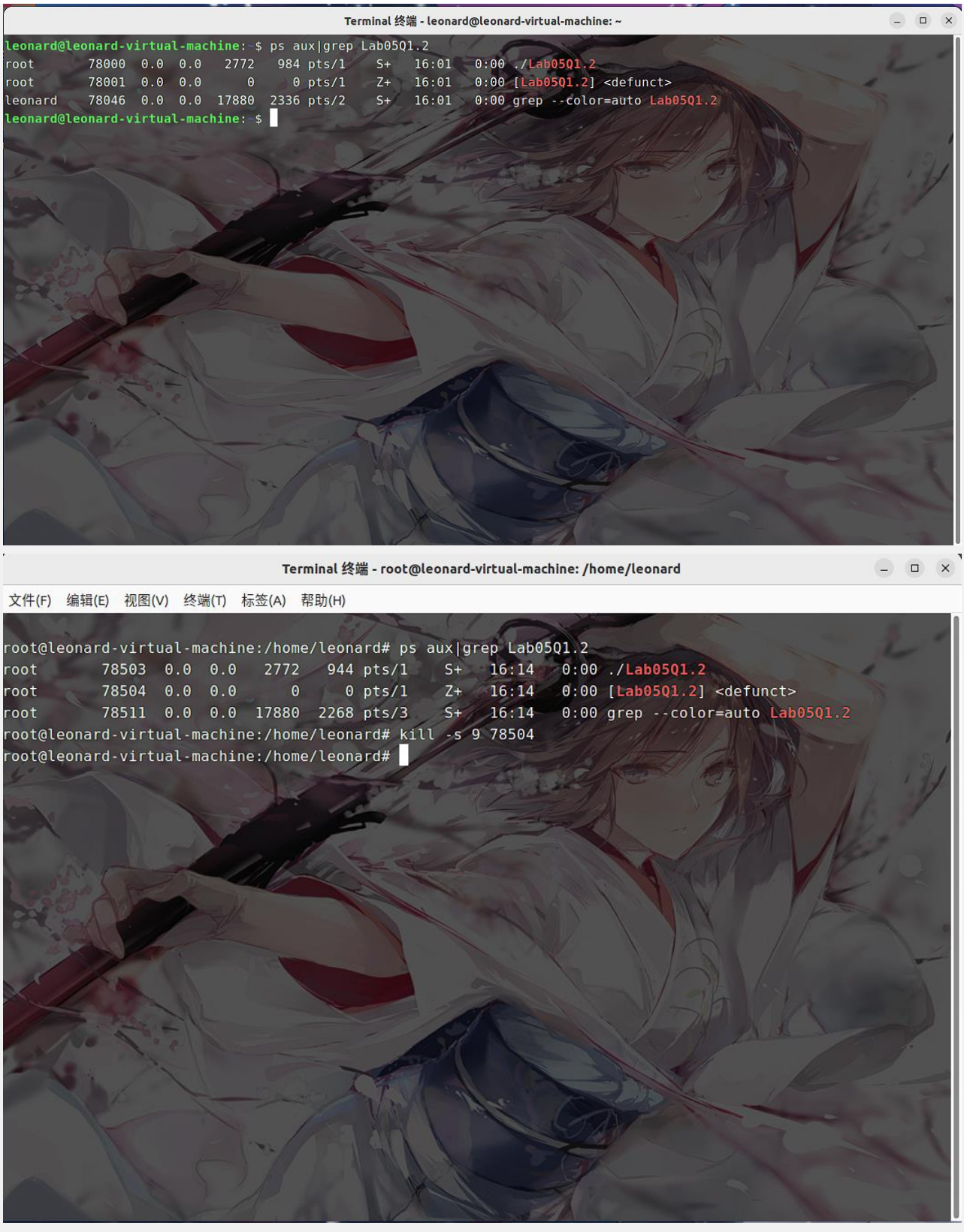
```
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
  int i = 100;
  pid_t pid=fork();
  if(pid < 0)
  {
    perror("fork failed.");
    exit(1);
  }
```

```
  if(pid > 0)
  {
    printf("This is the parent process. My PID is %d.\n", getpid());
    for(; i > 0; i--)
    {
      sleep(1);
    }
  }
  else if(pid == 0)
  {
    printf("This is the child process. My PID is: %d. My PPID is: %d.\n",
getpid(), getppid());
  }

  return 0;
}
```

运行、杀死僵尸进程的过程截图





2. 进程控制

2.1 fork 与 wait

编写一段C程序，由父进程创建两个子进程，父进程打印字符 B，两个子进程分别打印 A和 C，并且要求最终的输出为 ABC。

```c
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>

int main(int argc, char **argv)
{
  pid_t pid,pid0,pid1;
  int status;
  pid = fork();
  if (pid == 0)
  {
    printf("A");
    exit(12);
  }
  else
  {
    pid0=wait(&status);
    pid1=fork();
    if(pid2==0)
    {
      printf("C\n");
      exit(0);
    }
    else
    {
      printf("B");
    }
  }
}
```

2.2 exec族函数

请编写一段C程序，该程序调用调用exec族函数，对当前目录使用 `ls -l`。

```c
#include <limits.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <errno.h>
#define PATHNAME_MAX 1000

int main()
{
    char buf[PATHNAME_MAX];

    /* 使用getcwd获取启动目录 */
    if (getcwd(buf, sizeof(buf))==NULL)
    {
        fprintf(stderr, "getcwd error: %s", strerror(errno));
        exit(1);
```

```
    }
    //printf("current work path: %s\n", buf);
    if(execl("/bin/ls", "ls", "-l",buf, (char *) 0)==-1)
    {
        fprintf(stderr, "execl error: %s", strerror(errno));
        exit(1);
    }
    return 0;
}
```

2.3 综合应用

请你编写一段C程序，该程序按顺序依次调用 题目1.1中的可执行文件分别执行 1+1与 1-1，随后调用 /bin/rm 删除 题目1.1的可执行文件。

```
#include <limits.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <errno.h>
#define PATHNAME_MAX 1000

int main()
{
    pid_t pid = fork();
    pid_t pid0 = fork();
    if (pid > 0 && pid0 > 0)
    {
        sleep(2);
        if (execl("/bin/rm", "rm", "-f", "L5Q1-1", (char *) 0) == -1)
        {
            fprintf(stderr, "execl error: %s", strerror(errno));
            exit(1);
        }
    }
    else if (pid == 0 && pid0 > 0)
    {
        if (execl("./L5Q1-1", "L5Q1-1", "0", "1", "1",(char *) 0) == -1)
        {
            fprintf(stderr, "execl error: %s", strerror(errno));
            exit(1);
        }
    }
    else if (pid0 == 0 && pid > 0)
    {
        sleep(1);
        if (execl("./L5Q1-1", "L5Q1-1", "1", "1", "1",(char *) 0) == -1)
        {
            fprintf(stderr, "execl error: %s", strerror(errno));
            exit(1);
```

```
            }
        }
        return 0;
    }
```

3. 进程通信

3.1 重定向

请编写一个C程序，使用文件描述符与重定向把你的学号输出到 `student.txt`文件中

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <unistd.h>
#include <fcntl.h>

int main()
{
    close(1);
    char buf[] = "21373339";
    int fd = open("./student.txt", O_WRONLY | O_CREAT, 0644);
    printf("%s\n", buf);
}
```

3.2 综合应用

请使用管道编写素数筛选的并发版本C程序 `primes.c`，程序原理见下面的介绍或者查看这个网站。该程序读入命令行的第一个参数n，随后输出1-n之间的素数。

运行命令

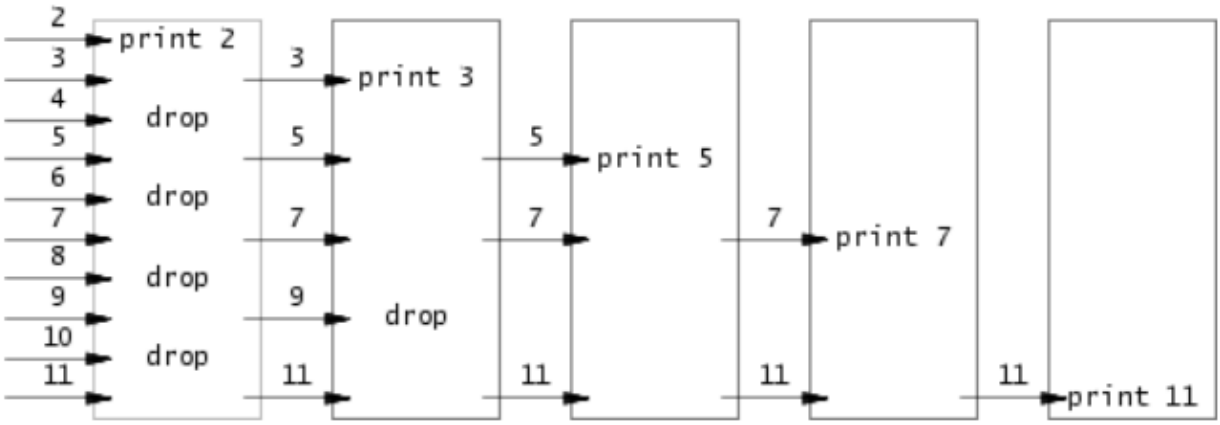
```
gcc primes.c -o primes
./primes 7
```

输出为

```
2
3
5
7
```

> 考虑所有小于1000的素数的生成。Eratosthenes的筛选法可以通过执行以下伪代码来模拟：

```
p = get a number from left neighbor
print p
loop:
    n = get a number from left neighbor
    if (p does not divide n)
        send n to right neighbor
p = 从左邻居中获取一个数
print p
loop:
    n = 从左邻居中获取一个数
    if (n不能被p整除)
        将n发送给右邻居
```



生成进程可以将数字2、3、4、...、1000输入管道的左端：行中的第一个进程消除2的倍数，第二个进程消除3的倍数，第三个进程消除5的倍数，依此类推。

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

void source(int max)
{
    int i;
    for (i = 2; i <= max; i++)
    {
        write(1, &i, sizeof(i));
    }
}

void cull(int p)
{
    int n;
    while (read(0, &n, sizeof(n)))
    {
        if (n % p != 0)
        {
            write(1, &n, sizeof(n));
```

```
        }
    }
}

void redirect(int k, int pd[])
{
    close(k);
    dup(pd[k]);
    close(pd[0]);
    close(pd[1]);
}

void sink()
{
    int pd[2];
    int p;
    if (read(0, &p, sizeof(p)))
    {
        printf("%d\n", p);
        pipe(pd);
        if (fork())
        {
            redirect(0, pd);
            sink();
        }
        else
        {
            redirect(1, pd);
            cull(p);
        }
    }
}

int main(int argc, char* argv[])
{
    int pd[2];
    pipe(pd);
    if (fork() > 0)
    { // 父进程
        // 重定向标准输入文件到pd[0]
        redirect(0, pd);
        sink();
    }
    else
    {
        //子进程重定向标准输出文件到pd[1]
        redirect(1, pd);
        source(atoi(argv[1]));
    }
    exit(0);
}
```