练习03

在课堂上,我们讨论了有多少 shell 命令是作为单独的模块化程序实现的。 shell 允许我们一次运行它们,或者,如果我们愿意,我们可以一次启动多个命令并让它们一起工作。

这就是下面发生的事情:

$ ps -ef | grep 重击

此命令使用 -ef 选项运行 ps。像这样运行时,ps 将打印在该服务器上运行的所有进程的报告
系统。竖条要求 shell 同时启动 grep 命令并将 ps 的输出发送为
输入到 grep。 Grep 是一个模式匹配程序。在这里,我们要求 grep 过滤掉除那些之外的所有进程
在命令行上以单词“bash”开头。所以这个命令将(大部分)打印所有 bash 副本的报告
当前在系统上运行。

对于本练习,我们将编写自己的代码来执行 shell 在这里执行的操作。我们已经知道如何开始
两个子进程,让其中一个运行 ps,一个运行 grep,然后等待它们都终止。新的
我们需要弄清楚的是如何让他们沟通。我们将使用(匿名)管道来做到这一点。

我给你一个部分实现,pipe.c,帮助你开始。它完成创建两个孩子的工作
进程并等待它们完成。你只需要添加通信代码和实际启动的代码
启动 ps 和 grep 程序。您需要执行以下操作。我在代码中留下了一些像“// ...”这样的注释,其中
你需要添加一些东西。

- 在开始创建子项之前,您需要创建一个管道。这样,孩子们将继承该文件
  通过管道读写的描述符。
- 在第一个子进程中启动 ps 之前,您需要重定向其标准输出,以便写入写入端
  的管道,而不是终端。这是一个两步过程,有点像我们重定向到文件所做的
  在之前的练习中。
  - 首先,使用 dup2() 关闭孩子的旧标准输出文件描述符并将其替换为
    管道的写入端。从此以后,孩子所做的任何打印都会自动写入
    管道。如果您需要打印出任何调试信息,您可能需要使用标准错误,因为它
    仍将前往航站楼。带在身边很方便。
  - 现在,子进程有两个文件描述符都写入管道,一个是从父进程继承的,另一个是
    它刚刚制作的副本作为标准输出。关闭原始的(从父级继承的那个),因为
    孩子现在不需要了。
- 在第一个孩子中,使用execl 启动带有“-ef”选项的ps。
- 在第二个孩子中放置类似的代码,用管道的读取端替换其标准输入。那么,有

这个孩子运行 grep,将单词“bash”作为命令行选项。

- 通常,您需要确保关闭所有不需要的文件描述符以读取或写入管道。

  Grep 将继续从其标准输入读取数据,直到到达 EOF。为此,需要读取管道中的所有缓冲数据,并且需要关闭所有用于写入管道的文件描述符。所以,例如,

  父级应该在创建管道后关闭其读写文件描述符的副本

  孩子们。父母不再需要它们中的任何一个（并且保持它们打开会导致 grep 孩子

  悬挂）。同样,孩子们不需要将管子的两端都打开,只需要打开他们需要使用的那一端即可。

一旦你的程序运行起来,它应该打印出一个运行命令列表,其中包含单词“bash”。这将
主要是系统上运行的所有 shell,可能还有您自己的程序启动的“grep bash”副本。

完成后,上交已完成的 pipe.c 程序的源代码。将文件提交到 Gradescope
作业名为 EX03。