CS213，2002年秋季
实验室作业L5：编写自己的Unix Shell分配：10月24日，到期：10月31日，11：59PM

哈里·博维克（bovik@cs.cmu.edu）是这项任务的负责人。

导言

这项任务的目的是更熟悉过程控制和信令的概念。您将通过编写一个支持作业控制的简单Unixshell程序来实现这一点。

后勤

你可以在一组最多两个人的工作中解决这个任务的问题。唯一的“上交”将是电子的。对作业的任何澄清和修改将张贴在课程网页上。

发出指示

> 站点-**SPEC IF IC**：在这里插入一个段落，解释教师将如何向学生分发**shlab-handout.tarfile**。这是我们在**CMU**使用的方向。

从复制文件shlab-handout.tarto保护目录（实验室目录）开始，您计划在其中完成您的工作。然后做以下工作：

- 键入命令tar XVFShlab-handout.tarto展开tarfile。
- 键入命令make以编译和链接一些测试例程。
- 在oftsh.c顶部的标题注释中键入您的团队成员名称和AndrewID。

看看tsh.c（microshell）文件，你会发现它包含了一个简单Unixshell的功能骨架..为了帮助您开始，我们已经实现了不那么有趣的函数。你的任务

是完成下面列出的剩余空函数。作为对您的理智检查，我们在我们的参考解决方案中列出了每个函数的大致代码行数（其中包括大量注释）。

- 解析和解释命令行的主要例程。[70行]
- 内置cmd：识别和解释内置命令：退出、FG、BG和jobs.line] [25
- 做bgfg：实现bgandfg内置命令。[50行]
- 等待：等待前台工作完成。[20行]
- 信号处理程序：捕获SIG CHILD信号。80行]
- 信号处理程序：捕获信号（ctrl-c）信号。[15行]
- 信号处理程序：捕获SIGTST P（ctrl-z）信号..[15行]

每次修改tsh.cfile时，请键入make以重新编译它。若要运行shell，请将tsh输入到命令行：

```
unix> ./tsh
[在此向您的shell输入命令]
```

Unix Shells概述

shell是代表用户运行程序的交互式命令行解释器。一个shell反复打印一个提示，在stdin上等待一个命令行，然后按照命令行的内容进行一些操作..

命令行是由空格分隔的ASCII文本单词的序列。命令行中的第一个字要么是内置命令的名称，要么是可执行文件的路径名。剩下的单词是命令行参数。如果第一个单词是内置命令，shell将立即执行当前进程中的命令。否则，该单词被假定为可执行程序的路径名。在这种情况下，shell分叉一个子进程，然后在子进程的上下文中加载和运行程序。由于解释单个命令行而创建的子进程统称为ajob。一般来说，一个作业可以由Unix管道连接的多个子进程组成。

如果命令行以ampers and“ & ”结尾，则作业在后台运行，这意味着shell在打印提示符前不等待作业终止，并等待下一个命令行。否则，作业将在地下运行，这意味着shell在等待下一个命令行之前等待作业终止。因此，在任何时候，最多只有一个工作可以在前台运行。然而，任意数量的作业可以在后台运行。

例如，键入命令行

```
tsh> 工作
```

导致shell执行内置作业命令。键入命令行

```
tsh> /bin/ls -l -d
```

在前台运行LS程序。按照惯例，shell确保在程序开始执行其主要例程时

```
int main(int argc, char *[])
```

argcandargv参数具有以下值：

- `argc == 3`,
- `argv[0] == ‘‘/bin/ls’’`,
- `argv[1]== ‘‘-l’’`,
- `argv[2]== ‘‘-d’’`.

或者，键入命令行

```
tsh> /bin/ls -l -d &
```

在后台运行ls程序。

Unixshell支持作业控制的概念，允许用户在后台和前台之间来回移动作业，并更改作业中进程的进程状态（运行、停止或终止）。键入ctrl-导致一个SIGINT信号被传递到前台作业中的每个进程。对SIGINT的默认操作是终止进程。类似地，键入ctrl-z会导致一个SIGTS TP信号被传递到前台作业中的每个进程。对SIGTSTP的默认操作是将进程放置在停止状态，在那里它保持不变，直到收到SIGCONT信号唤醒为止。 Unixshell还提供各种支持作业控制的内置命令。例如：

- 作业：列出正在运行和停止的背景作业。
- BG<作业>：将停止的背景作业更改为正在运行的背景作业。
- FG<作业>：将停止或运行的后台作业更改为前台运行。
- 杀死<工作>：终止一份工作。

TSH规范

您的tshshell应该具有以下功能：

- 提示符应该是字符串“ tsh ” ”.

- 用户键入的命令行应该由名称和零或多个参数组成，所有这些参数都由一个或多个空格分隔。如果name是一个内置命令，那么tsh应该立即处理它，并等待下一个命令行。否则，tsh应该假设名称是可执行文件的路径，它在初始子进程的上下文中加载和运行该文件（在此上下文中，术语是指这个初始子进程）。

- 不需要支持管道（\\）或I/O重定向（<和>）。
- 键入ctrl-c（ctrl-z）应该导致SIGINT（S IGTST P）信号被发送到当前的前台作业，以及该作业的任何后代（例如，它分叉的任何子进程）。如果没有前景作业，那么信号应该没有影响。
- 如果命令行以ampers and&结尾，则tsh应该在后台运行作业。否则，它应该在前台运行工作。
- 每个作业可以通过进程ID（P ID）或作业ID（JID）来识别，JID是TSH分配的正整数。在命令行中，JID应由前缀“ % ”表示。例如，“ %5 ”表示JID5，“ 5 ”表示PID5。（我们已经为您提供了操作工作清单所需的所有程序。

- tsh应支持以下内置命令：
  - 退出命令终止shell。
  - 职务命令列出所有背景职务。
  - 通过向其发送SIGCONT信号，bg<job>命令重新启动<job>，然后在后台运行..该<作业>参数可以是PID或JID。
  - 通过向其发送SIGCONT信号，fg<job>命令重新启动<job>，然后在前台运行..该<作业>参数可以是PID或JID。
- 它应该收割它所有的僵尸孩子。如果任何作业终止是因为它接收到一个信号，它没有捕捉到，那么tsh应该识别这个事件，并打印一条带有作业的PID和违规信号的描述的消息。

检查你的工作

我们提供了一些工具来帮助您检查您的工作。

参考解决方案。**Linux**可执行文件**tshref**是**shell**的参考解决方案。运行此程序以解决有关**shell**应该如何运行的任何问题。您的**shell**应该发出与参考解决方案相同的输出（当然，除了**PID**，它从运行到运行）。

炮弹司机。**sdriver.plprogram**作为子进程执行**shell**，按照跟踪文件的指令发送命令和信号，并捕获和显示**shell**的输出。

使用-h参数找出sriver.pl的用法：

```
unix> ./sdriver.pl -h
使用方法：sdriver.pl[-hv]-t<trace>-s<shellprog>-a<args>选项：

  -h            打印此消息更详细的跟
  -v            踪文件
  -t <trace>
  -s <shell>    测试Shell参数的Shell程
  -a <args>     序
  -g            为自动评分器生成输出
```

我们还提供了16个跟踪文件（Trace{01-16}.txt），您将与shell驱动程序一起使用这些文件来测试shell的正确性。编号较低的跟踪文件做非常简单的测试，编号较高的测试做更复杂的测试。

您可以使用跟踪文件trace01.txt（例如）通过键入：

```
unix> ./sdriver.pl -t trace01.txt -s ./tsh -a "-p"
```

（-a“ -p ”参数告诉你的shell不要发出提示），或者

```
unix> 做test01
```

同样，要将结果与引用shell进行比较，可以通过键入来运行引用shell上的跟踪驱动程序：

```
unix>  ./sdrive.pl-t trace01.txt-s./tshref-a“ -p ”
```

或

```
unix> make rtest01
```

供您参考， tshref.out给出了所有种族的参考解决方案的输出。这可能比在所有跟踪文件上手动运行shell驱动程序更方便。

跟踪文件的好处在于，如果您交互运行shell，它们将生成相同的输出（除了标识跟踪的初始注释之外）。例如：

```
贝斯>做测试15
./sdriver.pl -t trace15.txt -s ./tsh -a "-p"
#
#  trace15.txt-把它放在一起
#
tsh> ./bogus
./bogus：命令未找到.tsh>./myspin
10
由信号2tsh>./myspin3&终止的作业（9721
)
[1] (9723) ./myspin 3 &
tsh> ./myspin 4 &
```

```
[2] (9725) ./myspin 4 &
tsh> jobs
[1] (9723) 运行[2]     ./myspin 3 &
 (9725) 运行tsh>fg     ./myspin 4 &
%1
工作[1] (9723) 被信号20tsh>工作停止

[1] (9723) 停止运行[     ./myspin 3 &
2] (9725) 运行tsh>bg     ./myspin 4 &
%3
%3: 没有这样的工作
tsh> bg %1
[1] (9723) ./myspin 3 &
tsh> jobs
[1] (9723) 运行[2]     ./myspin 3 &
 (9725) 运行tsh>fg     ./myspin 4 &
%1
tsh>退出贝
斯>
```

暗示

- 阅读教材第8章（特殊控制流程）的每一个单词。
- 使用跟踪文件来指导shell的开发。从trace01.txt开始，确保您的shell产生与引用shell相同的输出。然后继续跟踪文件trace02.txt，以此类推..
- 等待，杀死，叉子，execve，setpgid和sigprocmaskfunctions将非常有用。WUN TRACED和WNO HANG选项waitpid也很有用。
- 当您实现信号处理程序时，请确保将SIGINT和SIGTS TP信号发送到整个前台进程组，在参数中使用“ -PID ”而不是“ PID ”到Killfunction。 sriver.plprogram测试此错误。
- 其中一个棘手的部分是决定waitfg和sigchldhandler函数之间的工作分配。我们建议采取以下办法：
  - 在waitfg中，使用一个围绕睡眠函数的繁忙循环..
  - 在sigchld处理程序中，使用一个调用来等待。

  虽然其他解决方案是可能的，例如调用waitpidinwaitfgand sigchld处理程序，但这些可能会非常令人困惑。在处理程序中完成所有的收获是更简单的。
- 在eval中，父方必须使用sigprocmask来阻止SIGCH LD信号，然后打开这些信号，再次使用sigprocmask，然后通过调用addjob将子添加到作业列表中。由于孩子继承了他们父母的阻塞向量，所以孩子必须确保在执行新程序之前解除SIGCH LD信号。

父级需要以这种方式阻止SIGCH LD信号，以避免在父级调用Addjob之前由sigchld处理程序（从而从作业列表中删除）收获子级的比赛条件。

- 程序，如多，少，vi，和emacsdo奇怪的东西与终端设置。不要从你的外壳中运行这些程序。坚持使用简单的基于文本的程序，如/bin/ls、 /bin/ps和/bin/echo。

- 当您从标准Unixshell中运行shell时，您的shell将在前台进程组中运行。如果您的shell随后创建了一个子进程，默认情况下该子进程也将是前景进程组的成员。由于键入ctrl-c将一个SIGINT发送到前台组中的每个进程，键入ctrl-c将向您的shell发送一个SIGINT，以及您的shell创建的每个进程，这显然是不正确的。

  这里是解决办法：在叉子之后，但在execve之前，子进程应该调用setpgid（0，0），这将子进程放在一个新的进程组中，其组ID与子进程的PID相同。这将确保前台进程组中只有一个进程，即您的shell。当您键入ctrl-c时，shell应该捕获产生的SIGINT，然后将其转发到适当的前台作业（或更准确地说，包含前台作业的进程组）。

评价

你的分数将根据以下分布从最高90分计算出来：

80 正确性：16个痕迹档案各5分..

10 风格要点。我们期待您有好的意见（5pts），并检查每一个返回值。
系统调用（5PTS
）。

您的解决方案外壳将在Linux机器上测试是否正确，使用相同的shell驱动程序和包含在实验室目录中的跟踪文件。您的外壳应该在这些跟踪上产生与参考外壳相同的输出，只有两个例外：

- PID可以（也将）是不同的。
- 在trace11.txt、 trace12.txt和trace13.txt中的/bin/ps命令的输出将不同于运行运行。但是，在/bin/ps命令的输出中，任何mysplit进程的运行状态都应该是相同的。

手提指示

网站：在这里插入一个段落，解释学生应该如何上交他们的**tsh.cfile**。这是我们在**CMU**使用的方向。

- 确保您已将您的姓名和AndrewID包含在tsh.c的标题注释中。
- 创建表单的团队名称：
  - “ ID ”，其中ID是您的安德鲁ID，如果您是单独工作，或
  - 其中ID1是第一个团队成员的AndrewID，ID2是第二个团队成员的AndrewID。
- 我们需要您以这种方式创建您的团队名称，以便我们可以自动升级您的任务。若要提交tsh.cfile，请键入：

  ```
  make handin TEAM=teamname
  ```

  其中Teamname是上面描述的团队名称。
- 在上交后，如果发现错误并想提交修改后的副本，请键入

  ```
  make handin TEAM=teamname VERSION=2
  ```

  继续增加版本号与每个提交。3.你应该看看情况，核实一下你的手
-

  ```
  /afs/cs.cmu.edu/academic/class/15213-f01/L5/handin
  ```

  您在此目录中有列表和插入权限，但没有读或写权限。

祝你好运！