Universidade Federal de Minas Gerais
Departamento de Ciência da Computação
Sistemas Operacionais – 2019/2

Trabalho Prático #1 – Shell Básico

1) Introdução

Nesse trabalho iremos explorar alguns conceitos da primeira parte da disciplina. Em particular, iremos rever os conceitos de *pipes* e estruturas de processos de *kernel*.

Neste trabalho você se familiarizará com a interface de chamadas de sistema do Linux implementando algumas funcionalidades de um shell bem simples. Para que você foque apenas na parte de chamadas de sistema, abra o arquivo fonte do shell `sh.c` e estude-o. O esqueleto do shell contém duas partes: *um processador de linhas de comando e o código para execução dos comandos*. Você não precisa modificar o processador de linhas de comando, **mas deve completar o código para execução dos comandos**. O processador de linhas só reconhece comandos simples como:

```
ls > y
cat < y | sort | uniq | wc > y1
cat y1
rm y1
ls | sort | uniq | wc
rm y
```

Se você não entende o que esses comandos fazem, estude o manual de um shell do Linux (por exemplo, do *bash*) bem como o manual de cada um dos comandos acima (`ls, cat, rm, sort, uniq, wc`) para se familiarizar. Copie e cole esses comandos num arquivo, por exemplo, `teste.sh`.

Você pode compilar o esqueleto do shell rodando:

```
$ gcc sh.c -o myshell.out
```

Nota: Nesta especificação colocamos um sinal de dólar (`$`) antes das linhas que devem ser executadas no shell do sistema (por exemplo, o bash). As linhas de comando sem dólar devem ser executadas no shell simplificado que você está implementando.

Esse comando irá produzir um arquivo a.out que você pode rodar:

```
$ ./myshell.out
```

Para sair do shell simplificado aperte `ctrl+d` (fim de arquivo). Teste o shell executando os comandos no arquivo `teste.sh`:

```
$ ./myshell.out < teste.sh
```

Essa execução irá falhar pois você ainda não implementou várias funcionalidades do shell. É isso que você fará nesse trabalho.

2) Executando Comandos Simples

Implemente comandos simples, como:

```
ls
```

O processador de linhas já constrói uma estrutura `execcmd` para você, a única coisa que você precisa fazer é escrever o código do `case ' '` (espaço) na função `runcmd`. Depois de escrever o código, teste execução de programas simples como:

```
ls
cat sh.c
```

Nota: Você não precisa implementar o código do programa `ls` e dos outros. O que você deve fazer é simplesmente implementar as funções no esqueleto do shell simplificado para permitir que ele execute comandos já existentes no sistema, como acima.

Dica: dê uma olhada no manual da função exec (`$ man 3 exec`). **Importante: *não* use a função `system` para implementar as funções do seu shell.**

3) Redirecionamento de Entrada e Saída

Implemente comandos com redirecionamento de entrada e saída para que você possa rodar:

```
echo "Sistemas Operacionais é legal" > x.txt
cat < x.txt
```

O processador de linhas já reconhece `">"` e `"<"` e constrói uma estrutura `redircmd` para você. Seu trabalho é apenas preencher o código na função `runcmd` para esses casos. Teste sua implementação com os comandos acima e outros comandos similares.

Dica: Dê uma olhada no manual das funções `open` e `close` (`$ man 2 open`). Se você não conhece o esquema de entrada e saída padrão de programas, dê uma olhada no artigo da Wikipedia[1].

4) Sequenciamento de Comandos

Implemente *pipes* para que você consiga rodar comandos tipo:

1 http://en.wikipedia.org/wiki/Standard_streams

```
ls | sort | uniq | wc
```

O processador de linhas já reconhece o `'|'` e constrói uma estrutura `pipecmd` pra você. A única coisa que você precisará fazer é completar o código para o `case '|'` na função `runcmd`. Teste sua implementação para o comando acima. Se precisar, leia a documentação das funções `pipe`[2], `fork`[3] e `close`[4].

5) Entrega

Esse trabalho poderá ser feito em dupla.

A data de entrega é 04/09/2019, quarta-feira, até as 23:55 via moodle.

Seu grupo deverá submeter no moodle *apenas* o arquivo `sh.c`, em um (único) arquivo chamado `sh.c` (o nome deve ser exatamente `sh.c` para que seu *shell* possa ser testado automaticamente). Além disso, deve constar um relatório descrevendo a solução, conforme modelo no arquivo report.txt disponibilizado.

Esse trabalho vale **10** pontos.

6) Esclarecimentos

1. **Não use a função `system` na sua implementação!** Use `fork` e `exec`[5].
2. Esse trabalho utiliza obrigatoriamente a linguagem C.
3. Um grupo de discussão para esse trabalho será aberto no moodle para resolver quaisquer dúvidas.
4. Seu shell será testado com o script `grade.sh` disponibilizado na especificação. A saída será conferida automaticamente. Por causa disso, *seu shell deve imprimir somente a saída dos programas* em casos onde não ocorre erro. Use o script `grade.sh` disponibilizado para verificar a corretude de sua implementação.
5. Comece a fazer esse trabalho o quanto antes!

2 https://linux.die.net/man/2/pipe
3 https://linux.die.net/man/2/fork
4 https://linux.die.net/man/2/close
5 https://linux.die.net/man/3/exec