Universidade Federal da Bahia

Semestre Letivo Suplementar 2020 **MAT566 - Sistemas Operacionais**

Componentes: Elivelton Rangel, Jane Barbosa, Mariese Conceição e Nilvan Santana

Documentação do programa Myshell

Essa documentação visa esclarecer aspectos técnicos da implementação do programa filecopy, e tem como objetivo facilitar o entendimento do código para que outras pessoas possam realizar manipulações no programa sem grande dificuldades.

O programa Myshell é um programa escrito na linguagem de programação C para simular o shell do sistema operacional, oferecendo comandos para gerenciar programas de usuário, além de trabalhar com chamadas de sistemas para manipulação de processos pai e filhos. O programa Myshell é capaz de executar, gerenciar e monitorar programas de nível do usuário, é importante ressaltar que o programa e invocado sem quaisquer argumentos.

Quando executado, o myshell deverá imprimir um prompt similar a myshell quando estiver pronto para aceitar comando de entrada. O programa deve ler ainda uma entrada de linha de comandos, aceitando vários parâmetros possíveis.

Utilização de bibliotecas

Para correta manipulação do programa foram utilizadas as seguintes bibliotecas: stdio.h (possui diversas funções de entrada e saída e por isso é utilizada por quase todos os programas escritos em C), stdlib.h (possui inúmeras funções e constantes básicas), unistd.h (fornece acesso a API do sistema), string.h (possui funções para manipular strings) e sys/wait.h (que contém um conjunto de macros que permite trabalhar com processos pai). Além das bibliotecas padrões, uma biblioteca denominada “error_handles.h/.c” que recebe todas as importações das bibliotecas mencionadas acima, e uma função responsável por reportar possíveis erros de sistema.

A biblioteca stdio.h é um cabeçalho da biblioteca padrão do C que possui diversas funções de entrada e saída e por isso é utilizada por quase todos os programas escritos em C. Graças a ela, podemos realizar leituras de dados digitados pelo usuário, como também exibir mensagens na tela do programa.

O stdlib.h é um cabeçalho da biblioteca em C de propósito geral padrão e possui inúmeras funções e constantes básicas. Graças a ele podemos utilizar funções de alocações de memória, controle de processos, conversões e outras funcionalidades.

Universidade Federal da Bahia

Semestre Letivo Suplementar 2020 MAT566 - Sistemas Operacionais

Componentes: Elivelton Rangel, Jane Barbosa, Mariese Conceição e Nilvan Santana

A string.h fornece funções que manipulam cadeias de caracteres. Como C não possui dados que correspondem à string, utiliza-se vetores e ponteiros. Esses vetores de caracteres terminam com o caracter nulo "\0".

O unistd.h é um cabeçalho que define diversos tipos e constantes simbólicas, além de declarar diversas funções. Por fim, as funções associadas a biblioteca sys/stat.h são utilizadas para manipular arquivos e fcntl.h permite funções de E/S de baixo nível.

Por fim o arquivo sys/wait.h contém um conjunto de macros, para testar-se o motivo do término do processo filho, bem como para se obter o código de retorno caso o processo tenha sido terminado normalmente ou o número do sinal caso o processo tenha sido cancelado.

Além disso, o programa também faz uma chamada ao arquivo function.c que possui algumas funções específicas implementadas no programa.

Os seguintes parâmetros globais foram utilizados pelo programa myshell:

- Char *PROGRAM_NAME

Além disso, define as entradas de linha de comando com tamanho máximo de 4096 caracteres, podendo manipular até 100 palavras em cada linha de comando.

A seguinte estrutura também foi criada neste programa: command (contendo a quantidade de argumentos e uma array de argumentos).

As seguintes funções foram declaradas para os comandos do builtin:

- **num_builtins:** ela retorna o tamanho de bits do arquivo command_str.
- **cmd_help:** ela utiliza como parâmetro um ponteiro para a estrutura de caracteres command. Imprime uma mensagem que manda utilizar umas funções para continuar e também os comandos de str. Ela retorna 1 para continuar a execução e utiliza os parâmetros da lista args.
- **cmd_chdir:** utiliza como parâmetro um ponteiro para a estrutura de caracteres command. Se o ponteiro retornar o número de parâmetros do arg menor ou igual a 1, ele irá imprimir a mensagem que os parâmetros insuficientes. E se o ponteiro retorna -1, ele irá imprimir uma mensagem de erro de execução do parâmetro.

Universidade Federal da Bahia

Semestre Letivo Suplementar 2020 MAT566 - Sistemas Operacionais

Componentes: Elivelton Rangel, Jane Barbosa, Mariese Conceição e Nilvan Santana

- **cmd_pwd:** utiliza como parâmetro um ponteiro para a estrutura de caracteres command. Ele cria um vetor de caracteres buff com tamanho de buffer definido como 1024. Ela utiliza a função getwd para imprimir o nome do diretório e retorna 1 para continuar a execução
- **cmd_exit:** utiliza como parâmetro um ponteiro para a estrutura de caracteres command. E retorna 0 para terminar a execução.
- **int cmd_start(struct command *cmd)**: A função start é utilizada para iniciar outro programa com argumentos de linha de comando, imprimindo o ID do processo do programa que está rodando, além de aceitar outra entrada de linha de comando.
- **int cmd_wait(struct command *cmd) e int cmd_wait_for(struct command *cmd)** são funções que não precisam de argumentos e faz com que o shell aguarde até que o processo finalize. Quando isso acontece, a função indica se o término foi normal ou não, incluindo o código de status do término ou o número e o nome do sinal respectivamente. Caso não exista processos que shell deva aguardar, o programa imprime uma mensagem e volta a aceitar novos comandos de entrada.
- **int cmd_run(struct command *cmd)**: a função run deve iniciar um programa, possivelmente com argumentos de linhas de comando, esperar que tal processo finalize e em seguida imprimir o status do término.
- **int cmd_watchdog(struct command *cmd)**: função que recebe um tempo limite (em segundos) e um comando que deve ser executado, então executa o comando da mesma forma que run. Contudo, se o comando demorar mais que o tempo limite, então o shell deve enviar um sinal SIGKILL para o processo filho, esperar até que o mesmo finalize e, então, apresentar o status da execução do mesmo.
- **Handle_error:** é uma função sem retorno que utiliza como parâmetros o status e o pid. A função imprime o status e o do processo que finalizou normalmente ou não.
- **Child_handler**: é uma função sem retorno, que utiliza como parâmetro o sinal do sistema. Se um comando ultrapassar um tempo limite, ela envia um sinal para o processo filho esperar ser finalizado através da função watchdog.
- **Alarm_handler:** é uma função sem retorno, que utiliza como parâmetro o sinal do sistema. Ela envia um alarme quando o timeout de um comando ultrapassar 1 através da função watchdog.

Universidade Federal da Bahia

Semestre Letivo Suplementar 2020 MAT566 - Sistemas Operacionais

Componentes: Elivelton Rangel, Jane Barbosa, Mariese Conceição e Nilvan Santana

- **Exec_command**: é uma função de retorno que tem como parâmetro um ponteiro para a estrutura command. Esta função duplica o processo atual dentro de um sistema operacional. Se o processo foi duplicado é chamado de processo filho. Caso o pid não seja criado ele obtém o seu próprio pid e envia uma mensagem que o processo está sendo criado. Se o ponteiro mostrar um lugar vazio será impresso uma mensagem de erro e retorna 1. Caso dê um erro na criação do pid, ele imprime a mensagem informando o erro e retorna 1.
- **int exec_command_wd**: nessa função é criado um novo processo e realizado um teste. Se o pid for igual a zero, ele deve exibir uma mensagem para o usuário e retornar o pid do processo. Se o pid for menor que zero, deve realizar uma chamada a função errno para tratar o tipo de erro.