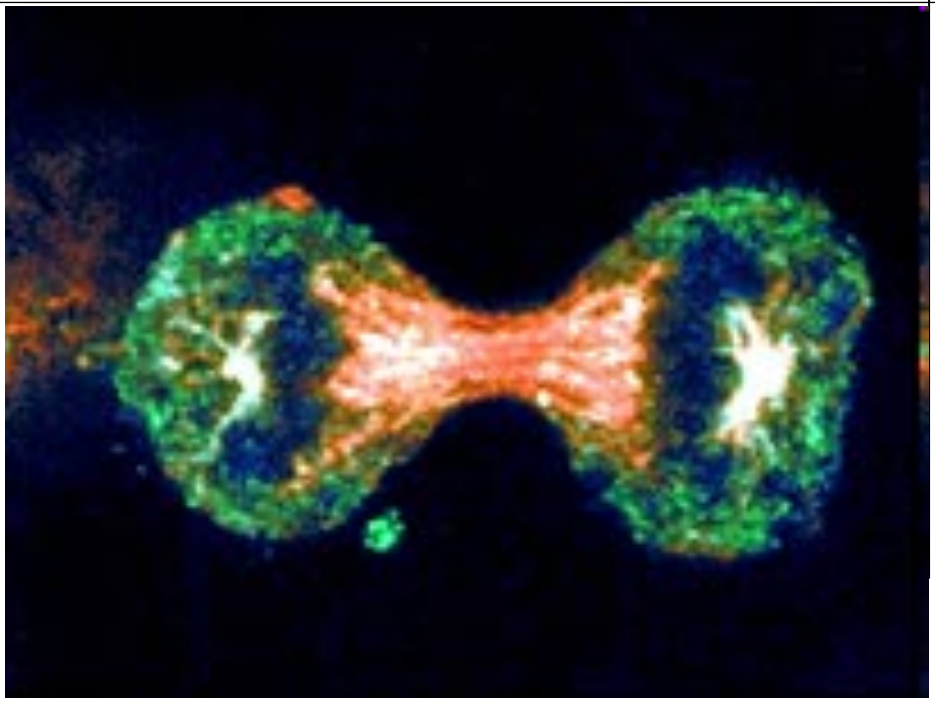
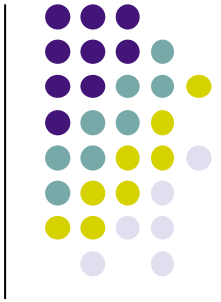
Criação de Processos

Fork( )
Exec( )



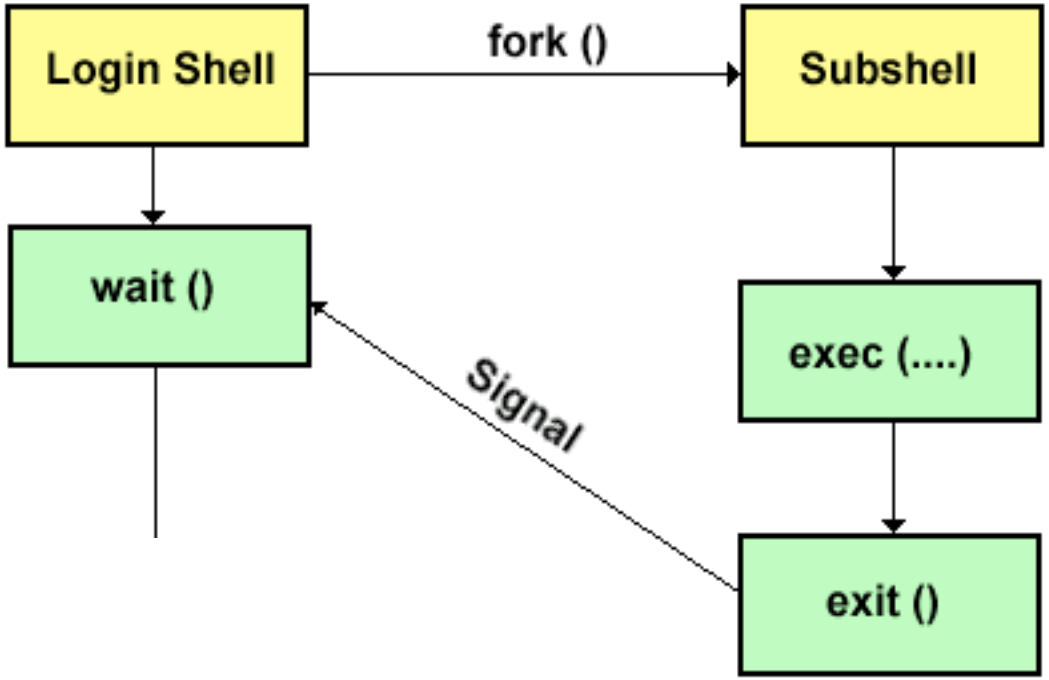
Chamadas de Sistema: Process Management



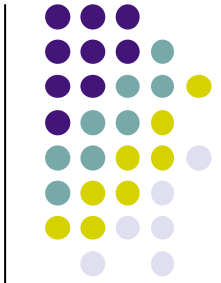
Gerenciamento de processos

Chamada	Descrição
pid = fork( )	Crie um processo filho idêntico ao processo pai
pid = waitpid(pid, &statloc, options)	Aguarde um processo filho terminar
s = execve(name, argv, environp)	Substitua o espaço de endereçamento do processo
exit(status)	Termine a execução do processo e retorne o estado

Chamada fork( ) / exec( )



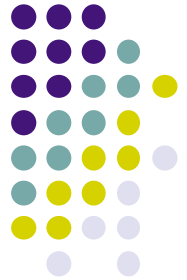
Esboço de uma shell



```
while (TRUE) {                                  /* repeat forever */
    type_prompt( );                             /* display prompt */
    read_command (command, parameters)          /* input from terminal */

if (fork() != 0) {                              /* fork off child process
    */
    /* Parent code */
    waitpid( -1, &status, 0);                   /* wait for child to exit */
} else {
    /* Child code */
    execve (command, parameters, 0);            /* execute command */
 }
}
```

As variantes de exec()



#include <unistd.h>

int execl(const char *_path_, **const char** *_arg0_, **...** _/*_, **(char *)0** _*/_**);**
int execv(const char *_path_, **char *const** _argv_**[]);**
int execle(const char *_path_, **const char** *_arg0_, **...** _/*_,
(char *)0, char *const _envp_**[]***_/_**);**
int execve(const char *_path_, **char *const** _argv_**[], char *const** _envp_**[]);**
int execlp(const char *_file_, **const char** *_arg0_, **...** _/*_, **(char *)0** _*/_**);**
int execvp(const char *_file_, **char *const** _argv_**[]);**

Entregável

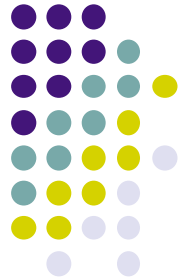
Um relatório (em um único arquivo, ASCII) c/ sufixo **txt** contendo, para cada exercício resolvido:

- Nome da Dupla
- O enunciado
- Código fonte do programa(s)
- A chamada do programa e sua saída
- Uma discusão sobre o motivo de ter obtido esse resultado

Enviar o relatório para valmeida@inf.puc-rio.br e endler@inf.puc-rio.br até 6a.feira as 23:59

Perguntas?

URL dos Labs



www.inf.puc-rio.br/~endler/courses/inf1019/transp/aulas-praticas

Exercícios

1a) Faça um programa composto por dois processos, processo pai e processo filho, onde:

- Pai - Imprime o seu pid, espera o filho terminar e imprime "Pai finalizado".
- Filho - Imprime o seu pid e o pid do seu pai, e no final imprime "Filho finalizado".

DICA: Procure na internet o comando para obter o PID (process id) de um processo.

1b) Crie processos pai e filho, e

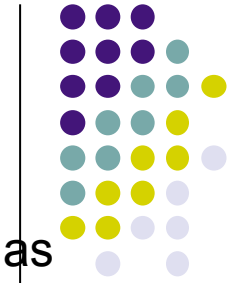
- Crie uma variável visível ao pai e ao filho iniciada com 0.
- O pai deve somar 50 a esta variável e imprimir: Pai <PID> - <valor>.
- O filho deve somar 10 a esta variável e imprimir: Filho <PID> - <valor>.
- O pai deve esperar o processo filho terminar e imprimir novamente o valor da variável: Pai após <PID> - <valor>.

O que você observou sobre o valor da variável no pai e no filho? Explique o por quê deste comportamento.

-

DICA: Para descobrir os protótipos das funções fork e waitpid execute o comando man no Terminal para acessar o manual: "man fork" e "man waitpid".

Exercícios



2) Faça um programa em que três processos q executam paralelamente as seguintes ações:

• Pai - Imprime os números de 1 a 50, com um intervalo de 2 segundos entre cada número. Após imprimir todos os números, imprime a frase “Processo pai vai morrer”.

• Filho1 - Imprime os números de 100 a 200, com um intervalo de 1 segundo entre cada número. Antes de imprimir os números, imprime a frase “Filho 1 foi criado”, e após imprimir todos os números, imprime a frase “Filho 1 vai morrer”.

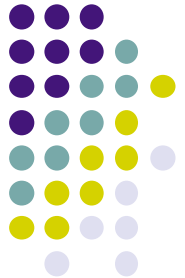
• Neto1- (filho do processo Filho1) imprime os números de 300 a 350, com um intervalo de 2 segundos entre cada número. Antes de imprimir os números, imprime a frase “Neto 1 foi criado” Após imprimir todos os números, imprime a frase “Neto 1 vai morrer”.

Importante:

- Em cada printf os processos devem imprimir o seu pid e o pid do seu pai.
- DICA: sleep(1) bloqueia o processo por 1 segundo.

É possível ver os processos executando em paralelo? Que alterações devem ser feitas em seu programa para que primeiro sejam exibidos só os prints do neto, depois só os do filho e depois só os do pai?

Exercícios



3) Crie um programa que realize o print da frase: "Alô mundo", no arquivo alomundo.c. Compile este programa.
Em seguida, crie um programa que execute o programa alomundo criado por você.

Importante:

- Utilize alguma função da família "execv" para realizar esta atividade.
- Para conhecer os protótipos das funções disponíveis execute o comando man no Terminal para acessar o manual: "man execv".

Transparências das aulas praticas na URL:

www.inf.puc-rio.br/~endler/courses/inf1019/transp