1. Crie um programa em C que cria um processo-filho e um pipe de comunicação. Faça com que o processo-pai envie os valores 1, 2, 3, 4, 5, 6, 7, 8, 9 e 10 para o processo-filho, com intervalos de 1 segundo entre cada envio. Depois de o processo-pai enviar o número 10, ele aguarda 1 segundo e termina a execução. O processo-filho escreve na tela cada valor recebido, e quando ele receber o valor 10, ele termina a execução.

#include <stdio.h>

#include <stdlib.h>

#include <signal.h>

#include <unistd.h>

#include <string.h>

int main()

{

int pid; // PID do processo filho

int fd[2]; // Descritores do pipe

int i;

// Cria o pipe

pipe(fd);

// Cria o processo

pid = fork();

// Codigo do filho

if(pid == 0)

{

while(1){

printf("Filho vai ler o pipe\n");

read(fd[0],&i,sizeof(int));

printf("Filho leu: %d\n",i);

if(i==10){

printf("Terminando processo filho\n");

exit(0);

}

}

}

// Codigo do pai

else

{

printf("Pai vai escrever no pipe\n");

for(i=1;i<=10;i++){

sleep(1);

write(fd[1],&i,sizeof(int));

}

sleep(1);

printf("Pai terminou de escrever no pipe\n");

}

return 0;

}

1. Crie um programa em C que cria um processo-filho e um pipe de comunicação. Utilize o pipe para executar a seguinte conversa entre processos:

FILHO: Pai, qual é a verdadeira essência da sabedoria?

PAI: Não façais nada violento, praticai somente aquilo que é justo e equilibrado.

FILHO: Mas até uma criança de três anos sabe disso!

PAI: Sim, mas é uma coisa difícil de ser praticada até mesmo por um velho como eu...

Neste exercício, quem recebe a mensagem via pipe é quem as escreve no terminal.

#include <stdio.h>

#include <stdlib.h>

#include <signal.h>

#include <unistd.h>

#include <string.h>

int main()

{

int pid; // PID do processo filho

char mensagem1[100] = "FILHO: Pai, qual é a verdadeira essência da sabedoria?\n";

char mensagem2[100] = "PAI: Não façais nada violento, praticai somente aquilo que é justo e equilibrado.\n";

char mensagem3[100] = "FILHO: Mas até uma criança de de três anos sabe disso!\n";

char mensagem4[100] = "PAI: Sim, mas é uma coisa difícil de ser praticada até mesmo por um velho como eu...\n";

int fd[2]; // Descritores do pipe

int i;

// Cria o pipe

pipe(fd);

// Cria o processo

pid = fork();

// Codigo do filho

if(pid == 0)

{

write(fd[1],mensagem1,100);

sleep(1);

read(fd[0],mensagem2,100);

printf("%s",mensagem2);

sleep(1);

write(fd[1],mensagem3,100);

sleep(1);

read(fd[0],mensagem4,100);

printf("%s",mensagem4);

sleep(1);

}

// Codigo do pai

else

{

read(fd[0],mensagem1,100);

printf("%s",mensagem1);

sleep(1);

write(fd[1],mensagem2,100);

sleep(1);

read(fd[0],mensagem3,100);

printf("%s",mensagem3);

sleep(1);

write(fd[1],mensagem4,100);

sleep(1);

}

return 0;

}

1. Crie um programa em C que cria dois processos-filhos e um pipe de comunicação. Utilize o pipe para executar a seguinte conversa entre processos:

FILHO1: Quando o vento passa, é a bandeira que se move.

FILHO2: Não, é o vento que se move.

PAI: Os dois se enganam. É a mente dos senhores que se move.

Neste exercício, quem recebe a mensagem via pipe é quem as escreve no terminal.

#include <stdio.h>

#include <stdlib.h>

#include <signal.h>

#include <unistd.h>

#include <string.h>

int main()

{

int pid1,pid2; // PID do processo filho

char mensagem1[100] = "FILHO1: Quando o vento passa, é a bandeira que se move.\n";

char mensagem2[100] = "FILHO2: Não, é o vento que se move.\n";

char mensagem3[100] = "PAI: Os dois se enganam. É a mente dos senhores que se move.\n";

int fd[2]; // Descritores do pipe

// Cria o pipe

pipe(fd);

// Cria o processo

pid1 = fork();

if(pid1 == 0)

{

//Filho 1

sleep(1);

read(fd[0],mensagem1,100);

printf("%s",mensagem1);

write(fd[1],mensagem2,100);

sleep(1);

}

else

{

pid2 = fork();

if(pid2 == 0){

//Filho 2

write(fd[1],mensagem1,100);

sleep(1);

read(fd[0],mensagem2,100);

printf("%s",mensagem2);

sleep(1);

write(fd[1],mensagem3,100);

sleep(1);

}

else{

//Pai

sleep(2);

read(fd[0],mensagem3,100);

printf("%s",mensagem3);

sleep(1);

}

}

return 0;

}

1. Crie um programa em C que cria um processo-filho e um pipe de comunicação. O processo-filho deverá pedir o nome do usuário, envia-lo para o pai via pipe, e o pai deverá escrever o nome do usuário no terminal.

#include <stdio.h>

#include <stdlib.h>

#include <signal.h>

#include <unistd.h>

#include <string.h>

int main()

{

int pid; // PID do processo filho

char usuario[30];

char buf[30];

int fd[2]; // Descritores do pipe

// Cria o pipe

pipe(fd);

// Cria o processo

pid = fork();

// Codigo do filho

if(pid == 0)

{

printf("Insira o nome do usuario: ");

gets(usuario);

write(fd[1],usuario,30);

}

// Codigo do pai

else

{

read(fd[0],buf,30);

printf("Usuario lido: %s\n",buf);

}

return 0;

}

1. Utilize o sinal de alarme para chamar a cada segundo o comando ps ordenando todos os processos de acordo com o uso da CPU. Ou seja, seu código atualizará a lista de processos a cada segundo. Além disso, o código deverá tratar o sinal do CTRL-C, escrevendo "Processo terminado!" na tela antes de terminar a execução do processo.

#include <signal.h>

#include <unistd.h>

#include <stdio.h>

#include <stdlib.h>

void tratamento\_alarme(int sig)

{

system("ps aux --sort=-pcpu");

alarm(1);

}

void sigint\_handler(int signum)

{

fprintf(stderr, "\nProcesso terminado!\n");

exit(0);

}

int main()

{

signal(SIGALRM, tratamento\_alarme);

signal(SIGINT, sigint\_handler);

alarm(1);

printf("Aperte CTRL+C para acabar:\n");

while(1);

return 0;

}