1. Crie um código em C para gerar três processos-filho usando o fork().

#include <stdio.h>

#include <stdlib.h>

#include <sys/types.h>

#include <unistd.h>

int main(int argc, char \*argv[]){

pid\_t child\_1, child\_2, child\_3;

child\_1 = fork();

if(child\_1 == 0){

printf("\* Este texto foi escrito no terminal pelo processo FILHO 1 (ID=%d) \*\n", (int) getpid());

}

else{

child\_2 = fork();

if(child\_2 == 0){

printf("\* Este texto foi escrito no terminal pelo processo FILHO 2 (ID=%d) \*\n", (int) getpid());

}

else{

child\_3 = fork();

if(child\_3 == 0){

printf("\* Este texto foi escrito no terminal pelo processo FILHO 3 (ID=%d) \*\n", (int) getpid());

}

else{

sleep(1);

printf("\* Este texto foi escrito no terminal pelo processo PAI (ID=%d) \*\n", (int) getpid());

printf("\* Seus filhos tem o ID %d, %d, %d. \*\n",child\_1,child\_2,child\_3);

}

}

}

return 0;

}

1. Crie um código em C que recebe o nome de diversos comandos pelos argumentos de entrada (argc e \*argv[]), e executa cada um sequencialmente usando system(). Por exemplo, considerando que o código criado recebeu o nome de 'serial\_system', e que ele foi executado na pasta '/Sistemas\_Embarcados/Code/06\_Processos', que contem diversos arquivos:

$ ./serial\_system pwd echo ls echo cal

$ ~/Sistemas\_Embarcados/Code/06\_Processos

$

$ Ex1.c Ex2b.c Ex4.c Ex6.c

$ Ex2a.c Ex3.c Ex5.c serial\_system

$

$ Março 2017

$ Do Se Te Qu Qu Se Sá

$ 1 2 3 4

$ 5 6 7 8 9 10 11

$ 12 13 14 15 16 17 18

$ 19 20 21 22 23 24 25

$ 26 27 28 29 30 31

#include <stdio.h>

#include <stdlib.h>

#include <sys/types.h>

#include <unistd.h>

int main(int argc, char \*argv[]){

int i;

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

system(argv[i]);

}

return 0;

}

1. Crie um código em C que recebe o nome de diversos comandos pelos argumentos de entrada (argc e \*argv[]), e executa cada um usando fork() e exec().

#include <stdio.h>

#include <stdlib.h>

#include <sys/types.h>

#include <unistd.h>

#include <string.h>

int main(int argc, char \*\*argv) {

int i;

pid\_t pid\_filho;

char \*lista\_de\_argumentos[2] = {"command-line",NULL};

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

pid\_filho = fork();

if(pid\_filho == 0){

sleep(1);

execvp(argv[i],lista\_de\_argumentos);

}

else{}

}

return 0;

}

1. Crie um código em C que gera três processos-filho usando o fork(), e que cada processo-filho chama a seguinte função uma vez:

int v\_global = 0; // Variavel global para este exemplo

void Incrementa\_Variavel\_Global(pid\_t id\_atual)

{

v\_global++;

printf("ID do processo que executou esta funcao = %d\n", id\_atual);

printf("v\_global = %d\n", v\_global);

}

#include <stdio.h>

#include <stdlib.h>

#include <sys/types.h>

#include <unistd.h>

int v\_global = 0; // Variavel global para este exemplo

void Incrementa\_Variavel\_Global(pid\_t id\_atual)

{

v\_global++;

printf("ID do processo que executou esta funcao = %d\n", id\_atual);

printf("v\_global = %d\n", v\_global);

}

int main(int argc, char \*argv[]){

pid\_t child\_1, child\_2, child\_3;

child\_1 = fork();

if(child\_1 == 0){

Incrementa\_Variavel\_Global((int)getpid());

}

else{

child\_2 = fork();

if(child\_2 == 0){

Incrementa\_Variavel\_Global((int)getpid());

}

else{

child\_3 = fork();

if(child\_3 == 0){

Incrementa\_Variavel\_Global((int)getpid());

}

else{

sleep(1);

printf("Este texto foi escrito no terminal pelo processo PAI (ID=%d)\n", (int) getpid());

}

}

}

return 0;

}

(Repare que a função Incrementa\_Variavel\_Global() recebe como entrada o ID do processo que a chamou.) Responda: a variável global v\_global foi compartilhada por todos os processos-filho, ou cada processo enxergou um valor diferente para esta variável?

A variável global foi compartilhada por todos os processos filhos.

1. Repita a questão anterior, mas desta vez faça com que o processo-pai também chame a função Incrementa\_Variavel\_Global(). Responda: a variável global v\_global foi compartilhada por todos os processos-filho e o processo-pai, ou cada processo enxergou um valor diferente para esta variável?

#include <stdio.h>

#include <stdlib.h>

#include <sys/types.h>

#include <unistd.h>

int v\_global = 0; // Variavel global para este exemplo

void Incrementa\_Variavel\_Global(pid\_t id\_atual)

{

v\_global++;

printf("ID do processo que executou esta funcao = %d\n", id\_atual);

printf("v\_global = %d\n", v\_global);

}

int main(int argc, char \*argv[]){

pid\_t child\_1, child\_2, child\_3;

child\_1 = fork();

if(child\_1 == 0){

Incrementa\_Variavel\_Global((int)getpid());

}

else{

child\_2 = fork();

if(child\_2 == 0){

Incrementa\_Variavel\_Global((int)getpid());

}

else{

child\_3 = fork();

if(child\_3 == 0){

Incrementa\_Variavel\_Global((int)getpid());

}

else{

sleep(1);

Incrementa\_Variavel\_Global((int)getpid());

}

}

}

return 0;

}