**ATIVIDADE DE LABORATÓRIO - Programação Paralela e Distribuída**

**Objetivo:**  relembrar o funcionamento de criação de processos e comunicação entre processos em uma mesma máquina (durante o semestre veremos como fazer o mesmo com processos em máquinas diferentes).

**Atividade:**Implementar um programa que leia números do teclado e cria dois processos dois filhos utilizando a chamada de sistema *fork*. Depois o processo pai envia cada número digitado para um dos filhos. O filho deve dizer se o número é primo ou não. Quando o filho termina, envia uma resposta informando ao pai que está pronto para receber novo trabalho. O filho imprime o número e se ele é ou não primo.

**Entrega:** submeter o programa fonte (em C) até o final do dia pelo Moodle. No próprio programa, colocar nos comentários, o nome do autor, a data e o que o programa faz.

**OBS:** No verso tem um exemplo de programa com fork e pipe.

/\*\* Programa que cria processo pesado usando fork. Usa pipe para comunicação entre os processos. Processo filho faz algo (no exemplo ele só dorme) e envia o tempo que levou para fazer este algo para o pai. Além disto cada um incrementa uma variável i.

Author: Avelino F. Zorzo [avelino.zorzo@pucrs.br](mailto:avelino.zorzo@pucrs.br)

\*\*/

#include <stdio.h>

#include <unistd.h>

#include <time.h>

#include <string.h>

#include <sys/types.h>

#define TAMMSG 80

int main() {

int fd[2], nbytes, i = 0;

pid\_t pid;

char string[TAMMSG], buffer[TAMMSG];

time\_t begin\_t, end\_t;

pipe(fd);

if ( (pid = fork()) == -1) {

printf("Erro ao criar processo = %d.\n",pid);

return(1);

}

if(pid == 0) { /\* Código para o processo filho. \*/

begin\_t = time(NULL);

printf("Filho iniciou ..\n");

sleep(3); /\* Faz algo, no caso dorme. \*/

end\_t = time(NULL);

sprintf(string, "%3.0f", difftime(end\_t,begin\_t));

write(fd[1], string, (strlen(string)+1));

i++;

printf("Filho terminou: valor de i é %d\n", i);

}

else { /\* Código para o processo pai.\*/

/\* Leia algo que foi enviado pelo filho. \*/

nbytes = read(fd[0], buffer, 8);

printf("Pai: recebeu: %s (%d)\n", buffer, nbytes);

i++;

printf("Pai: valor de i é %d\n", i);

}

return(0);

}