



Nome : Rubio Torres Castro Viana  
 Matéria : Sistemas operacionais  
 Matrícula : 201622040350  
 Data : 02/04/2019  
 Professor : Dr.Bruno André

## Comunicação entre processos

### 1 OBJETIVO

Implemente uma solução para o seguinte problema: contar o número de ocorrências de um caracter em vetor de caracteres de 1GB. O vetor deve ser gerado por um processo P1. Outros N processos, não filhos de P1, deverão contar as ocorrências do caracter em uma parte (de tamanho 1GB/N) do vetor. O processo P1 será responsável por informar aos N processos qual o caracter deve ser contado e exibir o resultado total da contagem.

### 2 PARÂMETROS

Foi optado por criar um servidor no qual ira ser gerador da memoria compartilhada e clientes que irão repartir essa memoria entre eles.

### 3 PROGRAMAS

//SERVIDOR\_\_\_\_\_

*/\*\*\*\*\*Processo "servidor" da memória compartilhada  
 \*\*\*\*Autor: Rubio Torres\*/*

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <fcntl.h>
#include <time.h>
#include <sys/shm.h>
#include <sys/stat.h>
#include <sys/mman.h>
#include <math.h>
#include <unistd.h>

int main(){

    const int SIZE = pow(1024,3);
    const char *name = "shared_memory";

    char char_requerido;
    int shm_fd,tamanho,contador=0;//contadora=0;
    int *ptr, *start;

    //ABERTURA DA MEMORIA COMO CREATOR OU READ & WRITE
    shm_fd = shm_open(name, O_CREAT | O_RDWR, 0666);

    ftruncate(shm_fd,1 + sizeof(int)*SIZE);

    //Mapear memória
    ptr = mmap(0,sizeof(int)*SIZE, PROT_READ | PROT_WRITE, M

```

```

        if (ptr == MAP_FAILED) {
            printf("Map failed\n");
            return -1;
        }

//SALVANDO A POSIÇÃO INICIAL DA MEMORIA
    start = ptr;

//AJUSTANDO AS VARIÁVEIS A FIM DE PESQUISA
    printf("Qual o char: ");
    scanf("%c",&char_requerido);
    printf("Quantos processos: ");
    scanf("%d",&tamanho);
    *ptr=(char)char_requerido;
    ptr++;
    *ptr=-1;
    ptr++;

//ESCREVE NA MEMORIA
    for(int j=0,i=1, processo=1;j<SIZE+1;j++,ptr++,i++){
        *ptr=(char)97 + rand() % 26;
        /*-----
        Verifica resposta
        if(*ptr==char_requerido){
            contadora++;
        }
        -----

```

```

//Faz controle da divisão da memoria
if(j==(SIZE/tamanho)*processo){
    printf("Esperando processo...\n");
    processo++;

//Espera o processo responder
    while(1){
        *ptr='\0';
        if(start[1]!=-1){

            //Soma resposta
            contador+=(long int)start[1];

            //Reinicia memoria
            ptr=start;
            *ptr=(char)char_requerido;
            ptr++;
            *ptr=-1;
            ptr++;
            break;
        }
    }
}

}

//MOSTRA A RESPOSTA
printf("A letra '%c' teve %d ocorrência(s)\n",char_requerido,contador);

```

```

        return 0;
    }

    /******Processo "cliente" da memória compartilhada
    *****Autor: Rubio Torres*/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <fcntl.h>
#include <sys/shm.h>
#include <sys/stat.h>
#include <sys/mman.h>
#include <math.h>
int main(){
    //Inicializa variaves
    const char *name = "shared_memory";
    const int SIZE = pow(1024,3);

    int shm_fd;
    int *start;
    int *ptr;
    int i,j;
    long int cont=0;

    //Inicializa leitura/escrita da memoria
    shm_fd = shm_open(name, O_RDWR, 0666);
    if (shm_fd == -1) {
        printf("shared memory failed\n");
        exit(-1);
    }

```

```

}

//Mapeia memória
ptr = mmap(0,SIZE*sizeof(int), PROT_READ | PROT_WRITE,
if (ptr == MAP_FAILED) {
    printf("Map failed\n");
    exit(-1);
}

//Salva começo da memória
start=ptr;

//Pula os dois primeiros termos
ptr+=2;

//Lê a memória
for(j=3;(char)ptr[j]!='\0';j++){

    //Faz comparação
    if((char)ptr[j]==(char)*start){
        cont++;
    }
}

// Escreve resultado na memória
*(start+1)=(long int)cont;

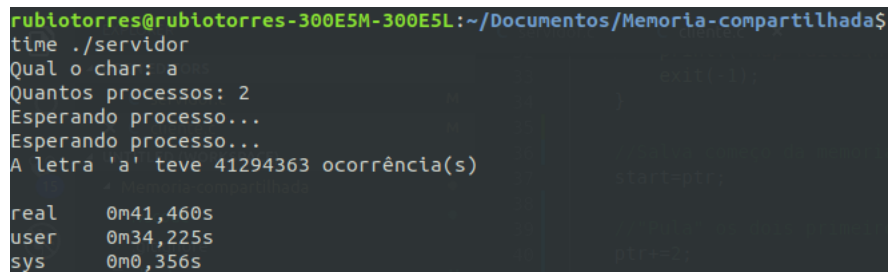
printf("Contagem registrada.\n");

```

```
    return 0;  
}
```

## 4 EXECUÇÃO

Primeiramente se compila e executa o servidor, que questionará sobre o char que deve ser procurado e o numero de processos que irá entrar em ação, após isso ele faz a divisão da informação, carrega a parte por parte e distribui nos processos a medida que ele vai sendo demandado.



```
rubiotorres@rubiotorres-300E5M-300E5L:~/Documentos/Memoria-compartilhada$  
time ./servidor  
Qual o char: a  
Quanto processos: 2  
Esperando processo...  
Esperando processo...  
A letra 'a' teve 41294363 ocorrência(s)  
  
real    0m41,460s  
user    0m34,225s  
sys     0m0,356s
```

Figura 1: Execução