Листинг:

#include <unistd.h>

#include <sys/types.h>

#include <stdio.h>

#include <stdlib.h>

#include <sys/ipc.h>

#include <sys/shm.h>

#include <sys/wait.h>

int sieve\_of\_eratosthenes(int \*\*primes, const int min, const int max) {

int \*isPrime = (int \*)malloc(sizeof(int) \* (max + 1));

int i;

for(i = 0; i <= max; ++i)

isPrime[i] = 1;

isPrime[0] = 0;

isPrime[1] = 0;

int count = 0;

for(i = 0; i <= max; ++i)

{

if(isPrime[i] == 1)

{

++count;

int j;

for(j = 2 \* i; j <= max; j += i)

isPrime[j] = 0;

}

}

\*primes = (int \*)malloc(sizeof(int) \* count);

int j = 0;

for(i = min; i <= max; ++i)

{

if(isPrime[i] == 1)

{

(\*primes)[j++] = i;

}

}

return count;

}

int main() {

int \*primes = NULL;

int size, size2;

int \*shm, \*s, shmid, count = 10;

/\* Создадим область разделяемой памяти \*/

if ((shmid = shmget(IPC\_PRIVATE, size, IPC\_CREAT | 0666)) < 0) {

perror("shmget");

exit(1);

}

/\* Получим доступ к разделяемой памяти \*/

if ((shm = shmat(shmid, NULL, 0)) == (int \*) -1) {

perror("shmat");

exit(1);

}

printf(" СТАРТ, ПРОЦЕСС C ID %d\n\n", getpid());

for (int i = 1; i < 3; i++){

sleep (rand()%4);

if (fork() == 0){

if(i==1){

s = shm;

printf("<-- ПЕРВЫЙ ПОДСЧЕТ В ИНТЕРВАЛЕ ОТ 10 ДО 40 -->\n", i, getpid());

int size = sieve\_of\_eratosthenes(&primes,10, 40);

for (int i = 0; i < 2; ++i){

s[i] = size; //\*s++ = i;

}

for (int i = 0; i < 1; ++i){

printf("Значение в %d сегменте памяти: %d\n", i,s[i]);

}

if (shmdt(shm) < 0) {

printf("Ошибка отключения\n");

exit(1);

}

printf(" -- ГОТОВО --%d НАЙДЕНО \n\n", size, getppid());

}

else

{

s = shm;

printf("<-- ВТОРОЙ ПОДСЧЕТ В ИНТЕРВАЛЕ ОТ 50 ДО 90 -->\n", i, getpid());

int size2 = sieve\_of\_eratosthenes(&primes,50, 90);

for (int i = 1; i < 3; ++i){

s[i] = size2; //\*s++ = i;

}

for (int i = 0; i < 2; ++i){

printf("Значение в %d сегменте памяти: %d\n", i,s[i]);//s[i]\*s[i] = \*s\*\*s++

}

if (shmdt(shm) < 0) {

printf("Ошибка отключения\n");

exit(1);

}

printf("-- ГОТОВО --%d НАЙДЕНО \n\n", size2, getppid());

}

exit(0);

}

}

for(int i=1; i<=3; i++)

wait(NULL);

printf("ЗАВЕРШЕНО \n" ,getpid());

}

Описание:

Программа ведет запись и чтение в разделяемой памяти, предварительно создавая область этой памяти. Запись ведется в двух порожденных процессах, чтение соответственно. Программа ведет поиск простых чисел в двух интервалах, заранее задаваемых в коде (здесь 10-40 и 50-90) через функцию sieve\_of\_eratosthenes(). Поиск в каждом интервале ведется в двух разных порожденных процессах. Каждый процесс записывает в область памяти и далее считывает значение из этой памяти.

Результат:

