Листинг:

#include <unistd.h>

#include <sys/types.h>

#include <stdio.h>

#include <stdlib.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;

printf(" NACHALO PROGRAMY, PROCESS %d\n\n", getpid());

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

sleep (rand()%4);

if (fork() == 0){

if(i==1){

printf("<-- POSWITAL PERVYI RAZ INTEVAL 10-40 -->\n", i, getpid());

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

printf("-- GOTOVO --%d NIDENO \n\n", size, getppid());

}

else{

printf("<-- POSWITAL VTOROI RAZ INTEVAL 50-90 -->\n", i, getpid());

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

printf("-- GOTOVO --%d NAIDENO \n\n", size2, getppid());

}

exit(0);

}

}

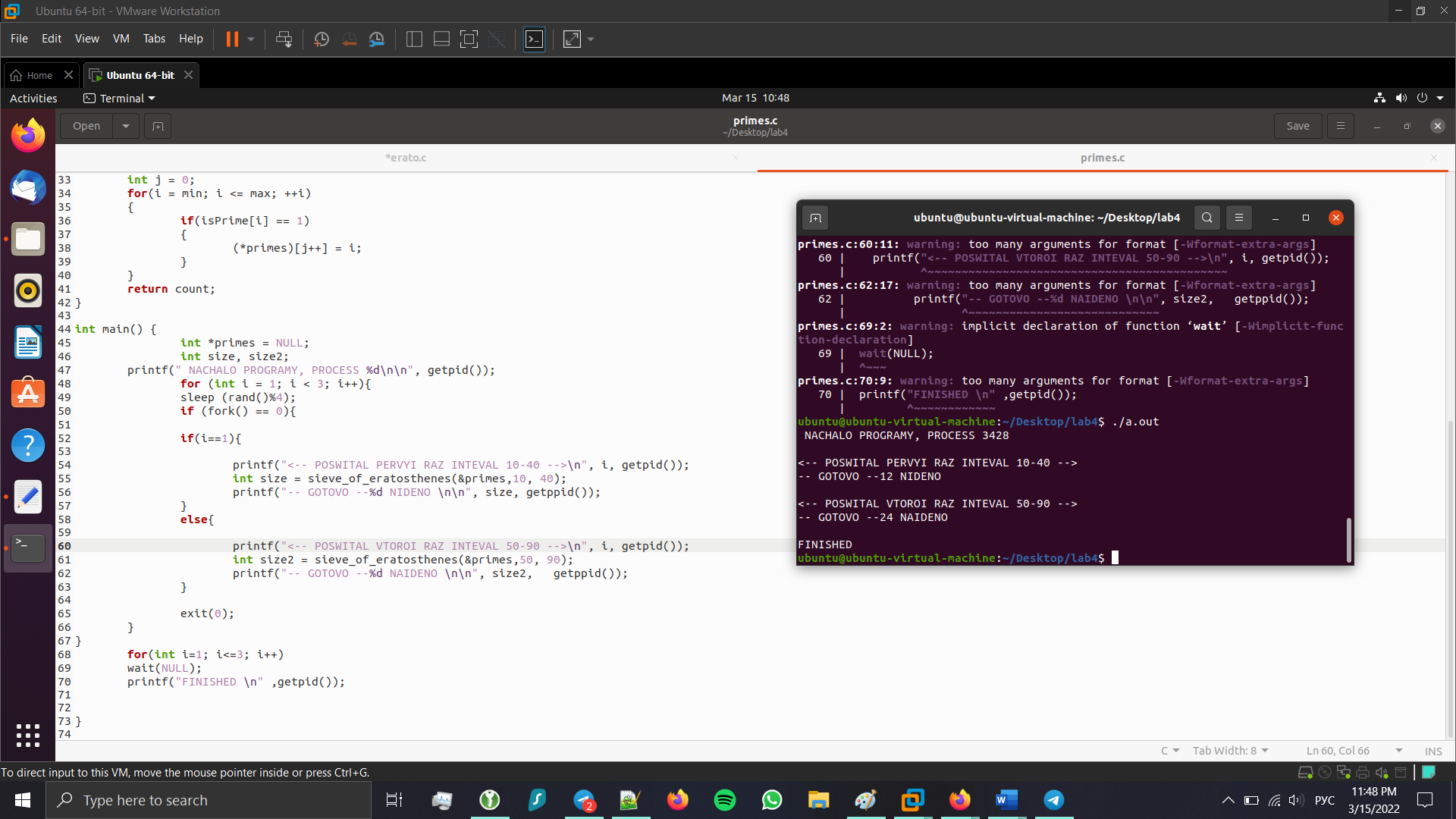
for(int i=1; i<=3; i++)

wait(NULL);

printf("FINISHED \n" ,getpid());

}

Результат выполнения программы:



Описание:

Программа выполняет поиск простых чисел посредством алгоритма Решето Эратосфена. Алгоритм занимает большую часть кода.

Поиск производится в 2 интервалах: от 10 до 40, от 50 до 90.

Запуская на каждый интервал 2 процесса: поиск в диапазоне и вывод результата. Таким образом запускается 4 процесса: первые 2 затем оставшиеся 2 процесса. Подходов таким образом 2 для каждого интервала.