**Practica 16**

**Num Factorial**

[**https://www.codechef.com/problems/NUMFACT**](https://www.codechef.com/problems/NUMFACT)

**Explicacion de problema:**

Alice has learnt factorization recently. Bob doesn't think she has learnt it properly and hence he has decided to quiz her. Bob gives Alice a very large number and asks her to find out the number of factors of that number. To make it a little easier for her, he represents the number as a product of N numbers. Alice is frightened of big numbers and hence is asking you for help. Your task is simple. Given N numbers, you need to tell the number of distinct factors of the product of these N numbers.

**Explicacion de codigo:**

Obtengo los factores primos, sacando el residuo de todos los numeros hasta a, si da igual a 0 signfica que si es factor. Se divide a hasta que ya no sea factor y se guarda el numero de veces que aparece el primo en la composicion, al final se usa la formula de multiplicacion de # de veces que aparece el primo + 1 en todos los numeros y eso da igual al # de divisores de ese numero.

**Explicacion de conversion a servicio:**

- Crear codigo daemon que este corriendo todo el tiempo

- Crear fucncion de signal para correr un codigo en especifico

- insertar codigo en el switch para que corra

- listo

**Codigo Daemon:**

**//** Andres Aguirre Gonzalez

// A01039656

// NUMFACT

// servicio de linux

#include <stdio.h>

#include <fcntl.h>

#include <signal.h>

#include <unistd.h>

#include <string.h>

#include <stdlib.h>

#define RUNNING\_DIR "/github/c-programs/P15/"

#define LOCK\_FILE "exampled.lock"

#define LOG\_FILE "lista.txt"

void log\_message(char \*filename, char \*message) {

FILE \*logfile;

logfile=fopen(filename,"a");

if(!logfile) return;

fprintf(logfile,"%s\n",message);

fclose(logfile);

}

static void signalHandler(int sig) {

switch(sig) {

case SIGUSR1:

log\_message(LOG\_FILE,"USR1 signal catched");

int exponente[1000000];

int size;

int t;

static const char filename[] = "input.txt";

FILE \*file = fopen(filename, "r");

char line[256]; /\* or other suitable maximum line size \*/

fgets(line, sizeof line, file);

t = atoi(line);

size = t;

int i;

int z = 0;

int results[2];

while (t--) {

int n;

fgets(line, sizeof line, file);

n = atoi(line);

memset(exponente, 0, sizeof exponente);

while (n--) {

int a;

fgets(line, sizeof line, file);

a = atoi(line);

int r = sqrt(a);

for (i=2; i<=r; i++) {

if(a % i == 0) {

while (a % i == 0) {

a /= i;

exponente[i] ++;

}

r = sqrt(a);

}

}

if (a > 1) exponente[a] ++;

}

long long res = 1;

for(i=2; i<1000000; i++){

res \*= (exponente[i] + 1);

}

results[z] = res;

z++;

}

fclose(file);

file = fopen("results.txt", "w+");

for (int r = 0; r < size; r++) {

sprintf(line, "%d", results[r]);

char \*tmp = line;

log\_message(LOG\_FILE, tmp);

fprintf(file, "%s\n", tmp);

}

fclose(file);

break;

case SIGUSR2:

log\_message(LOG\_FILE,"USR2 signal catched");

exit(0);

break;

}

}

void daemonize() {

int i,lfp;

char str[10];

// if(getppid()!=1) return; /\* already a daemon \*/

i=fork();

if (i<0) exit(1); /\* fork error \*/

if (i>0) exit(0); /\* parent exits \*/

/\* child (daemon) continues \*/

setsid(); /\* obtain a new process group \*/

for (i=getdtablesize();i>=0;--i) close(i); /\* close all descriptors \*/

i=open("/dev/null",O\_RDWR); dup(i); dup(i); /\* handle standart I/O \*/

umask(027); /\* set newly created file permissions \*/

chdir(RUNNING\_DIR); /\* change running directory \*/

lfp=open(LOCK\_FILE,O\_RDWR|O\_CREAT,0640);

if (lfp<0) exit(1); /\* can not open \*/

if (lockf(lfp,F\_TLOCK,0)<0) exit(0); /\* can not lock \*/

/\* first instance continues \*/

sprintf(str,"%d\n",getpid());

write(lfp,str,strlen(str)); /\* record pid to lockfile \*/

signal(SIGCHLD,SIG\_IGN); /\* ignore child \*/

signal(SIGTSTP,SIG\_IGN); /\* ignore tty signals \*/

signal(SIGTTOU,SIG\_IGN);

signal(SIGTTIN,SIG\_IGN);

signal(SIGUSR1,signalHandler); /\* catch hangup signal \*/

signal(SIGUSR2,signalHandler); /\* catch kill signal \*/

}

int main() {

int i=0;

FILE \*arch;

daemonize();

while(1) {

arch=fopen("lista.txt","a+");

if(arch==NULL)

return(1);

i++;

fprintf(arch,"%i.- I'm printing this to a file \n",i);

fclose(arch);

sleep(5);

} /\* run \*/

return(0);

}

**Evidencia:** <https://www.youtube.com/watch?v=IzwSSwQLwbQ&feature=youtu.be>