INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO
SISTEMAS OPERATIVOS
GRUPO: 2CM8

INTEGRANTES:
MENDOZA FLORES ERICK GABRIEL
NAVA VIVAS ANA PAOLA

PRACTICA 4: HILOS

La práctica consiste en realizar la multiplicación de matrices, pero utilizando hilos. Está dividida en dos partes:
a) No se pueden declarar matrices globales, por lo tanto el padre, deberá indicarle a cada hilo las filas que les tocan, una vez que los hilos terminen la multiplicación de sus filas, deberán de regresar los resultados hacia el padre para que este los almacene en la matriz resultante y despues la imprima.
b) En este inciso, se deben usar matrices globales.

Sin variables Globales

```
struct matrixbit{
  int id,ini,fin;
  int **A,**B,**res;
  int colsB,cA_rB,prntrange;
};
```

Estructura:
Se define la estructura sobre la cuál van a viajar los datos de cálculo, inlcuyendo la matriz resultante, que será solo un fragmento de la matriz resultante (que cambia con cada hilo)

```
  //Contenedor para los valores de retorno
  struct matrixbit ** mtxres = malloc(hilos*sizeof(struct matrixbit *));;
  //Recuperar valores de retorno
  for(int i = 0; i < hilos; i++) pthread_join(thread[i],(void **)&mtxres[i]);
  //Ordenar valores de retorno para impresión
  for(int i = 0, k=0; i < hilos; i++){
    if(mtxres[k]->id-1==i){for(int it=0;it<mtxres[k]->prntrange+1;it++){
        for(int it1=0;it1<colsB;it1++)printf("%d ",mtxres[k]->res[it][it1]);
        puts("");}k=0;}
    else{i--;k++;}};
```

Rutina:
Se define la función a ejecutar por cada hilo.

```
void * procCalc(void * mat)
{
  struct matrixbit * mtx = (struct matrixbit *) mat;
  int it1,it2,it3,accum;
  for(mtx->ini,it3=0;mtx->ini<=mtx->fin;mtx->ini++,it3++)
    {for(it1=0;it1<mtx->colsB;it1++){
        for(it2=0,accum=0;it2<mtx->cA_rB;it2++)
          accum+=mtx->A[it3][it2]*mtx->B[it2][it1];
        mtx->res[it3][it1]=accum;}}
  pthread_exit((void*)mtx);
}
```

Recuperardatos:
Finalmente, se recuperan los datos en estructuras contenedoras y por medio de un for y del id de las estructuras se imprimen los resultados en el orden correcto.

Con variables globales

Primero se declara una estructura con dos datos de tipo entero. Se declara un prototipo de función vacía de tipo puntero, que será la encargada de realizar la multiplicación de matrices. Se declaran también las variables globales que vendrán siendo las matrices.

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <unistd.h>
4  #include <pthread.h>
5  #include <time.h>
6
7  typedef struct matricita
8  {
9    int ini,fin;}mt;
10
11 void *multiplicacion(void* arg);
12
13 int **matrizA;
14 int **matrizB;
15 int **matrizC;
16 int NFA,NCA,NFB,NCB;
17
18 int main(void)
19 {
20   int num,numa,i,j,k,l,nump,tareas,residuo;
21   pid_t pid;
22   printf("\nnumero de filas matriz A: ");
23   scanf("%d",&NFA);
24   printf("\nnumero de columnas A= numero de filas B: ");
25   scanf("%d",&NCA);
26   NFB=NCA;
27   printf("\nnumero de columnas B: ");
28   scanf("%d",&NCB);
```

En el main se van a capturar los tamaños de las matrices A y B, y luego se van a aplicar con malloc, es por eso que se habían declarado como punteros dobles. Después se van a llenar usando random.

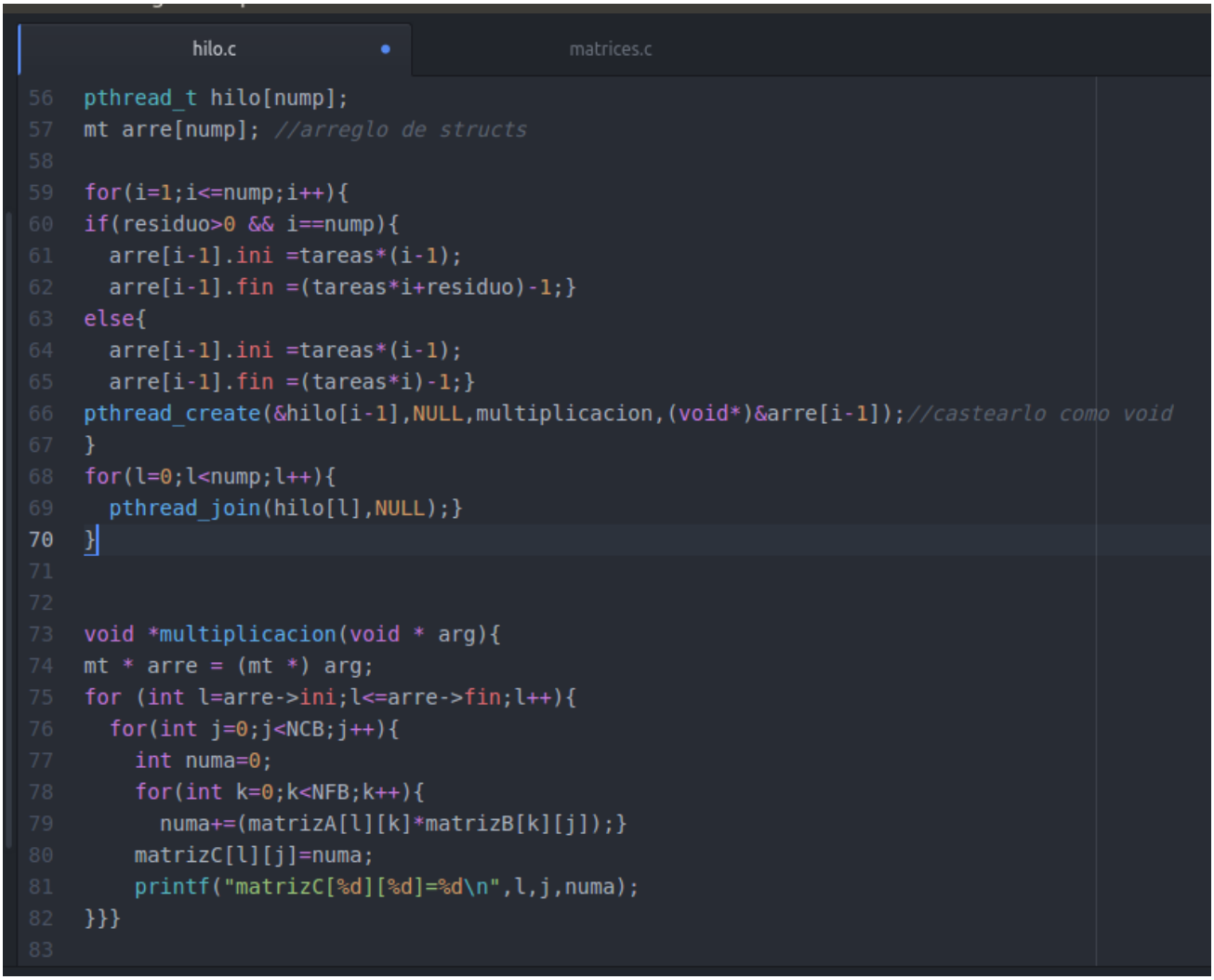
```
29    printf("\nnumero de procesos/hilos:\n");
30    scanf("%d",&nump);
31    tareas=NFA/nump;
32    residuo=NFA%nump;
33
34    matrizA=malloc(NFA*sizeof(int*));//al numero de filas le doy un numero indeterminado de espacios
35      for(i=0;i<NFA;i++){//a cada fila, le doy
36        matrizA[i]=malloc(NCA*sizeof(int));//el numero de columnas que es
37      }
38    matrizB=malloc(NFB*sizeof(int*));//al numero de filas le doy un numero indeterminado de espacios
39      for(i=0;i<NFB;i++){//a cada fila, le doy
40        matrizB[i]=malloc(NCB*sizeof(int));//el numero de columnas que es
41      }
42    matrizC=malloc(NFA*sizeof(int*));//al numero de filas le doy un numero indeterminado de espacios
43      for(i=0;i<NFA;i++){//a cada fila, le doy
44        matrizC[i]=malloc(NCB*sizeof(int));//el numero de columnas que es
45      }
46
47       for (i = 0; i < NFA; i++){//lleno la matriz A
48          for (j = 0; j < NCA; j++){
49            matrizA[i][j]=1+rand()%(9);
50        printf("matrizA[%d][%d]: %d \n",i,j,matrizA[i][j]);}}
51       for (i = 0; i < NFB; i++){//lleno la matriz B
52          for (j = 0; j < NCB; j++){
53            matrizB[i][j]=1+rand()%(9);
54        printf("matrizB[%d][%d]: %d \n",i,j,matrizB[i][j]);}}
55
```

Se va a declarar un arreglo de hilos y otro de structs, con tamaño igual al número total de procesos que se escogió.

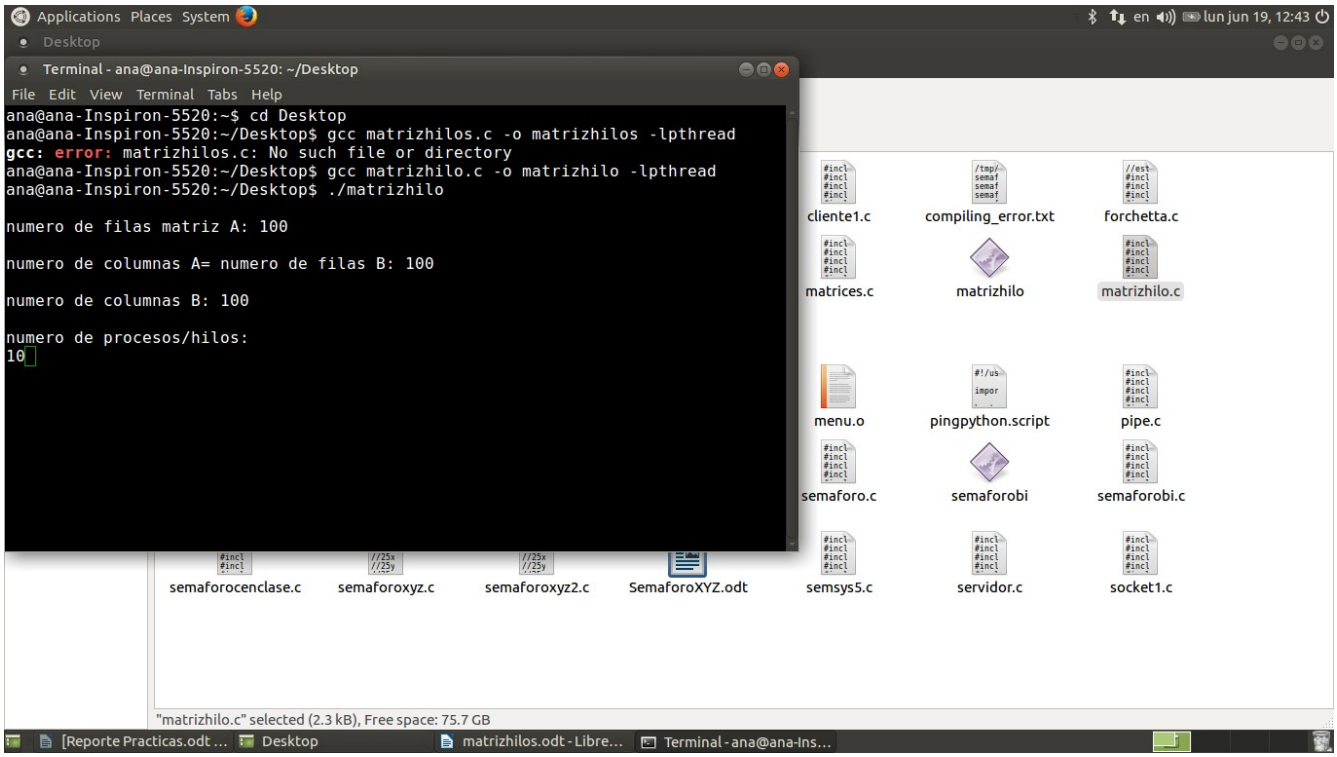
Dentro de un for colocamos el algoritmo que decidirá cuantas tareas realizará cada proceso, es aquí cuando utilizamos el arreglo de structs, así, cada struct será un proceso, y los valores de ini y fin serviran para saber de qué fila a que fila de la matrizA va a calcular por proceso.

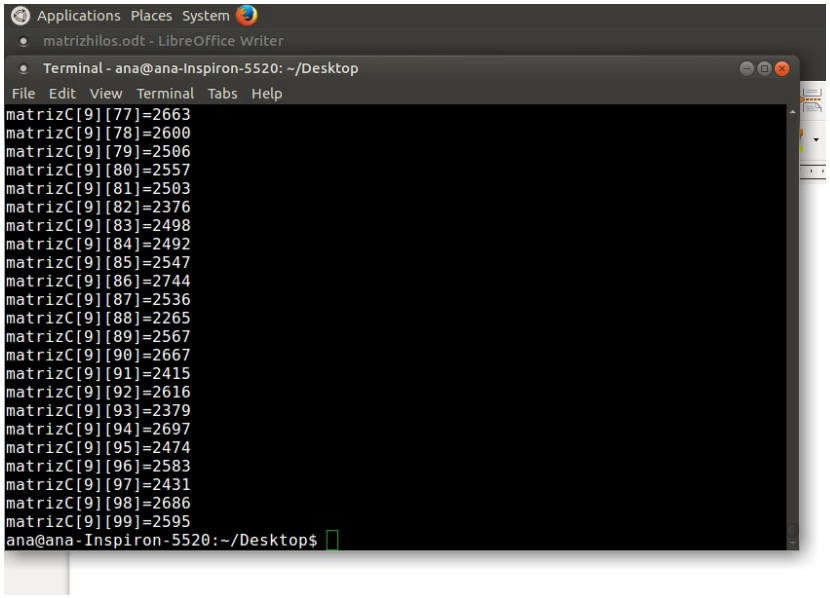
Dentro del mismo for, se coloca la función pthread_create, con la cual se le asigna a cada hilo uno de los structs, previamente casteados a void, que pasan por la función de multiplicación.

En la función de multiplicación, se vuelve a castear a struct el argumento que entró como vacío. Luego se sigue el algoritmo para multiplicar la matrizA por la matrizB y se imprimer de una vez el resultado.



Y por último la función pthread_join, viene siendo como un wait para hilos.


Applications Places System
en lun jun 19, 12:43
Desktop
Terminal - ana@ana-Inspiron-5520: ~/Desktop
File Edit View Terminal Tabs Help
ana@ana-Inspiron-5520:~$ cd Desktop
ana@ana-Inspiron-5520:~/Desktop$ gcc matrizhilos.c -o matrizhilos -lpthread
gcc: error: matrizhilos.c: No such file or directory
ana@ana-Inspiron-5520:~/Desktop$ gcc matrizhilo.c -o matrizhilo -lpthread
ana@ana-Inspiron-5520:~/Desktop$ ./matrizhilo
numero de filas matriz A: 100
numero de columnas A= numero de filas B: 100
numero de columnas B: 100
numero de procesos/hilos:
10
cliente1.c
compiling_error.txt
forchetta.c
matrices.c
matrizhilo
matrizhilo.c
menu.o
pingpython.script
pipe.c
semaforo.c
semaforobi
semaforobi.c
semaforocenclase.c
semaforoxyz.c
semaforoxyz2.c
SemaforoXYZ.odt
semsys5.c
servidor.c
socket1.c
"matrizhilo.c" selected (2.3 kB), Free space: 75.7 GB
[Reporte Practicas.odt ...
Desktop
matrizhilos.odt - Libre...
Terminal - ana@ana-Ins...


Applications Places System
matrizhilos.odt - LibreOffice Writer
Terminal - ana@ana-Inspiron-5520: ~/Desktop
File Edit View Terminal Tabs Help
matrizC[9][77]=2663
matrizC[9][78]=2600
matrizC[9][79]=2506
matrizC[9][80]=2557
matrizC[9][81]=2503
matrizC[9][82]=2376
matrizC[9][83]=2498
matrizC[9][84]=2492
matrizC[9][85]=2547
matrizC[9][86]=2744
matrizC[9][87]=2536
matrizC[9][88]=2265
matrizC[9][89]=2567
matrizC[9][90]=2667
matrizC[9][91]=2415
matrizC[9][92]=2616
matrizC[9][93]=2379
matrizC[9][94]=2697
matrizC[9][95]=2474
matrizC[9][96]=2583
matrizC[9][97]=2431
matrizC[9][98]=2686
matrizC[9][99]=2595
ana@ana-Inspiron-5520:~/Desktop$