# Linux

## 5.-

### Con procesos:

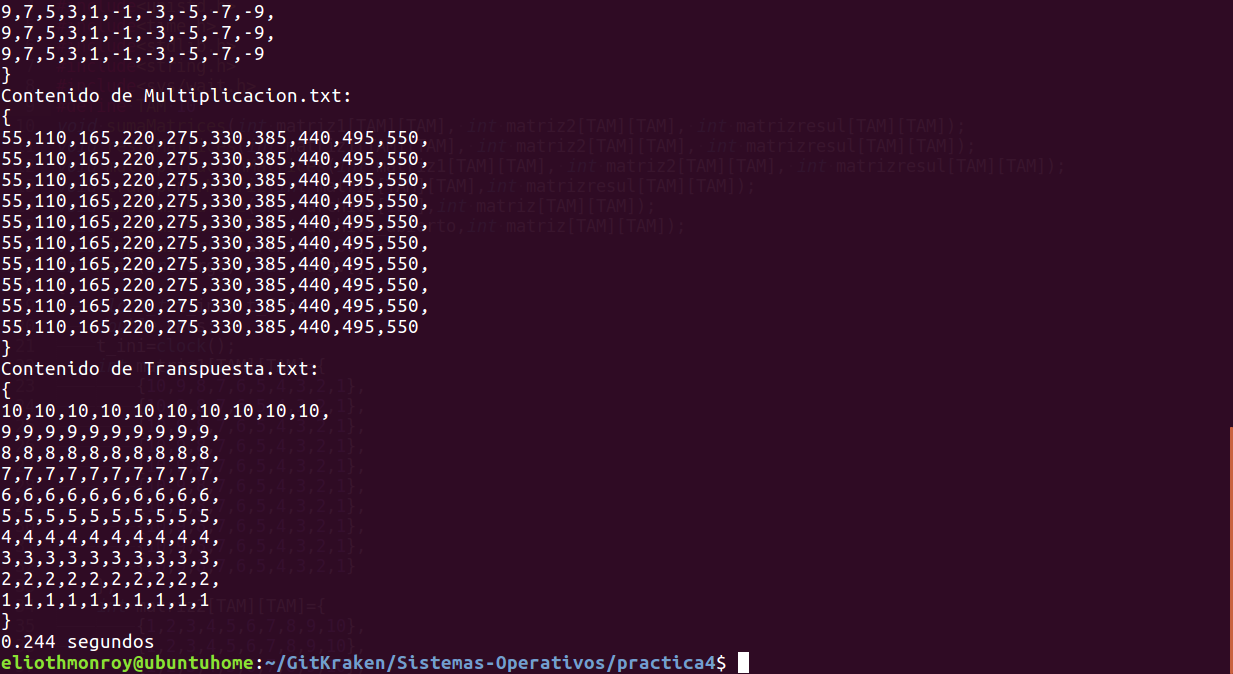
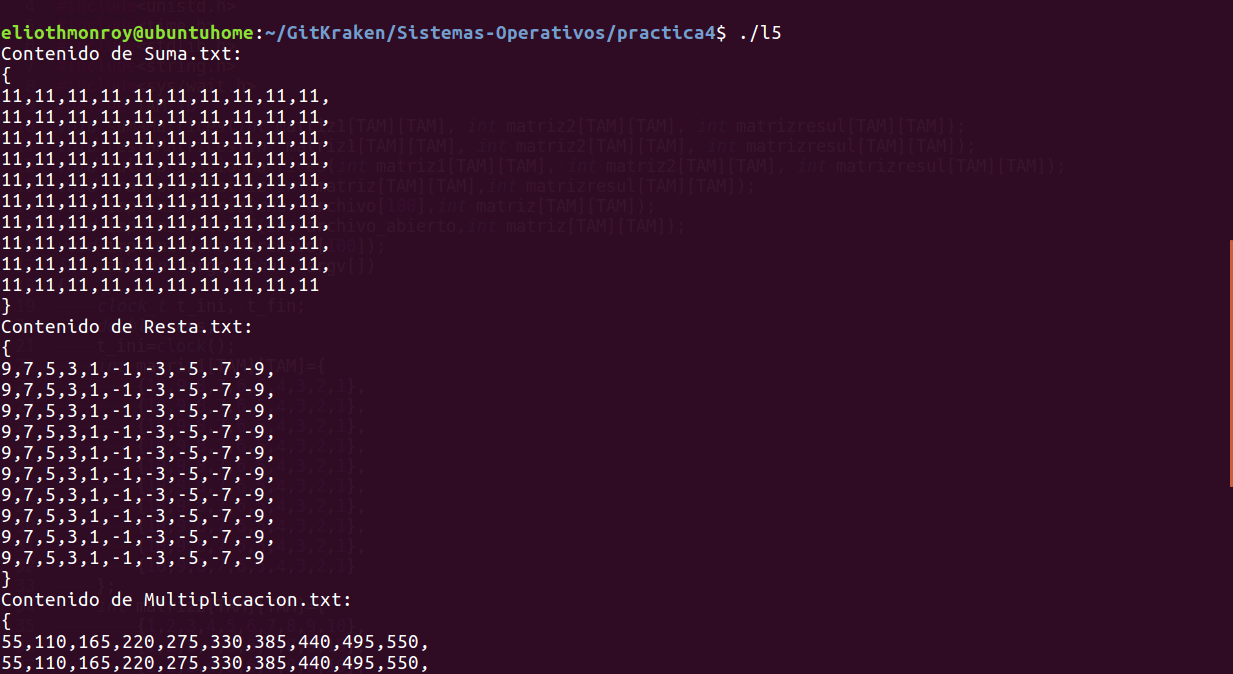
### l5.c

|  |
| --- |
| #include<stdio.h>  #include<sys/stat.h>  #include<fcntl.h>  #include<unistd.h>  #include<time.h>  #include<stdlib.h>  #include<string.h>  #include<sys/wait.h>  #define TAM 10  **void** sumaMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]);  **void** restaMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]);  **void** multiplicacionMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]);  **void** transponerMatriz(**int** matriz[TAM][TAM],**int** matrizresul[TAM][TAM]);  **void** imprimirMatriz(**char** archivo[100],**int** matriz[TAM][TAM]);  **void** imprimirMatrizT(**int** archivo\_abierto,**int** matriz[TAM][TAM]);  **void** imprimir(**char** archivo[100]);  **int** main(**int** argc, **char** \*argv[])  {  **clock\_t** t\_ini, t\_fin;  **double** secs;  t\_ini=clock();  **int** matriz1[TAM][TAM]={  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1}  };  **int** matriz2[TAM][TAM]={  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10}  };  **int** matrizresul[TAM][TAM];  **pid\_t** pid;  **if**((pid=fork())==0){  sumaMatrices(matriz1,matriz2,matrizresul);  exit(0);  }**else**{  **if**((pid=fork())==0){  restaMatrices(matriz1,matriz2,matrizresul);  exit(0);  }**else**{  **if**((pid=fork())==0){  multiplicacionMatrices(matriz1,matriz2,matrizresul);  exit(0);  }**else**{  **if**((pid=fork())==0){  transponerMatriz(matriz1,matrizresul);  **struct** stat estatus;  **int** archivo\_abierto=0;  **char** archivo[100]="Transpuesta.txt";  archivo\_abierto=open(archivo,O\_RDWR|O\_CREAT|O\_TRUNC|O\_APPEND);  imprimirMatrizT(archivo\_abierto,matrizresul);  transponerMatriz(matriz2,matrizresul);  imprimirMatrizT(archivo\_abierto,matrizresul);  close(archivo\_abierto);  exit(0);  }**else**{  **if**((pid=fork())==0){  imprimir("Suma.txt");  imprimir("Resta.txt");  imprimir("Multiplicacion.txt");  imprimir("Transpuesta.txt");  exit(0);  }  }  }  }  }  **for**(**int** i=0;i<5;i++){  wait(NULL);  }  t\_fin = clock();  secs = (**double**)(t\_fin - t\_ini) / CLOCKS\_PER\_SEC;  printf("%.16g segundos**\n**", secs \* 1000.0);  **return** 0;  }  **void** sumaMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]){  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  matrizresul[i][j]=matriz1[i][j]+matriz2[i][j];  }  }  imprimirMatriz("Suma.txt",matrizresul);  }  **void** restaMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]){  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  matrizresul[i][j]=matriz1[i][j]-matriz2[i][j];  }  }  imprimirMatriz("Resta.txt",matrizresul);  }  **void** multiplicacionMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]){  **int** i,j,k=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  matrizresul[i][j]=0;  **for**(k=0;k<10;k++){  matrizresul[i][j]+=matriz1[i][k]\*matriz2[k][j];  }  }  }  imprimirMatriz("Multiplicacion.txt",matrizresul);  }  **void** transponerMatriz(**int** matriz[TAM][TAM],**int** matrizresul[TAM][TAM]){  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  matrizresul[i][j]=matriz[j][i];  }  }  }  **void** imprimirMatriz(**char** archivo[100],**int** matriz[TAM][TAM]){  **struct** stat estatus;  **int** archivo\_abierto=0;  **char** contenido[2];  archivo\_abierto=open(archivo,O\_RDWR|O\_CREAT|O\_TRUNC|O\_APPEND);  write(archivo\_abierto, "{**\n**", strlen("{**\n**"));  *//puts("{");*  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  sprintf(contenido, "%i", matriz[i][j]);  write(archivo\_abierto, contenido, strlen(contenido));  *//printf("%i", matriz[i][j]);*  **if**(i==9 && j==9){  }**else**{  *//printf(",");*  write(archivo\_abierto, ",", strlen(","));  }  }  write(archivo\_abierto, "**\n**", strlen("**\n**"));  *//puts("\n");*  }  *//puts("}");*  write(archivo\_abierto, "}**\n**", strlen("}**\n**"));  close(archivo\_abierto);  }  **void** imprimirMatrizT(**int** archivo\_abierto,**int** matriz[TAM][TAM]){  *//printf("%i\n", archivo\_abierto);*  **char** contenido[2];  write(archivo\_abierto, "{**\n**", strlen("{**\n**"));  *//puts("{");*  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  sprintf(contenido, "%i", matriz[i][j]);  write(archivo\_abierto, contenido, strlen(contenido));  *//printf("%i", matriz[i][j]);*  **if**(i==9 && j==9){  }**else**{  *//printf(",");*  write(archivo\_abierto, ",", strlen(","));  }  }  write(archivo\_abierto, "**\n**", strlen("**\n**"));  *//puts("\n");*  }  *//puts("}");*  write(archivo\_abierto, "}**\n**", strlen("}**\n**"));  }  **void** imprimir(**char** archivo[100]){  **struct** stat estatus;  **int** archivo\_abierto=0;  archivo\_abierto=open(archivo,O\_RDONLY);  stat(archivo,&estatus);  **char** contenido[estatus.st\_size];  read(archivo\_abierto, contenido, estatus.st\_size);  **int** i=0;  printf("Contenido de %s:**\n**", archivo);  **while** ((\*(contenido + i) != '\0') && i+1 <=estatus.st\_size) {  **if**(\*(contenido +i) > 0){  printf("%c", \*(contenido+i));  **if**(\*(contenido+i)=='}'){  puts("");  **break**;  }  }  i++;  }  close(archivo\_abierto);  } |

### Compilación



### Ejecución y resultado

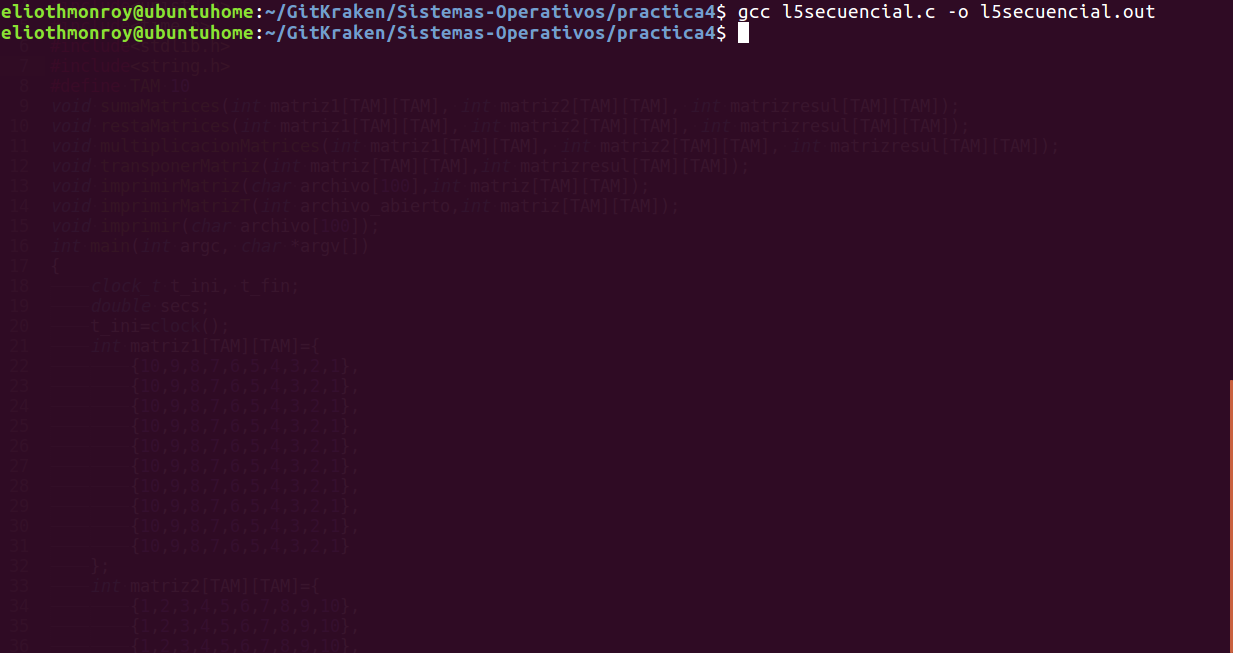


### Versión secuencial

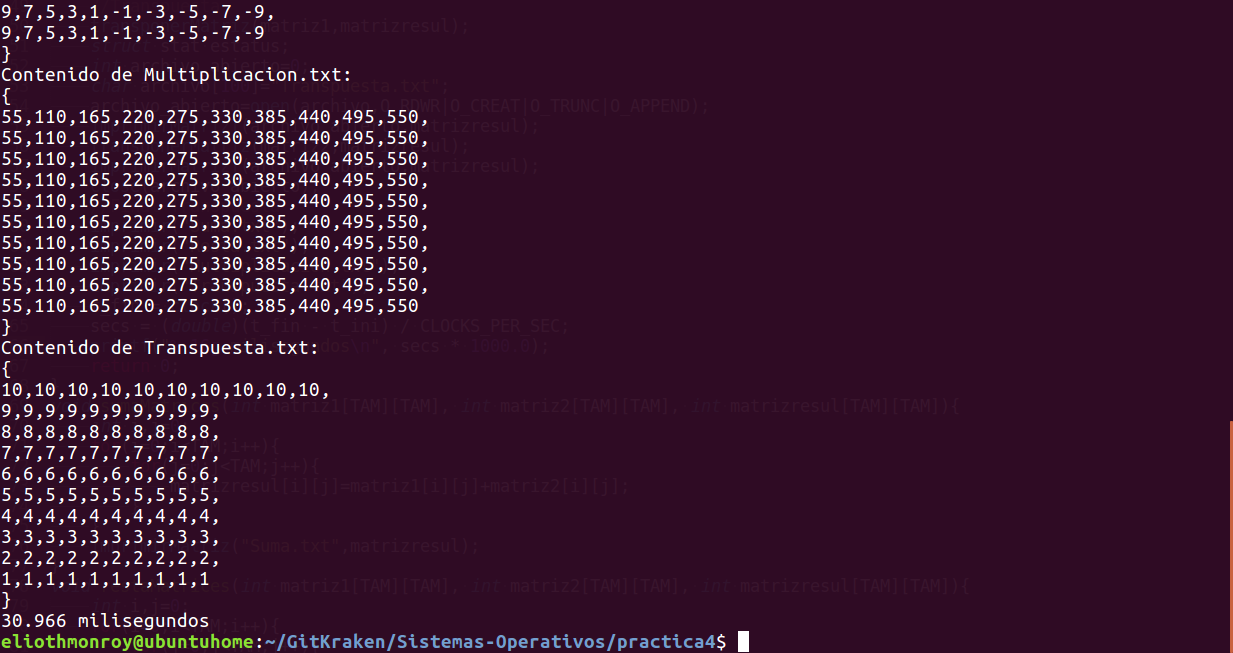
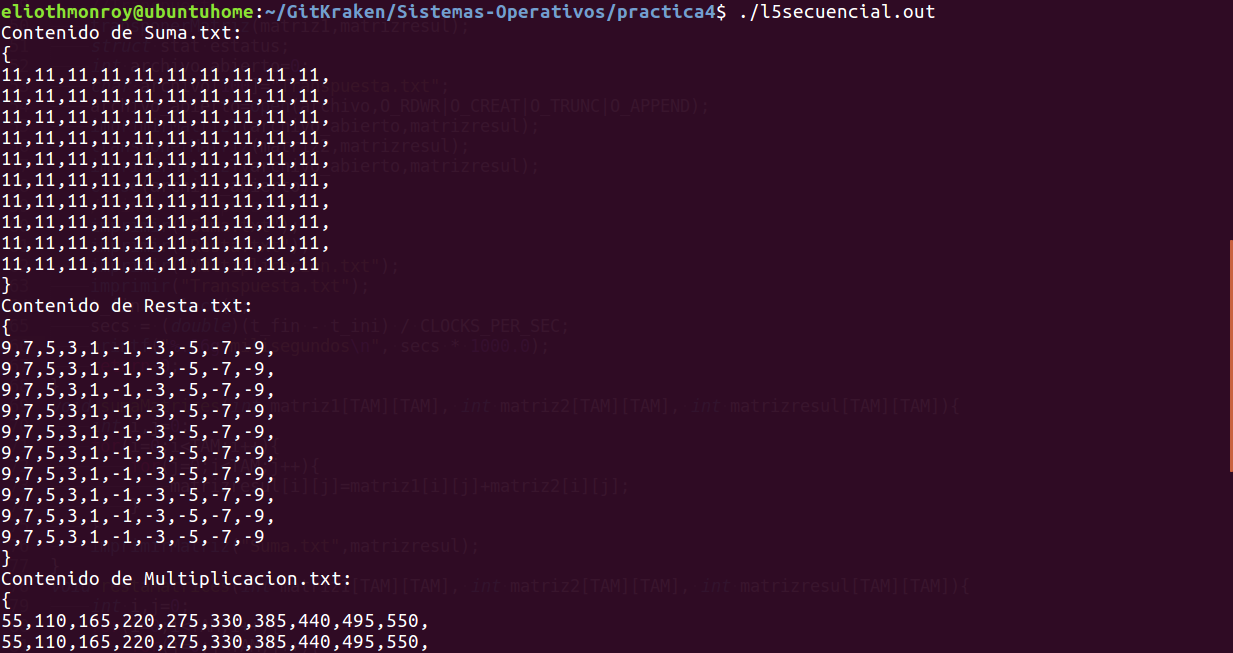
### l5secuencial.c

|  |
| --- |
| #include<stdio.h>  #include<sys/stat.h>  #include<fcntl.h>  #include<unistd.h>  #include<time.h>  #include<stdlib.h>  #include<string.h>  #define TAM 10  **void** sumaMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]);  **void** restaMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]);  **void** multiplicacionMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]);  **void** transponerMatriz(**int** matriz[TAM][TAM],**int** matrizresul[TAM][TAM]);  **void** imprimirMatriz(**char** archivo[100],**int** matriz[TAM][TAM]);  **void** imprimirMatrizT(**int** archivo\_abierto,**int** matriz[TAM][TAM]);  **void** imprimir(**char** archivo[100]);  **int** main(**int** argc, **char** \*argv[])  {  **clock\_t** t\_ini, t\_fin;  **double** secs;  t\_ini=clock();  **int** matriz1[TAM][TAM]={  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1}  };  **int** matriz2[TAM][TAM]={  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10}  };  **int** matrizresul[TAM][TAM];  sumaMatrices(matriz1,matriz2,matrizresul);  restaMatrices(matriz1,matriz2,matrizresul);  multiplicacionMatrices(matriz1,matriz2,matrizresul);  *//Transpuesta*  transponerMatriz(matriz1,matrizresul);  **struct** stat estatus;  **int** archivo\_abierto=0;  **char** archivo[100]="Transpuesta.txt";  archivo\_abierto=open(archivo,O\_RDWR|O\_CREAT|O\_TRUNC|O\_APPEND);  imprimirMatrizT(archivo\_abierto,matrizresul);  transponerMatriz(matriz2,matrizresul);  imprimirMatrizT(archivo\_abierto,matrizresul);  close(archivo\_abierto);  *//Final Transpuesta*  imprimir("Suma.txt");  imprimir("Resta.txt");  imprimir("Multiplicacion.txt");  imprimir("Transpuesta.txt");  t\_fin = clock();  secs = (**double**)(t\_fin - t\_ini) / CLOCKS\_PER\_SEC;  printf("%.16g milisegundos**\n**", secs \* 1000.0);  **return** 0;  }  **void** sumaMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]){  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  matrizresul[i][j]=matriz1[i][j]+matriz2[i][j];  }  }  imprimirMatriz("Suma.txt",matrizresul);  }  **void** restaMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]){  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  matrizresul[i][j]=matriz1[i][j]-matriz2[i][j];  }  }  imprimirMatriz("Resta.txt",matrizresul);  }  **void** multiplicacionMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]){  **int** i,j,k=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  matrizresul[i][j]=0;  **for**(k=0;k<10;k++){  matrizresul[i][j]+=matriz1[i][k]\*matriz2[k][j];  }  }  }  imprimirMatriz("Multiplicacion.txt",matrizresul);  }  **void** transponerMatriz(**int** matriz[TAM][TAM],**int** matrizresul[TAM][TAM]){  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  matrizresul[i][j]=matriz[j][i];  }  }  }  **void** imprimirMatriz(**char** archivo[100],**int** matriz[TAM][TAM]){  **struct** stat estatus;  **int** archivo\_abierto=0;  **char** contenido[2];  archivo\_abierto=open(archivo,O\_RDWR|O\_CREAT|O\_TRUNC|O\_APPEND);  write(archivo\_abierto, "{**\n**", strlen("{**\n**"));  *//puts("{");*  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  sprintf(contenido, "%i", matriz[i][j]);  write(archivo\_abierto, contenido, strlen(contenido));  *//printf("%i", matriz[i][j]);*  **if**(i==9 && j==9){  }**else**{  *//printf(",");*  write(archivo\_abierto, ",", strlen(","));  }  }  write(archivo\_abierto, "**\n**", strlen("**\n**"));  *//puts("\n");*  }  *//puts("}");*  write(archivo\_abierto, "}**\n**", strlen("}**\n**"));  close(archivo\_abierto);  }  **void** imprimirMatrizT(**int** archivo\_abierto,**int** matriz[TAM][TAM]){  *//printf("%i\n", archivo\_abierto);*  **char** contenido[2];  write(archivo\_abierto, "{**\n**", strlen("{**\n**"));  *//puts("{");*  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  sprintf(contenido, "%i", matriz[i][j]);  write(archivo\_abierto, contenido, strlen(contenido));  *//printf("%i", matriz[i][j]);*  **if**(i==9 && j==9){  }**else**{  *//printf(",");*  write(archivo\_abierto, ",", strlen(","));  }  }  write(archivo\_abierto, "**\n**", strlen("**\n**"));  *//puts("\n");*  }  *//puts("}");*  write(archivo\_abierto, "}**\n**", strlen("}**\n**"));  }  **void** imprimir(**char** archivo[100]){  **struct** stat estatus;  **int** archivo\_abierto=0;  archivo\_abierto=open(archivo,O\_RDONLY);  stat(archivo,&estatus);  **char** contenido[estatus.st\_size];  read(archivo\_abierto, contenido, estatus.st\_size);  **int** i=0;  printf("Contenido de %s:**\n**", archivo);  **while** ((\*(contenido + i) != '\0') && i+1 <=estatus.st\_size) {  **if**(\*(contenido +i) > 0){  printf("%c", \*(contenido+i));  **if**(\*(contenido+i)=='}'){  puts("");  **break**;  }  }  i++;  }  close(archivo\_abierto);  } |

### Compilación



### Ejecución y salida



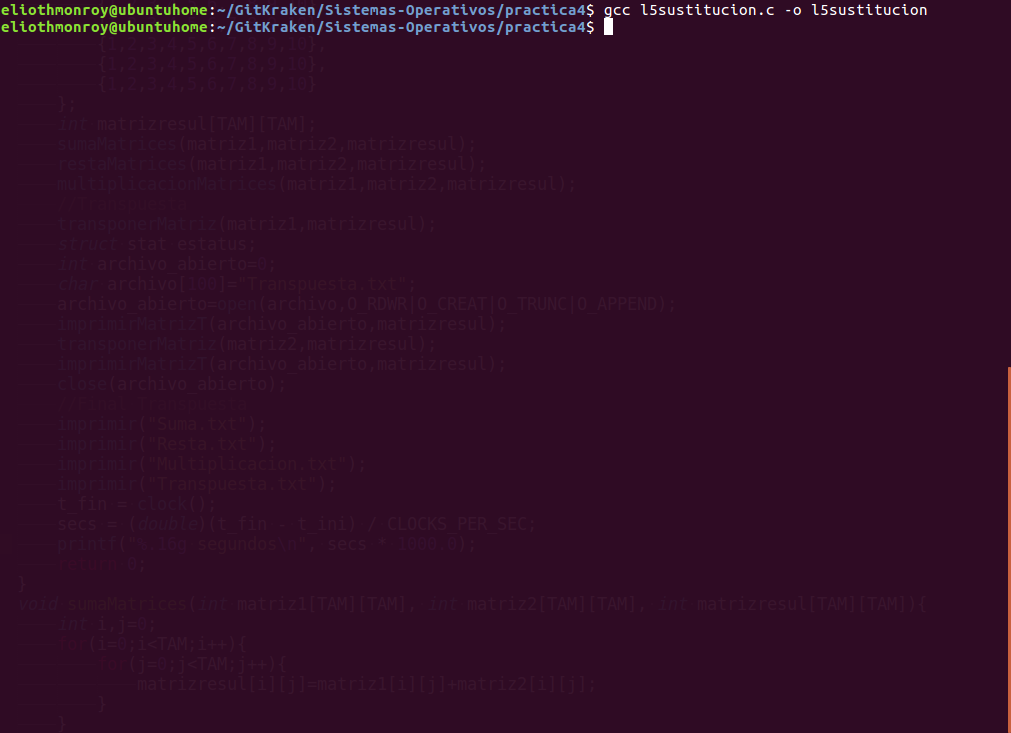
## 8.-

### Código proceso padre

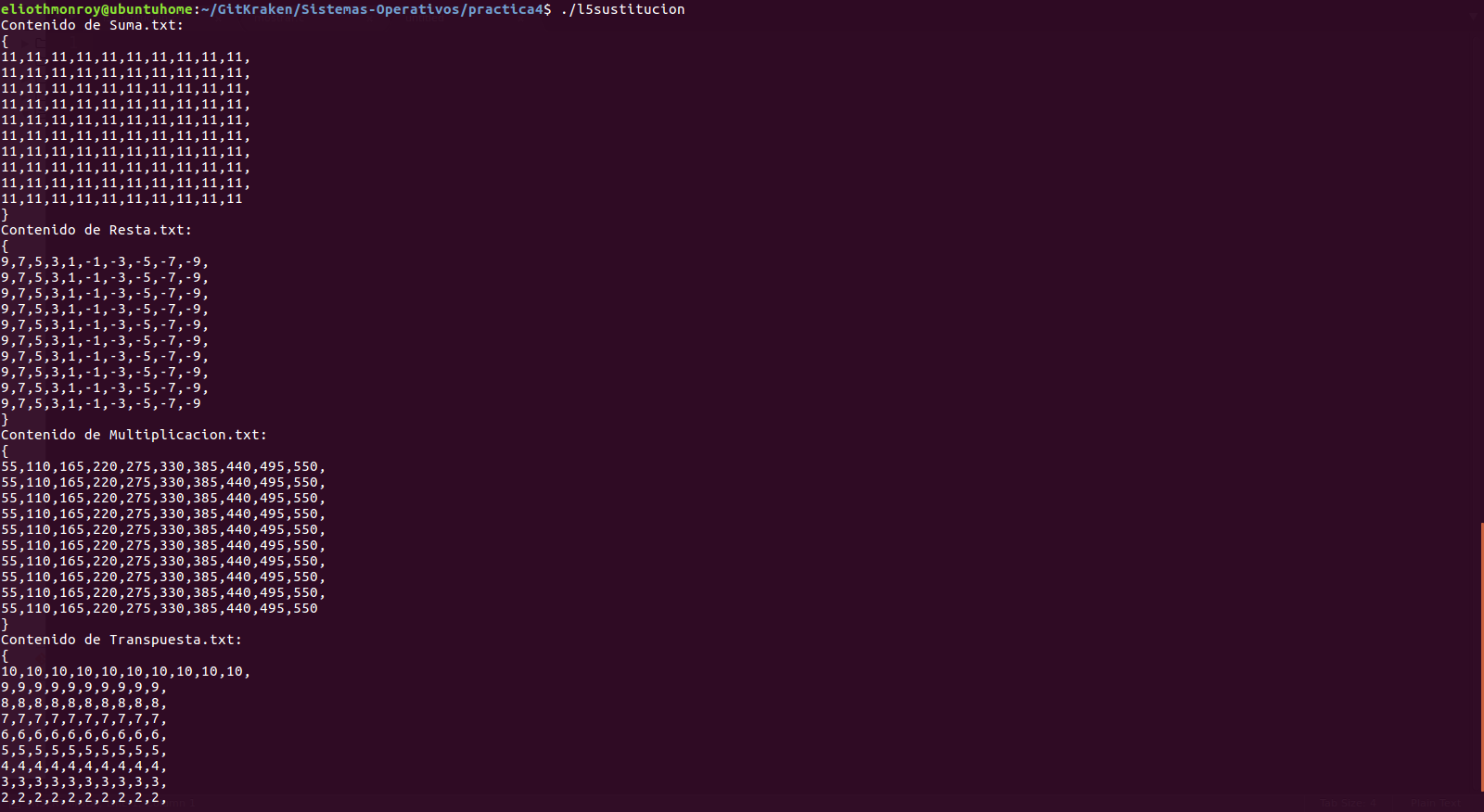
l5sustitucion.c

|  |
| --- |
| #include<stdio.h>  *//#include<windows.h>*  #include<sys/stat.h>  #include<fcntl.h>  #include<unistd.h>  #include<time.h>  #include<stdlib.h>  #include<sys/wait.h>  **int** main(**int** argc, **char** \*argv[])  {  **clock\_t** t\_ini, t\_fin;  **double** secs;  t\_ini=clock();  **pid\_t** pid;  **if**((pid=fork())==0){  execv("./suma",argv);  }**else**{  **if**((pid=fork())==0){  execv("./resta",argv);  }**else**{  **if**((pid=fork())==0){  execv("./multiplicacion",argv);  }**else**{  **if**((pid=fork())==0){  execv("./transpuesta",argv);  }**else**{  **if**((pid=fork())==0){  execv("./mostrar",argv);  }  }  }  }  }  **for**(**int** i=0;i<5;i++){  wait(NULL);  }  *//Parte final para obtener tiempos de ejecución*  t\_fin = clock();  secs = (**double**)(t\_fin - t\_ini) / CLOCKS\_PER\_SEC;  printf("%.16g milisegundos**\n**", secs \* 1000.0);  exit(0);  } |

### Compilación



### Ejecución y salida

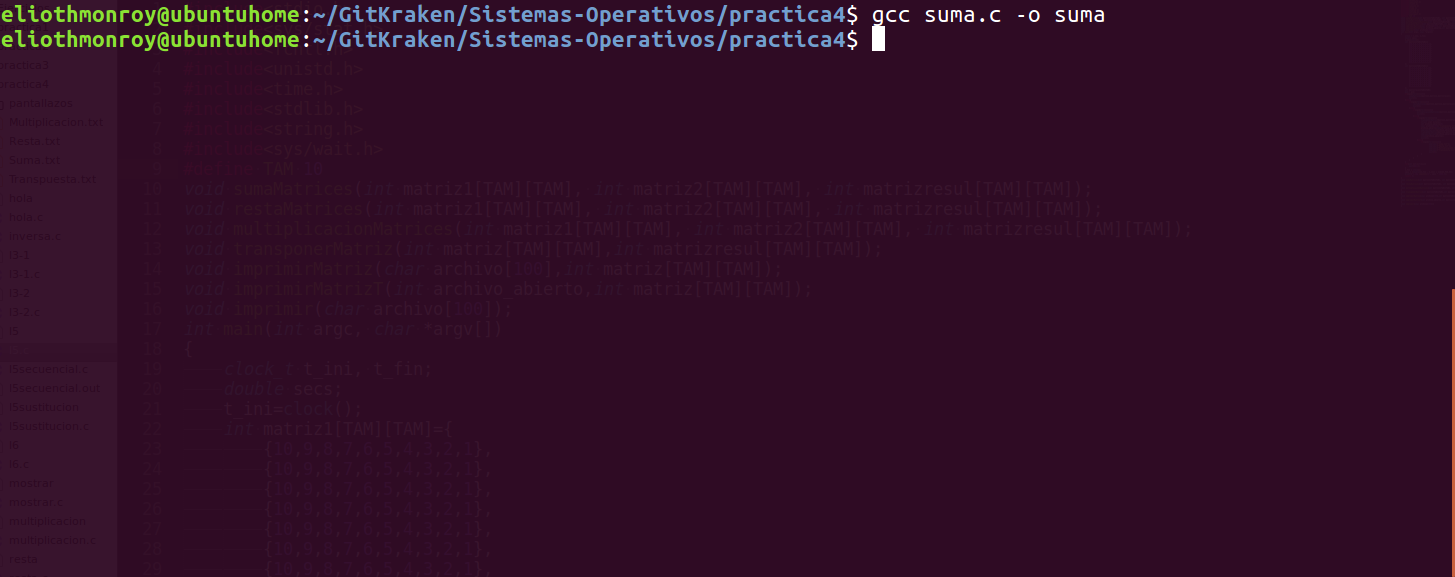


### Códigos procesos hijos

### suma.c

|  |
| --- |
| #include<stdio.h>  *//#include<windows.h>*  #include<stdlib.h>  #include<sys/stat.h>  #include<fcntl.h>  #include<string.h>  #include<unistd.h>  #define TAM 10  **void** sumaMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]);  **void** imprimirMatriz(**int** matriz[TAM][TAM]);  **int** main(**void**)  {  **int** matriz1[TAM][TAM]={  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1}  };  **int** matriz2[TAM][TAM]={  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10}  };  **int** matrizresul[TAM][TAM];  sumaMatrices(matriz1,matriz2,matrizresul);  imprimirMatriz(matrizresul);  exit(0);  }  **void** sumaMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]){  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  matrizresul[i][j]=matriz1[i][j]+matriz2[i][j];  }  }  }  **void** imprimirMatriz(**int** matriz[TAM][TAM]){  **struct** stat estatus;  **int** archivo\_abierto=0;  **char** archivo[100]="Suma.txt";  **char** contenido[2];  archivo\_abierto=open(archivo,O\_RDWR|O\_CREAT|O\_TRUNC|O\_APPEND);  write(archivo\_abierto, "{**\n**", strlen("{**\n**"));  *//puts("{");*  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  sprintf(contenido, "%i", matriz[i][j]);  write(archivo\_abierto, contenido, strlen(contenido));  *//printf("%i", matriz[i][j]);*  **if**(i==9 && j==9){  }**else**{  *//printf(",");*  write(archivo\_abierto, ",", strlen(","));  }  }  write(archivo\_abierto, "**\n**", strlen("**\n**"));  *//puts("\n");*  }  *//puts("}");*  write(archivo\_abierto, "}**\n**", strlen("}**\n**"));  close(archivo\_abierto);  } |

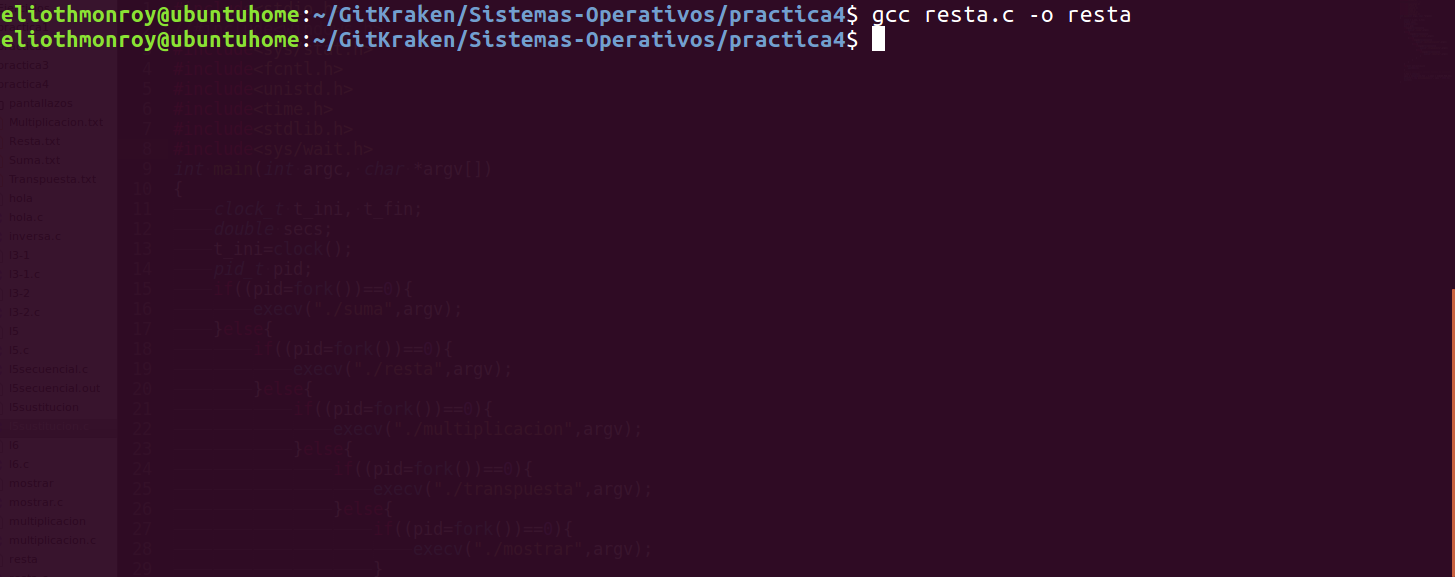
### Compilación



### resta.c

|  |
| --- |
| #include<stdio.h>  *//#include<windows.h>*  #include<stdlib.h>  #include<sys/stat.h>  #include<fcntl.h>  #include<string.h>  #include<unistd.h>  #define TAM 10  **void** restaMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]);  **void** imprimirMatriz(**int** matriz[TAM][TAM]);  **int** main(**void**)  {  **int** matriz1[TAM][TAM]={  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1}  };  **int** matriz2[TAM][TAM]={  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10}  };  **int** matrizresul[TAM][TAM];  restaMatrices(matriz1,matriz2,matrizresul);  imprimirMatriz(matrizresul);  exit(0);  }  **void** restaMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]){  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  matrizresul[i][j]=matriz1[i][j]-matriz2[i][j];  }  }  }  **void** imprimirMatriz(**int** matriz[TAM][TAM]){  **struct** stat estatus;  **int** archivo\_abierto=0;  **char** archivo[100]="Resta.txt";  **char** contenido[2];  archivo\_abierto=open(archivo,O\_RDWR|O\_CREAT|O\_TRUNC|O\_APPEND);  write(archivo\_abierto, "{**\n**", strlen("{**\n**"));  *//puts("{");*  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  sprintf(contenido, "%i", matriz[i][j]);  write(archivo\_abierto, contenido, strlen(contenido));  *//printf("%i", matriz[i][j]);*  **if**(i==9 && j==9){  }**else**{  *//printf(",");*  write(archivo\_abierto, ",", strlen(","));  }  }  write(archivo\_abierto, "**\n**", strlen("**\n**"));  *//puts("\n");*  }  *//puts("}");*  write(archivo\_abierto, "}**\n**", strlen("}**\n**"));  close(archivo\_abierto);  } |

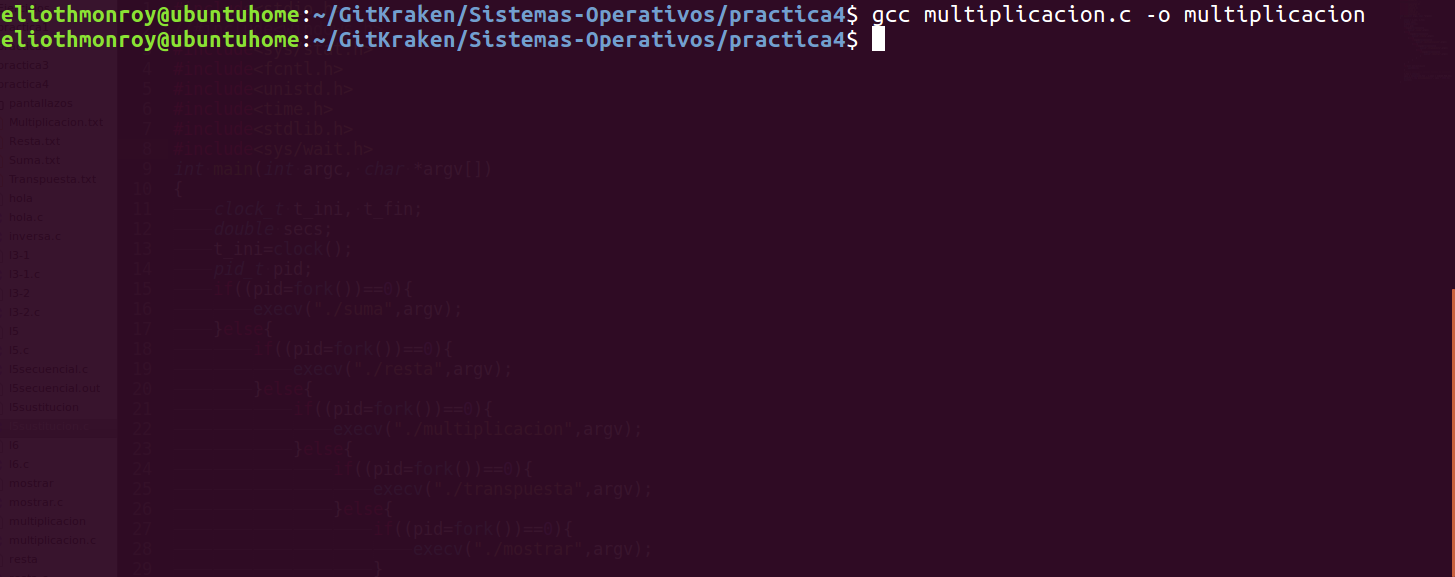
### Compilación



### multiplicación.c

|  |
| --- |
| #include<stdio.h>  *//#include<windows.h>*  #include<stdlib.h>  #include<sys/stat.h>  #include<fcntl.h>  #include<string.h>  #include<unistd.h>  #define TAM 10  **void** multiplicacionMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]);  **void** imprimirMatriz(**int** matriz[TAM][TAM]);  **int** main(**void**)  {  **int** matriz1[TAM][TAM]={  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1}  };  **int** matriz2[TAM][TAM]={  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10}  };  **int** matrizresul[TAM][TAM];  multiplicacionMatrices(matriz1,matriz2,matrizresul);  imprimirMatriz(matrizresul);  exit(0);  }  **void** multiplicacionMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]){  **int** i,j,k=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  matrizresul[i][j]=0;  **for**(k=0;k<10;k++){  matrizresul[i][j]+=matriz1[i][k]\*matriz2[k][j];  }  }  }  }  **void** imprimirMatriz(**int** matriz[TAM][TAM]){  **struct** stat estatus;  **int** archivo\_abierto=0;  **char** archivo[100]="Multiplicacion.txt";  **char** contenido[2];  archivo\_abierto=open(archivo,O\_RDWR|O\_CREAT|O\_TRUNC|O\_APPEND);  write(archivo\_abierto, "{**\n**", strlen("{**\n**"));  *//puts("{");*  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  sprintf(contenido, "%i", matriz[i][j]);  write(archivo\_abierto, contenido, strlen(contenido));  *//printf("%i", matriz[i][j]);*  **if**(i==9 && j==9){  }**else**{  *//printf(",");*  write(archivo\_abierto, ",", strlen(","));  }  }  write(archivo\_abierto, "**\n**", strlen("**\n**"));  *//puts("\n");*  }  *//puts("}");*  write(archivo\_abierto, "}**\n**", strlen("}**\n**"));  close(archivo\_abierto);  } |

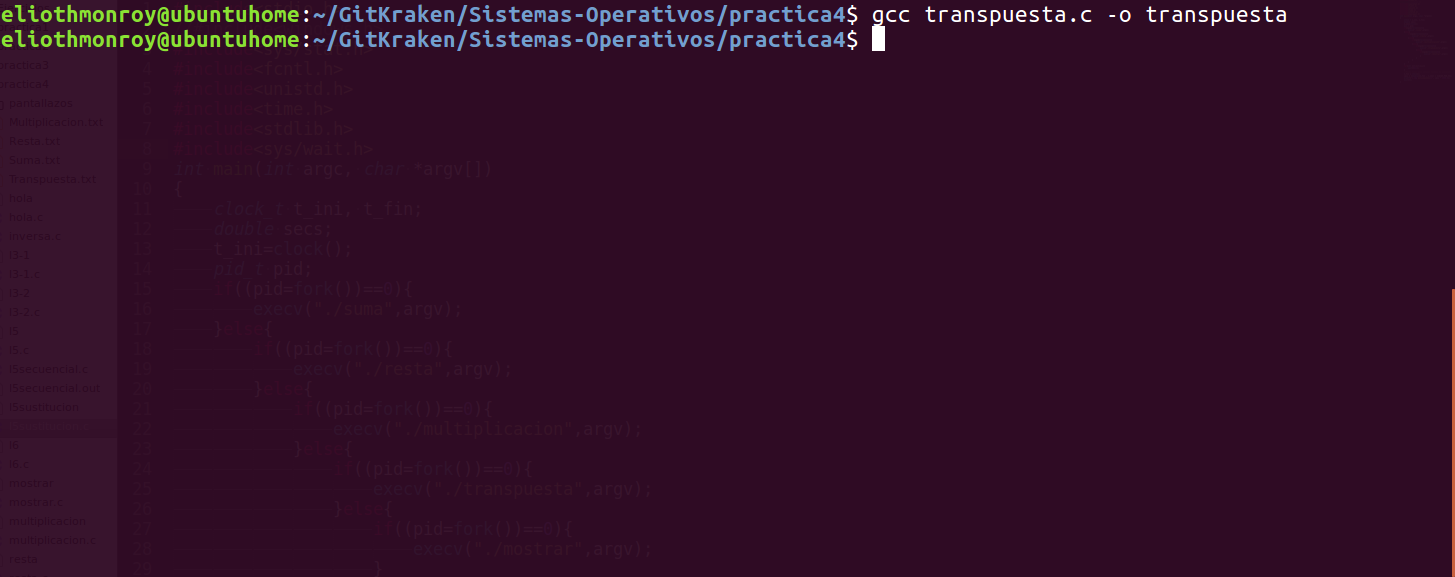
### Compilación



### transpuesta.c

|  |
| --- |
| #include<stdio.h>  *//#include<windows.h>*  #include<stdlib.h>  #include<sys/stat.h>  #include<fcntl.h>  #include<string.h>  #include<unistd.h>  #define TAM 10  **void** transponerMatriz(**int** matriz[TAM][TAM],**int** matrizresul[TAM][TAM]);  **void** imprimirMatriz(**int** archivo\_abierto,**int** matriz[TAM][TAM]);  *//info:http://www.ditutor.com/matrices/matriz\_traspuesta.html*  **int** main(**void**)  {  **int** matriz1[TAM][TAM]={  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1}  };  **int** matriz2[TAM][TAM]={  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10}  };  **int** matrizresul[TAM][TAM];  transponerMatriz(matriz1,matrizresul);  **struct** stat estatus;  **int** archivo\_abierto=0;  **char** archivo[100]="Transpuesta.txt";  archivo\_abierto=open(archivo,O\_RDWR|O\_CREAT|O\_TRUNC|O\_APPEND);  imprimirMatriz(archivo\_abierto,matrizresul);  transponerMatriz(matriz2,matrizresul);  imprimirMatriz(archivo\_abierto,matrizresul);  close(archivo\_abierto);  exit(0);  }  **void** transponerMatriz(**int** matriz[TAM][TAM],**int** matrizresul[TAM][TAM]){  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  matrizresul[i][j]=matriz[j][i];  }  }  }  **void** imprimirMatriz(**int** archivo\_abierto,**int** matriz[TAM][TAM]){  *//printf("%i\n", archivo\_abierto);*  **char** contenido[2];  write(archivo\_abierto, "{**\n**", strlen("{**\n**"));  *//puts("{");*  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  sprintf(contenido, "%i", matriz[i][j]);  write(archivo\_abierto, contenido, strlen(contenido));  *//printf("%i", matriz[i][j]);*  **if**(i==9 && j==9){  }**else**{  *//printf(",");*  write(archivo\_abierto, ",", strlen(","));  }  }  write(archivo\_abierto, "**\n**", strlen("**\n**"));  *//puts("\n");*  }  *//puts("}");*  write(archivo\_abierto, "}**\n**", strlen("}**\n**"));  } |

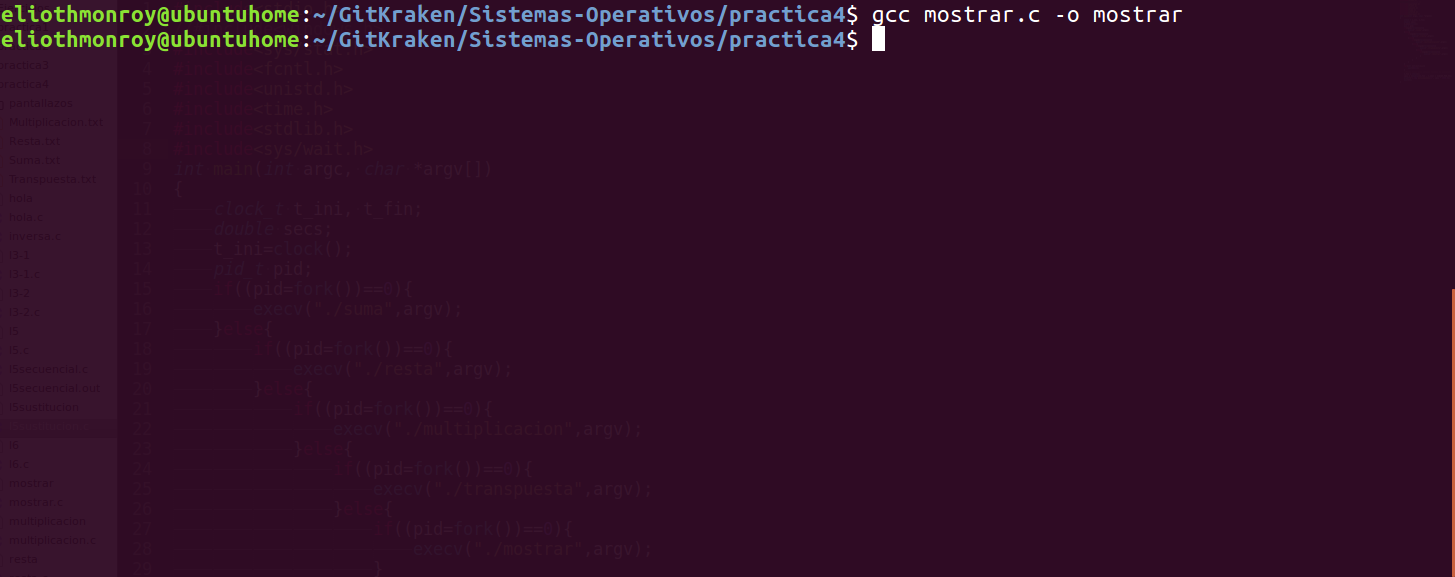
### Compilación



### mostrar.c

|  |
| --- |
| #include<sys/stat.h>  #include<stdio.h>  #include<string.h>  #include<fcntl.h>  #include<unistd.h>  #include<stdlib.h>  **void** imprimir(**char** archivo[100]);  **int** main(**int** argc, **char** **const** \*argv[])  {  *//Las estructuras a usar*  imprimir("Suma.txt");  imprimir("Resta.txt");  imprimir("Multiplicacion.txt");  imprimir("Transpuesta.txt");  **return** 0;  }  **void** imprimir(**char** archivo[100]){  **struct** stat estatus;  **int** archivo\_abierto=0;  archivo\_abierto=open(archivo,O\_RDONLY);  stat(archivo,&estatus);  **char** contenido[estatus.st\_size];  read(archivo\_abierto, contenido, estatus.st\_size);  **int** i=0;  printf("Contenido de %s:**\n**", archivo);  **while** ((\*(contenido + i) != '\0') && i+1 <=estatus.st\_size) {  **if**(\*(contenido +i) > 0){  printf("%c", \*(contenido+i));  **if**(\*(contenido+i)=='}'){  puts("");  **break**;  }  }  i++;  }  close(archivo\_abierto);  } |

### Compilación



# Windows

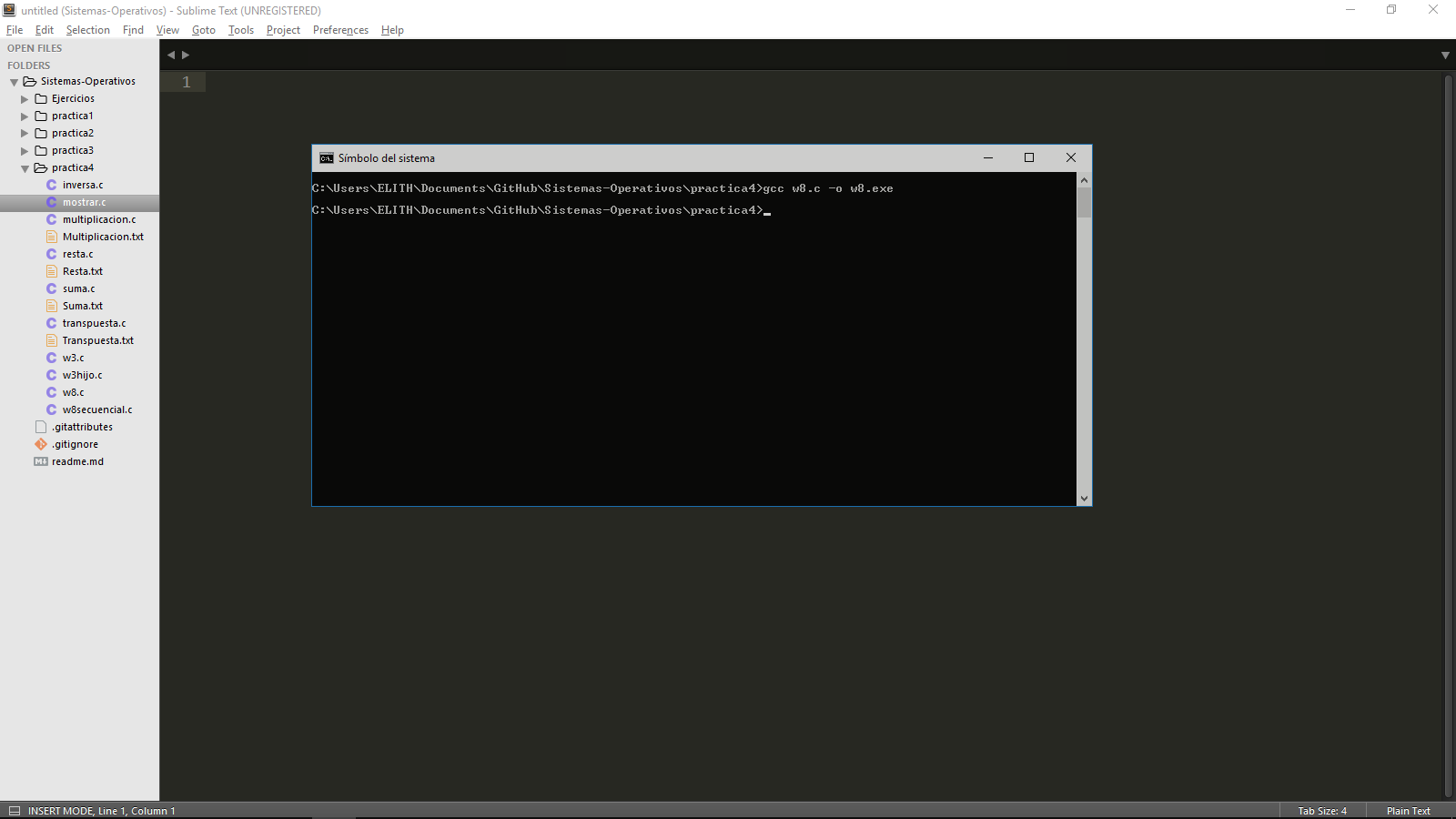
## 8.-

### Código programa padre usando procesos:

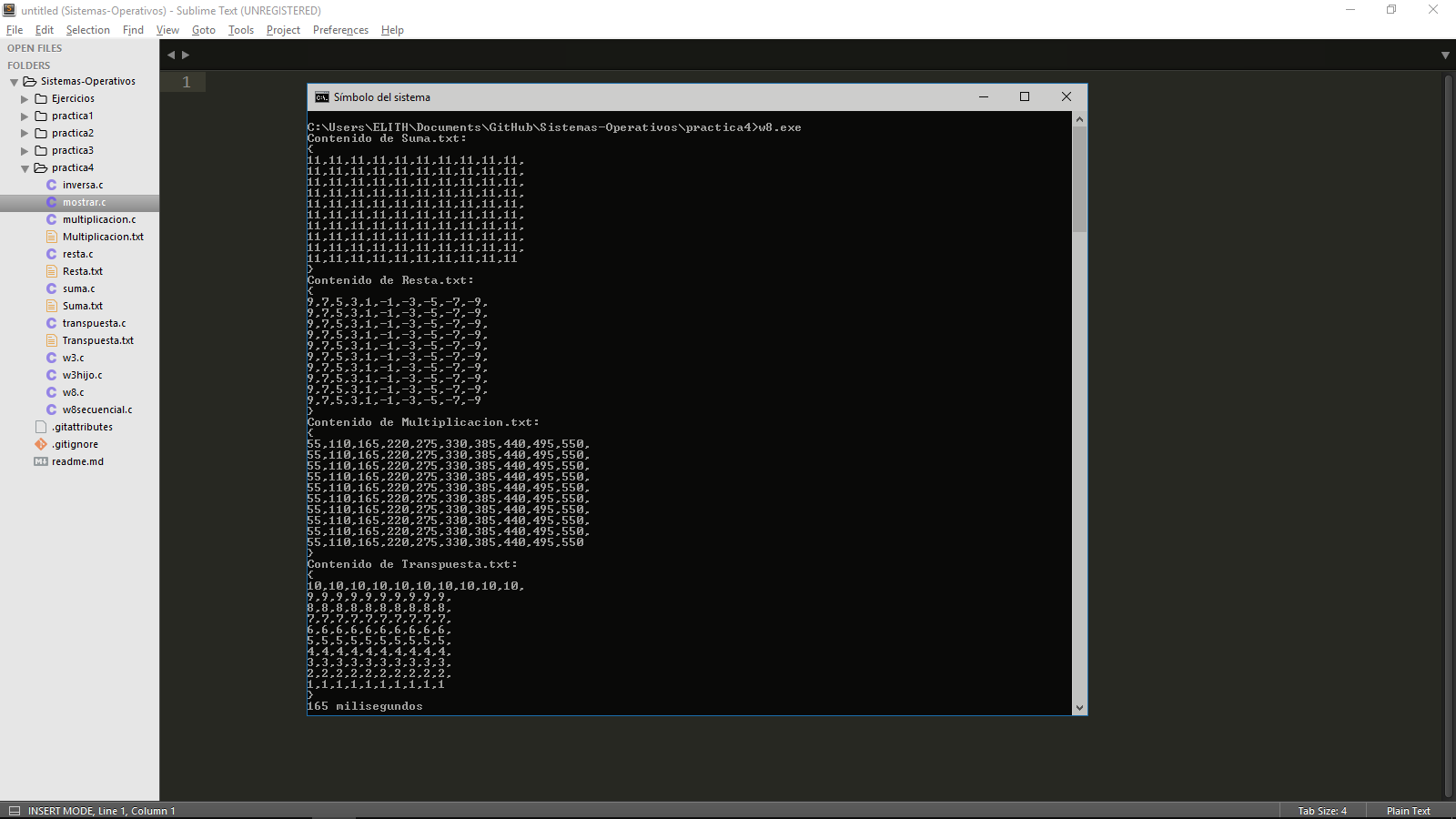
### w8.c

|  |
| --- |
| #include<stdio.h>  #include<windows.h>  #include<sys/stat.h>  #include<fcntl.h>  #include<unistd.h>  #include<time.h>  **int** main(**int** argc, **char** \*argv[])  {  **clock\_t** t\_ini, t\_fin;  **double** secs;  t\_ini=clock();  *//Estrucutra de información inicial para Windows*  STARTUPINFO si;  PROCESS\_INFORMATION pi;  **int** i;  ZeroMemory(&si, **sizeof**(si));  si.cb=**sizeof**(si);  ZeroMemory(&pi,**sizeof**(pi));  *//Creación del proceso hijo 1 suma*  **if**(!CreateProcess(NULL,"suma",NULL,NULL,FALSE,0,NULL,NULL,&si,&pi)){  printf("Fallo al invocar CreateProcess(%d)**\n**", GetLastError());  **return**;  }  *//Creación del proceso hijo 2 resta*  **if**(!CreateProcess(NULL,"resta",NULL,NULL,FALSE,0,NULL,NULL,&si,&pi)){  printf("Fallo al invocar CreateProcess(%d)**\n**", GetLastError());  **return**;  }  *//Creación del proceso hijo 3 multiplicacion*  **if**(!CreateProcess(NULL,"multi",NULL,NULL,FALSE,0,NULL,NULL,&si,&pi)){  printf("Fallo al invocar CreateProcess(%d)**\n**", GetLastError());  **return**;  }  *//Creación del proceso hijo 4 transpuesta*  **if**(!CreateProcess(NULL,"trans",NULL,NULL,FALSE,0,NULL,NULL,&si,&pi)){  printf("Fallo al invocar CreateProcess(%d)**\n**", GetLastError());  **return**;  }    *//Proceso Padre*  *//WaitForSingleObject sirve para ejecutar al hijo primero*  WaitForSingleObject(pi.hProcess,INFINITE);  *//Creación del proceso hijo 5 mostrar*  **if**(!CreateProcess(NULL,"mostrar",NULL,NULL,FALSE,0,NULL,NULL,&si,&pi)){  printf("Fallo al invocar CreateProcess(%d)**\n**", GetLastError());  **return**;  }  WaitForSingleObject(pi.hProcess,INFINITE);  *//Terminación controlada del proceso e hilo asociado a la ejecución*  CloseHandle(pi.hProcess);  CloseHandle(pi.hThread);  *//Parte final para obtener tiempos de ejecución*  t\_fin = clock();  secs = (**double**)(t\_fin - t\_ini) / CLOCKS\_PER\_SEC;  printf("%.16g milisegundos**\n**", secs \* 1000.0);  } |

### Compilación:



### Ejecución y resultados:

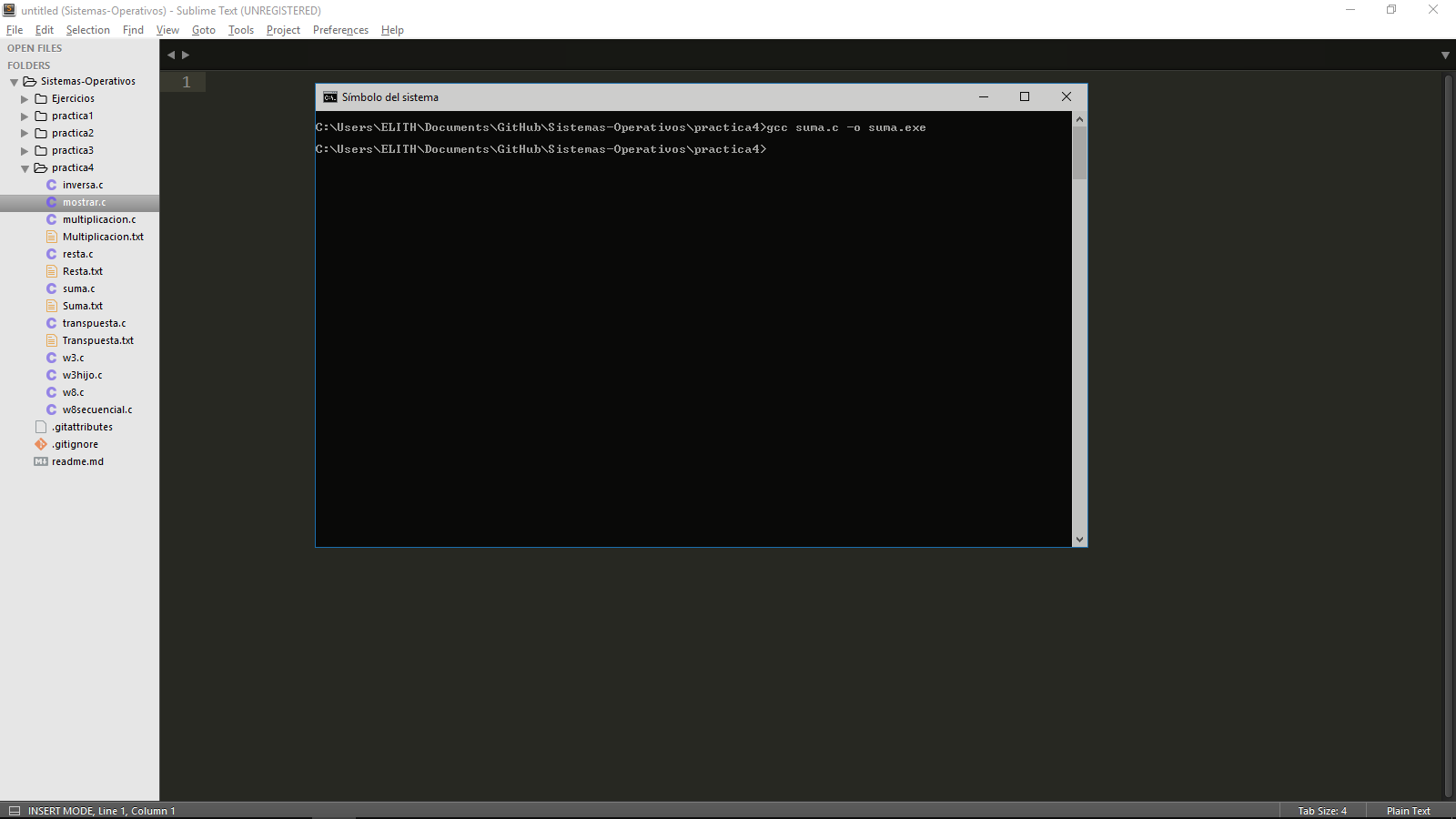


### Código de cada uno de los procesos hijos:

### suma.c

|  |
| --- |
| #include<stdio.h>  #include<windows.h>  #include<sys/stat.h>  #include<fcntl.h>  #include<string.h>  #include<unistd.h>  #define TAM 10  **void** sumaMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]);  **void** imprimirMatriz(**int** matriz[TAM][TAM]);  **int** main(**void**){  **int** matriz1[TAM][TAM]={  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1}  };  **int** matriz2[TAM][TAM]={  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10}  };  **int** matrizresul[TAM][TAM];  sumaMatrices(matriz1,matriz2,matrizresul);  imprimirMatriz(matrizresul);  exit(0);  }  **void** sumaMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]){  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  matrizresul[i][j]=matriz1[i][j]+matriz2[i][j];  }  }  }  **void** imprimirMatriz(**int** matriz[TAM][TAM]){  **struct** stat estatus;  **int** archivo\_abierto=0;  **char** archivo[100]="Suma.txt";  **char** contenido[2];  archivo\_abierto=open(archivo,O\_RDWR|O\_CREAT|O\_TRUNC|O\_APPEND);  write(archivo\_abierto, "{**\n**", strlen("{**\n**"));  *//puts("{");*  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  sprintf(contenido, "%i", matriz[i][j]);  write(archivo\_abierto, contenido, strlen(contenido));  *//printf("%i", matriz[i][j]);*  **if**(i==9 && j==9){  }**else**{  *//printf(",");*  write(archivo\_abierto, ",", strlen(","));  }  }  write(archivo\_abierto, "**\n**", strlen("**\n**"));  *//puts("\n");*  }  *//puts("}");*  write(archivo\_abierto, "}**\n**", strlen("}**\n**"));  close(archivo\_abierto);  } |

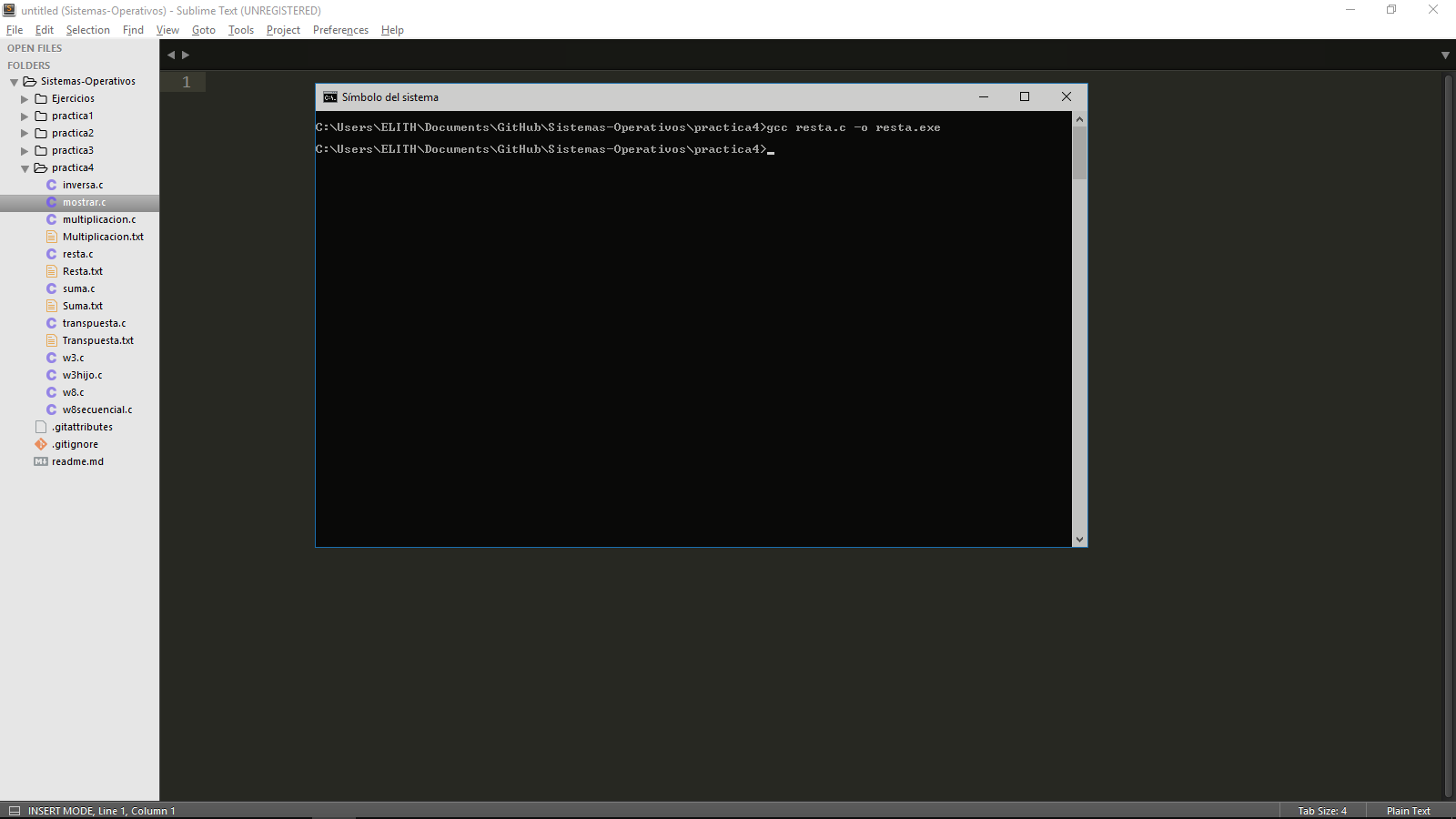
### Compilación:



### resta.c

|  |
| --- |
| #include<stdio.h>  #include<windows.h>  #include<sys/stat.h>  #include<fcntl.h>  #include<string.h>  #include<unistd.h>  #define TAM 10  **void** restaMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]);  **void** imprimirMatriz(**int** matriz[TAM][TAM]);  **int** main(**void**)  {  **int** matriz1[TAM][TAM]={  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1}  };  **int** matriz2[TAM][TAM]={  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10}  };  **int** matrizresul[TAM][TAM];  restaMatrices(matriz1,matriz2,matrizresul);  imprimirMatriz(matrizresul);  exit(0);  }  **void** restaMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]){  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  matrizresul[i][j]=matriz1[i][j]-matriz2[i][j];  }  }  }  **void** imprimirMatriz(**int** matriz[TAM][TAM]){  **struct** stat estatus;  **int** archivo\_abierto=0;  **char** archivo[100]="Resta.txt";  **char** contenido[2];  archivo\_abierto=open(archivo,O\_RDWR|O\_CREAT|O\_TRUNC|O\_APPEND);  write(archivo\_abierto, "{**\n**", strlen("{**\n**"));  *//puts("{");*  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  sprintf(contenido, "%i", matriz[i][j]);  write(archivo\_abierto, contenido, strlen(contenido));  *//printf("%i", matriz[i][j]);*  **if**(i==9 && j==9){  }**else**{  *//printf(",");*  write(archivo\_abierto, ",", strlen(","));  }  }  write(archivo\_abierto, "**\n**", strlen("**\n**"));  *//puts("\n");*  }  *//puts("}");*  write(archivo\_abierto, "}**\n**", strlen("}**\n**"));  close(archivo\_abierto);  } |

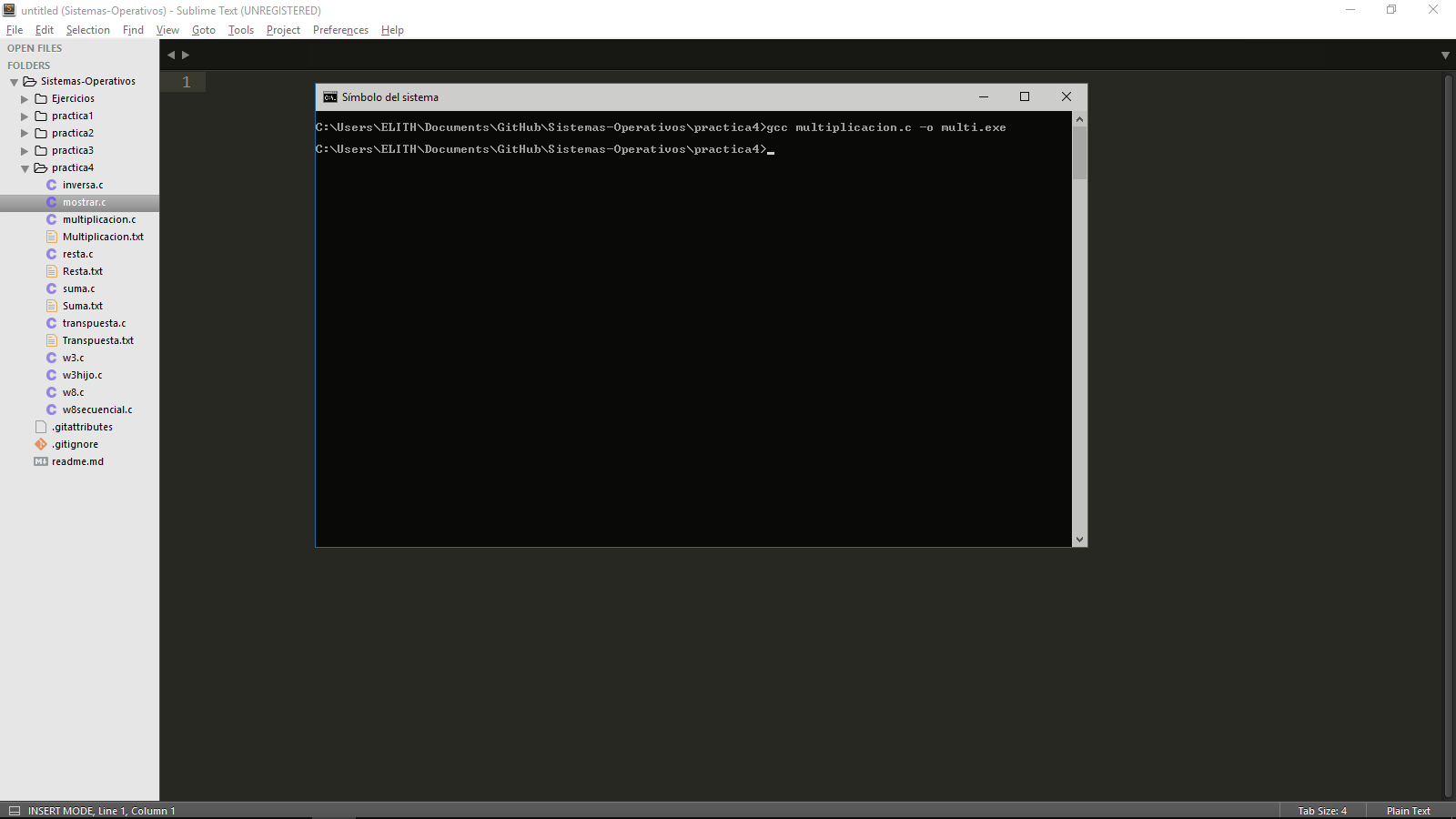
### Compilación



### multiplicación.c

|  |
| --- |
| #include<stdio.h>  #include<windows.h>  #include<sys/stat.h>  #include<fcntl.h>  #include<string.h>  #include<unistd.h>  #define TAM 10  **void** multiplicacionMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]);  **void** imprimirMatriz(**int** matriz[TAM][TAM]);  **int** main(**void**)  {  **int** matriz1[TAM][TAM]={  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1}  };  **int** matriz2[TAM][TAM]={  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10}  };  **int** matrizresul[TAM][TAM];  multiplicacionMatrices(matriz1,matriz2,matrizresul);  imprimirMatriz(matrizresul);  exit(0);  }  **void** multiplicacionMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]){  **int** i,j,k=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  matrizresul[i][j]=0;  **for**(k=0;k<10;k++){  matrizresul[i][j]+=matriz1[i][k]\*matriz2[k][j];  }  }  }  }  **void** imprimirMatriz(**int** matriz[TAM][TAM]){  **struct** stat estatus;  **int** archivo\_abierto=0;  **char** archivo[100]="Multiplicacion.txt";  **char** contenido[2];  archivo\_abierto=open(archivo,O\_RDWR|O\_CREAT|O\_TRUNC|O\_APPEND);  write(archivo\_abierto, "{**\n**", strlen("{**\n**"));  *//puts("{");*  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  sprintf(contenido, "%i", matriz[i][j]);  write(archivo\_abierto, contenido, strlen(contenido));  *//printf("%i", matriz[i][j]);*  **if**(i==9 && j==9){  }**else**{  *//printf(",");*  write(archivo\_abierto, ",", strlen(","));  }  }  write(archivo\_abierto, "**\n**", strlen("**\n**"));  *//puts("\n");*  }  *//puts("}");*  write(archivo\_abierto, "}**\n**", strlen("}**\n**"));  close(archivo\_abierto);  } |

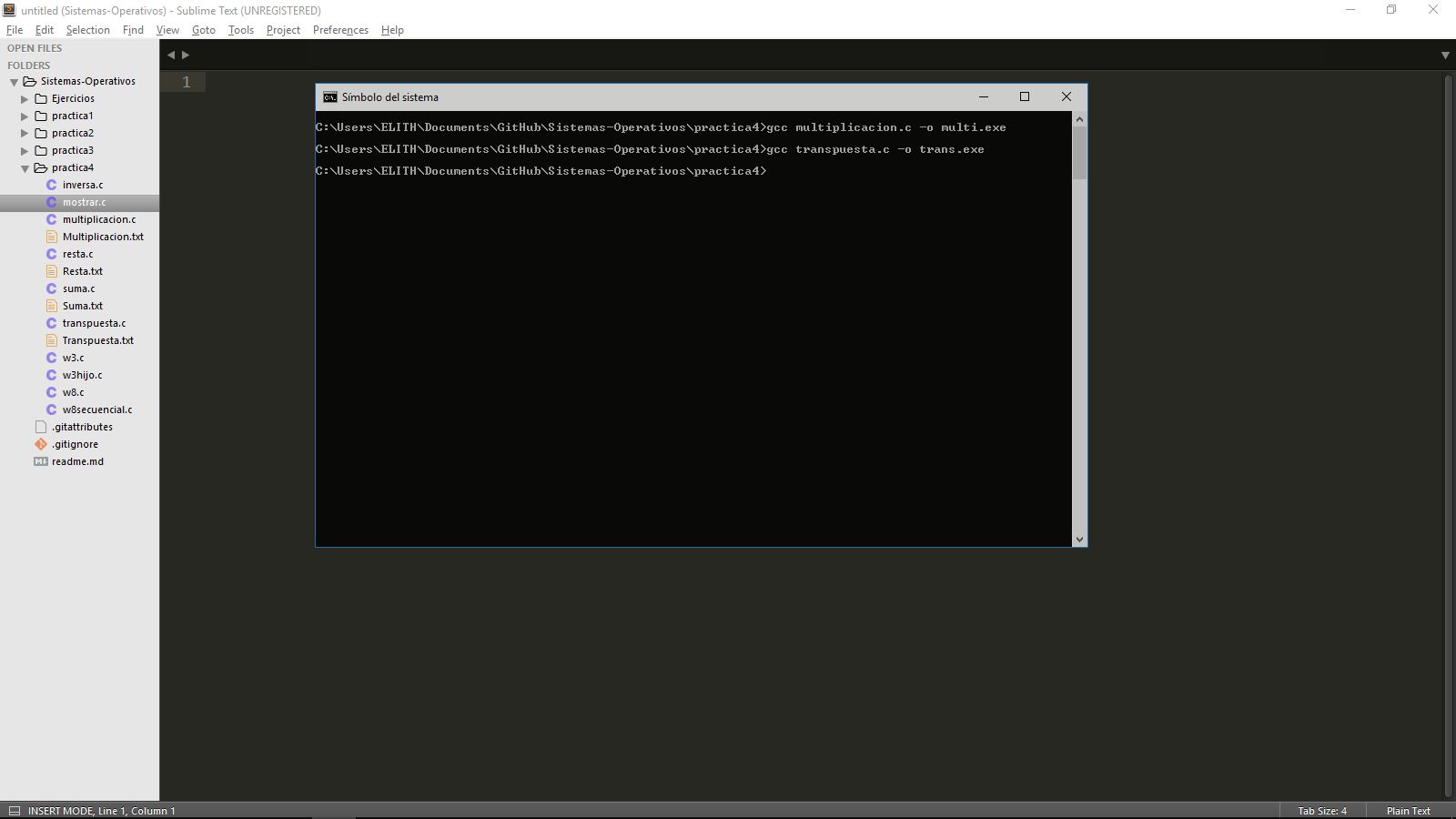
### Compilación:



### transpuesta.c

|  |
| --- |
| #include<stdio.h>  #include<windows.h>  #include<sys/stat.h>  #include<fcntl.h>  #include<string.h>  #include<unistd.h>  #define TAM 10  **void** transponerMatriz(**int** matriz[TAM][TAM],**int** matrizresul[TAM][TAM]);  **void** imprimirMatriz(**int** archivo\_abierto,**int** matriz[TAM][TAM]);  *//info:http://www.ditutor.com/matrices/matriz\_traspuesta.html*  **int** main(**void**)  {  **int** matriz1[TAM][TAM]={  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1}  };  **int** matriz2[TAM][TAM]={  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10}  };  **int** matrizresul[TAM][TAM];  transponerMatriz(matriz1,matrizresul);  **struct** stat estatus;  **int** archivo\_abierto=0;  **char** archivo[100]="Transpuesta.txt";  archivo\_abierto=open(archivo,O\_RDWR|O\_CREAT|O\_TRUNC|O\_APPEND);  imprimirMatriz(archivo\_abierto,matrizresul);  transponerMatriz(matriz2,matrizresul);  imprimirMatriz(archivo\_abierto,matrizresul);  close(archivo\_abierto);  exit(0);  }  **void** transponerMatriz(**int** matriz[TAM][TAM],**int** matrizresul[TAM][TAM]){  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  matrizresul[i][j]=matriz[j][i];  }  }  }  **void** imprimirMatriz(**int** archivo\_abierto,**int** matriz[TAM][TAM]){  *//printf("%i\n", archivo\_abierto);*  **char** contenido[2];  write(archivo\_abierto, "{**\n**", strlen("{**\n**"));  *//puts("{");*  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  sprintf(contenido, "%i", matriz[i][j]);  write(archivo\_abierto, contenido, strlen(contenido));  *//printf("%i", matriz[i][j]);*  **if**(i==9 && j==9){  }**else**{  *//printf(",");*  write(archivo\_abierto, ",", strlen(","));  }  }  write(archivo\_abierto, "**\n**", strlen("**\n**"));  *//puts("\n");*  }  *//puts("}");*  write(archivo\_abierto, "}**\n**", strlen("}**\n**"));  } |

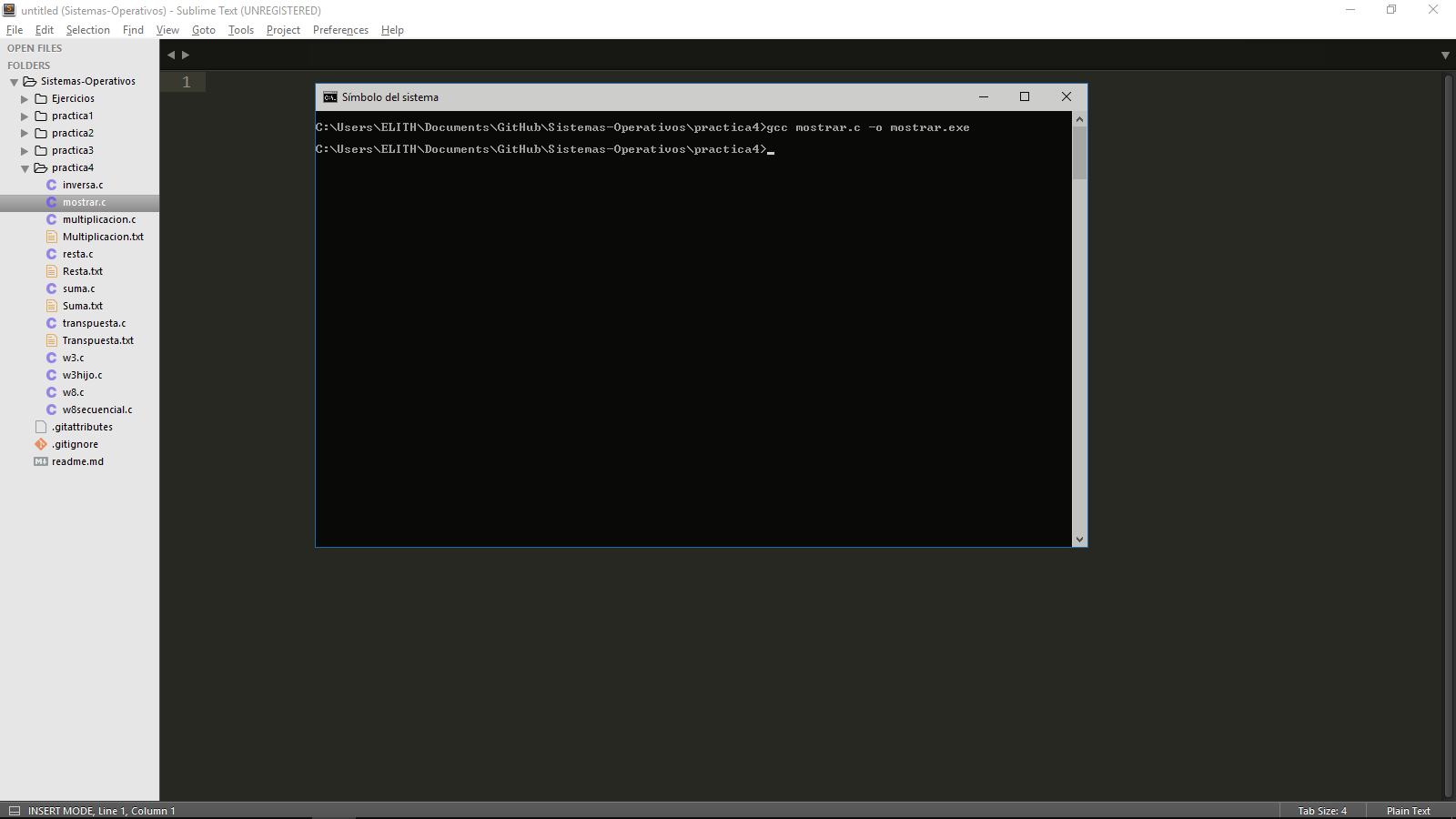
### Compilación:



mostrar.c

|  |
| --- |
| #include <sys/stat.h>  #include <stdio.h>  #include <string.h>  #include <fcntl.h>  #include <unistd.h>  **void** imprimir(**char** archivo[100]);  **int** main(**int** argc, **char** **const** \*argv[])  {  *//Las estructuras a usar*  imprimir("Suma.txt");  imprimir("Resta.txt");  imprimir("Multiplicacion.txt");  imprimir("Transpuesta.txt");  **return** 0;  }  **void** imprimir(**char** archivo[100]){  **struct** stat estatus;  **int** archivo\_abierto=0;  archivo\_abierto=open(archivo,O\_RDONLY);  stat(archivo,&estatus);  **char** contenido[estatus.st\_size];  read(archivo\_abierto, contenido, estatus.st\_size);  **int** i=0;  printf("Contenido de %s:**\n**", archivo);  **while** ((\*(contenido + i) != '\0') && i+1 <=estatus.st\_size) {  **if**(\*(contenido +i) > 0){  printf("%c", \*(contenido+i));  **if**(\*(contenido+i)=='}'){  puts("");  **break**;  }  }  i++; }  close(archivo\_abierto);} |

### Compilación:

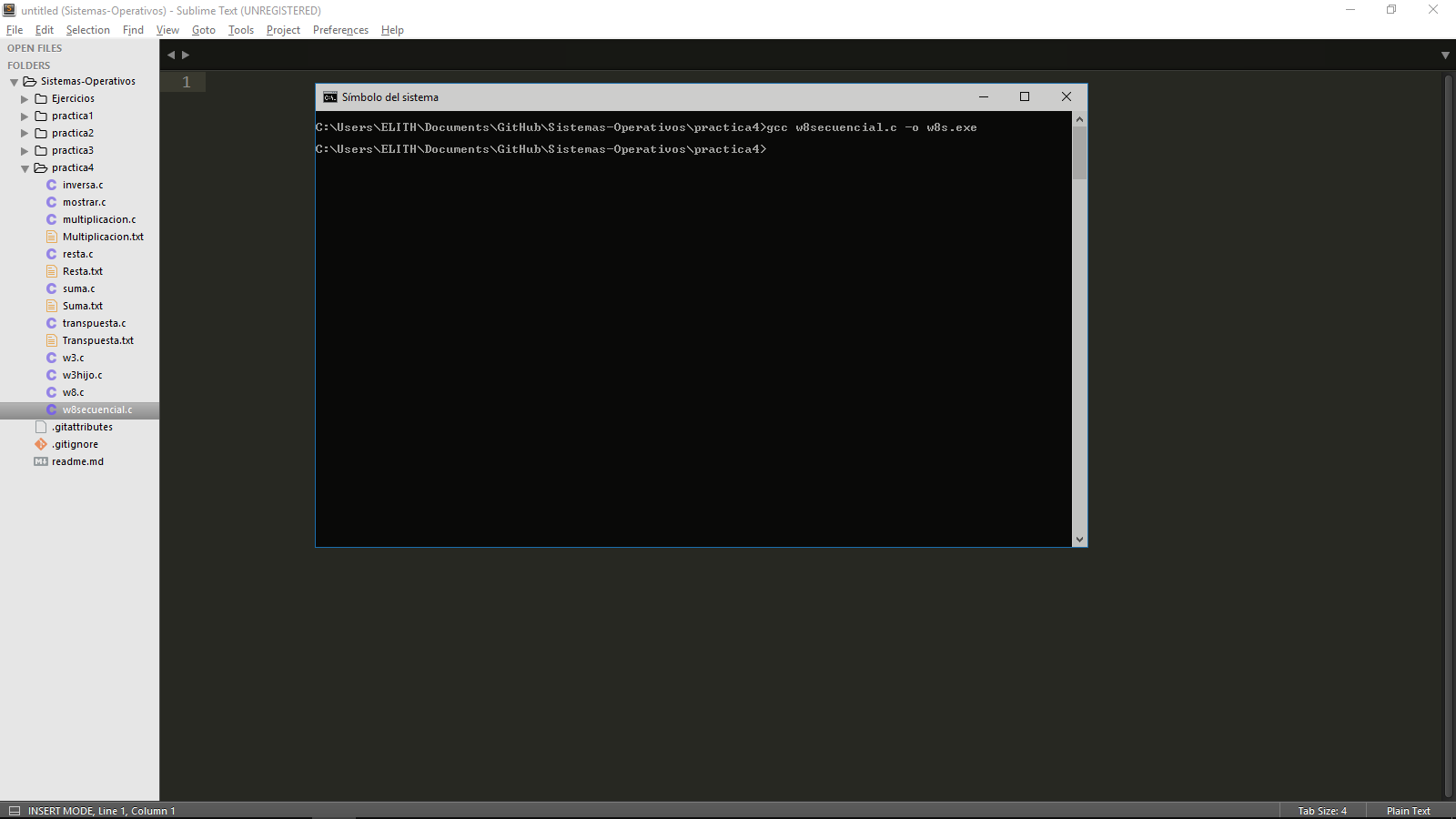


### Código programa padre versión secuencial:

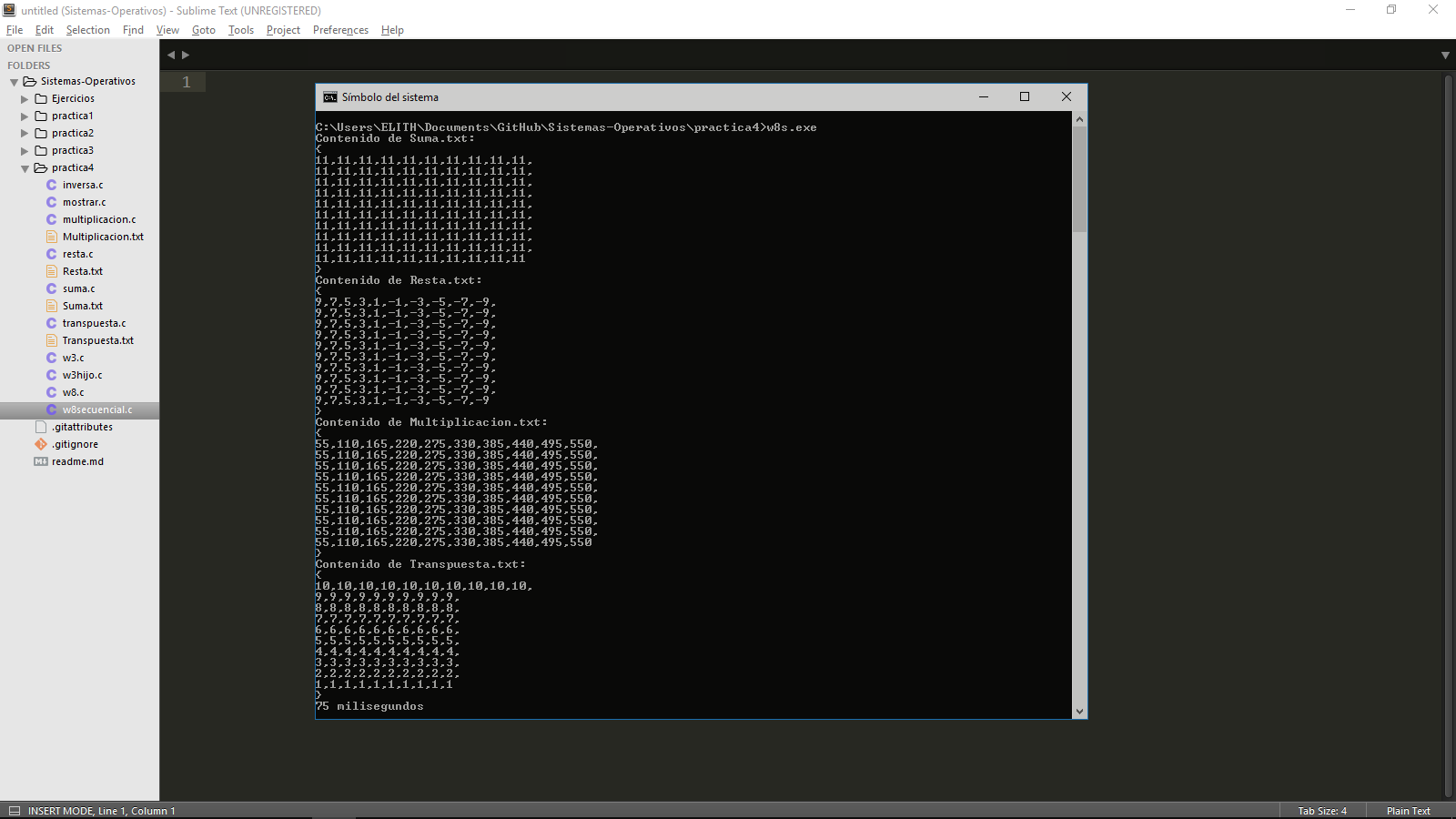
### w8secuencial.c

|  |
| --- |
| #include<stdio.h>  #include<windows.h>  #include<sys/stat.h>  #include<fcntl.h>  #include<unistd.h>  #include<time.h>  #define TAM 10  **void** sumaMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]);  **void** restaMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]);  **void** multiplicacionMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]);  **void** transponerMatriz(**int** matriz[TAM][TAM],**int** matrizresul[TAM][TAM]);  **void** imprimirMatriz(**char** archivo[100],**int** matriz[TAM][TAM]);  **void** imprimirMatrizT(**int** archivo\_abierto,**int** matriz[TAM][TAM]);  **void** imprimir(**char** archivo[100]);  **int** main(**int** argc, **char** \*argv[])  {  **clock\_t** t\_ini, t\_fin;  **double** secs;  t\_ini=clock();  **int** matriz1[TAM][TAM]={  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1},  {10,9,8,7,6,5,4,3,2,1}  };  **int** matriz2[TAM][TAM]={  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10},  {1,2,3,4,5,6,7,8,9,10}  };  **int** matrizresul[TAM][TAM];  sumaMatrices(matriz1,matriz2,matrizresul);  restaMatrices(matriz1,matriz2,matrizresul);  multiplicacionMatrices(matriz1,matriz2,matrizresul);  *//Transpuesta*  transponerMatriz(matriz1,matrizresul);  **struct** stat estatus;  **int** archivo\_abierto=0;  **char** archivo[100]="Transpuesta.txt";  archivo\_abierto=open(archivo,O\_RDWR|O\_CREAT|O\_TRUNC|O\_APPEND);  imprimirMatrizT(archivo\_abierto,matrizresul);  transponerMatriz(matriz2,matrizresul);  imprimirMatrizT(archivo\_abierto,matrizresul);  close(archivo\_abierto);  *//Final Transpuesta*  imprimir("Suma.txt");  imprimir("Resta.txt");  imprimir("Multiplicacion.txt");  imprimir("Transpuesta.txt");  t\_fin = clock();  secs = (**double**)(t\_fin - t\_ini) / CLOCKS\_PER\_SEC;  printf("%.16g milisegundos**\n**", secs \* 1000.0);  **return** 0;  }  **void** sumaMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]){  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  matrizresul[i][j]=matriz1[i][j]+matriz2[i][j];  }  }  imprimirMatriz("Suma.txt",matrizresul);  }  **void** restaMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]){  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  matrizresul[i][j]=matriz1[i][j]-matriz2[i][j];  }  }  imprimirMatriz("Resta.txt",matrizresul);  }  **void** multiplicacionMatrices(**int** matriz1[TAM][TAM], **int** matriz2[TAM][TAM], **int** matrizresul[TAM][TAM]){  **int** i,j,k=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  matrizresul[i][j]=0;  **for**(k=0;k<10;k++){  matrizresul[i][j]+=matriz1[i][k]\*matriz2[k][j];  }  }  }  imprimirMatriz("Multiplicacion.txt",matrizresul);  }  **void** transponerMatriz(**int** matriz[TAM][TAM],**int** matrizresul[TAM][TAM]){  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  matrizresul[i][j]=matriz[j][i];  }  }  }  **void** imprimirMatriz(**char** archivo[100],**int** matriz[TAM][TAM]){  **struct** stat estatus;  **int** archivo\_abierto=0;  **char** contenido[2];  archivo\_abierto=open(archivo,O\_RDWR|O\_CREAT|O\_TRUNC|O\_APPEND);  write(archivo\_abierto, "{**\n**", strlen("{**\n**"));  *//puts("{");*  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  sprintf(contenido, "%i", matriz[i][j]);  write(archivo\_abierto, contenido, strlen(contenido));  *//printf("%i", matriz[i][j]);*  **if**(i==9 && j==9){  }**else**{  *//printf(",");*  write(archivo\_abierto, ",", strlen(","));  }  }  write(archivo\_abierto, "**\n**", strlen("**\n**"));  *//puts("\n");*  }  *//puts("}");*  write(archivo\_abierto, "}**\n**", strlen("}**\n**"));  close(archivo\_abierto);  }  **void** imprimirMatrizT(**int** archivo\_abierto,**int** matriz[TAM][TAM]){  *//printf("%i\n", archivo\_abierto);*  **char** contenido[2];  write(archivo\_abierto, "{**\n**", strlen("{**\n**"));  *//puts("{");*  **int** i,j=0;  **for**(i=0;i<TAM;i++){  **for**(j=0;j<TAM;j++){  sprintf(contenido, "%i", matriz[i][j]);  write(archivo\_abierto, contenido, strlen(contenido));  *//printf("%i", matriz[i][j]);*  **if**(i==9 && j==9){  }**else**{  *//printf(",");*  write(archivo\_abierto, ",", strlen(","));  }  }  write(archivo\_abierto, "**\n**", strlen("**\n**"));  *//puts("\n");*  }  *//puts("}");*  write(archivo\_abierto, "}**\n**", strlen("}**\n**"));  }  **void** imprimir(**char** archivo[100]){  **struct** stat estatus;  **int** archivo\_abierto=0;  archivo\_abierto=open(archivo,O\_RDONLY);  stat(archivo,&estatus);  **char** contenido[estatus.st\_size];  read(archivo\_abierto, contenido, estatus.st\_size);  **int** i=0;  printf("Contenido de %s:**\n**", archivo);  **while** ((\*(contenido + i) != '\0') && i+1 <=estatus.st\_size) {  **if**(\*(contenido +i) > 0){  printf("%c", \*(contenido+i));  **if**(\*(contenido+i)=='}'){  puts("");  **break**;  }  }  i++;  }  close(archivo\_abierto);  } |

### Compilación:



### Ejecución y resultado:



### Comparación entre tiempos de ejecución

El tiempo de ejecución del programa padre usando procesos fue de: 165 milisegundos.

El tiempo de ejecución del programa padre versión secuencial fue de: 75 milisegundos.

Podemos a primera vista que el programa secuencial fue más rápido que el programa que usa procesos, ya que el programa con procesos se demoró más del doble que el programa secuencial.

Pero no siempre es así, depende de los procesos que se estén ejecutando en el procesador lo que hace que estos tiempos de ejecución cambien. En algunas ocasiones el programa secuencial tardaba más que el programa que usaba procesos, pero la mayoría de las veces el programa secuencial fue más rápido que el programa que usaba procesos.

Creo que esto es debido a que el programa secuencial es ejecutado como un solo proceso, el cual no aporta mucha carga al procesador debido a que es una aplicación pequeña, mientras que el programa que usa procesos, el padre crea varios procesos lo cuales el procesador debe ejecutar, y el padre no termina hasta que todos los procesos hayan terminado, pero en realidad no sabemos qué tanta prioridad le dé el procesador a todos estos procesos hijos, por lo cual creo esa es la razón por la cual el programa que usa procesos resulta ser un poco más lento que el secuencial.

### Observaciones (o eso como se llame en el reporte)

La creación de procesos en Windows solo nos permite crear procesos mediante sustitución de código, lo cual hace que debamos de tener un archivo ejecutable para cada uno de los procesos que queramos crear. Situación que en Linux es distinta, ya que este Sistema Operativos permite la creación de procesos tanto por sustitución de código como por copia exacta de código.

La programación fue un tanto complicada, pero en el aspecto relacionado a los programas, ya que en realidad trabajar con procesos no es muy complicado, particularmente en Linux el proceso es muy sencillo, es Windows se requiere de un poco más de trabajo para hacer funcionar a los procesos.