

COMPILADORES

GRUPO 1

TAREA PRÁCTICA 1

ESTUDIANTE: MIGUEL ANGEL SALAS DIAZ

PROFESOR: ADRIAN ULISES MERCADO MARTÍNEZ

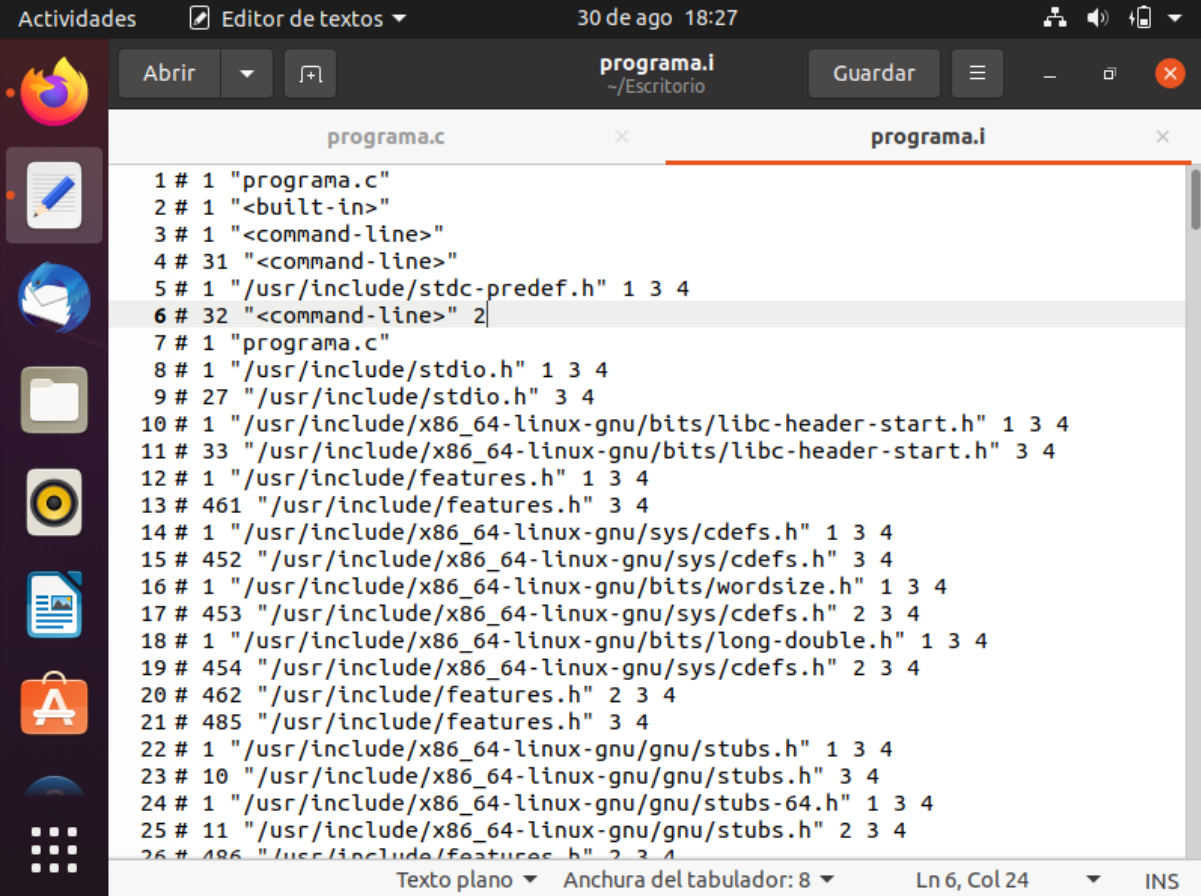
SEMESTRE 2023-1

31/08/2022

PREGUNTAS.

Usar el siguiente comando: `cpp programa.c > programa.i`

(a) Busque los archivos `.h` y revise su contenido



```
1 # 1 "programa.c"
2 # 1 "<built-in>"
3 # 1 "<command-line>"
4 # 31 "<command-line>"
5 # 1 "/usr/include/stdc-predef.h" 1 3 4
6 # 32 "<command-line>" 2
7 # 1 "programa.c"
8 # 1 "/usr/include/stdio.h" 1 3 4
9 # 27 "/usr/include/stdio.h" 3 4
10 # 1 "/usr/include/x86_64-linux-gnu/bits/libc-header-start.h" 1 3 4
11 # 33 "/usr/include/x86_64-linux-gnu/bits/libc-header-start.h" 3 4
12 # 1 "/usr/include/features.h" 1 3 4
13 # 461 "/usr/include/features.h" 3 4
14 # 1 "/usr/include/x86_64-linux-gnu/sys/cdefs.h" 1 3 4
15 # 452 "/usr/include/x86_64-linux-gnu/sys/cdefs.h" 3 4
16 # 1 "/usr/include/x86_64-linux-gnu/bits/wordsize.h" 1 3 4
17 # 453 "/usr/include/x86_64-linux-gnu/sys/cdefs.h" 2 3 4
18 # 1 "/usr/include/x86_64-linux-gnu/bits/long-double.h" 1 3 4
19 # 454 "/usr/include/x86_64-linux-gnu/sys/cdefs.h" 2 3 4
20 # 462 "/usr/include/features.h" 2 3 4
21 # 485 "/usr/include/features.h" 3 4
22 # 1 "/usr/include/x86_64-linux-gnu/gnu/stubs.h" 1 3 4
23 # 10 "/usr/include/x86_64-linux-gnu/gnu/stubs.h" 3 4
24 # 1 "/usr/include/x86_64-linux-gnu/gnu/stubs-64.h" 1 3 4
25 # 11 "/usr/include/x86_64-linux-gnu/gnu/stubs.h" 2 3 4
26 # 486 "/usr/include/features.h" 2 3 4
```

(b) Compare el contenido de `programa.i` con el de `stdio.h` y `stdlib.h`, indique de forma general las similitudes entre los archivos `.h` y el `.i`

Parece que el `programa.i` contiene unas partes de `stdio.h` y `stdlib.h`, están en un lenguaje de alto nivel porque contiene palabras entendibles.

(c) Observe lo que ocurrió con los comentarios y las directivas de preprocesador. ¿los comentarios se borraron?

4. Ejecute la siguiente instrucción: `gcc -S programa.i`

(a) ¿Qué opción se agrega para que `gcc` muestre todas las advertencias durante este proceso?

La opción `-Wall` indica los errores y advertencias de compilación.

(b) ¿Qué le indica a `gcc` la opción `-S`?

Crear un nuevo archivo llamado `programa.s` que es en lenguaje ensamblador

(c) ¿Que contiene el archivo de salida y cual es su extensión?

Su extensión es .s y contiene el código de programa traducido a ensamblador

5. Ejecute la siguiente instrucción: `as programa.s -o programa.o`

```
.file "programa.c"
.text
.section .rodata
.LC0:
.string "Hola Mundo!"
.LC2:
.string "Resultado : %f\n"
.text
.globl main
.type main, @function
main:
.LFB6:
.cfi_startproc
endbr64
pushq %rbp
.cfi_def_cfa_offset 16
.cfi_offset 6, -16
movq %rsp, %rbp
.cfi_def_cfa_register 6
subq $16, %rsp
leaq .LC0(%rip), %rdi
call puts@PLT
movss .LC1(%rip), %xmm0
movss %xmm0, -4(%rbp)
cvtss2sd -4(%rbp), %xmm0
leaq .LC2(%rip), %rdi
movl $1, %eax
call printf@PLT
movl $0, %eax
leave
.cfi_def_cfa 7, 8
ret
.cfi_endproc
.LFE6:
.size main, .-main
.section .rodata
```

(a) ¿Formule una hipótesis sobre el contenido del archivo .o?

Programa objeto en lenguaje ensamblador

(b) ¿Que representa el contenido del programa .o de acuerdo con el diagrama de un sistema de procesamiento de lenguaje?

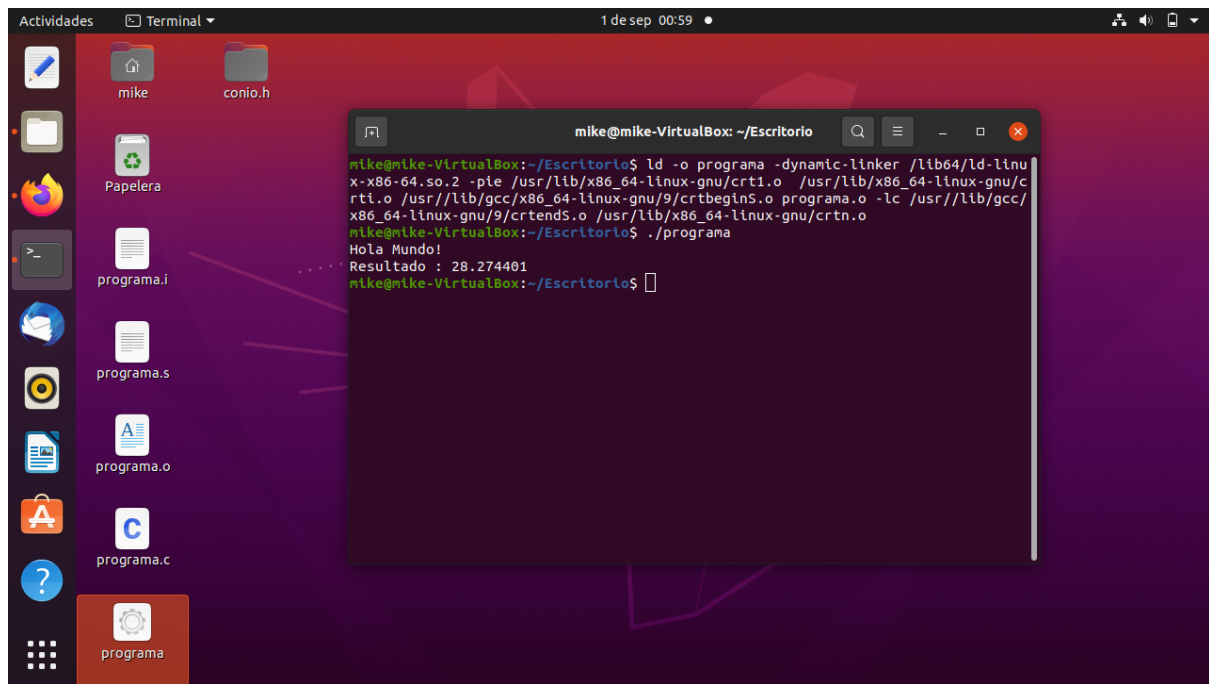
El lenguaje objeto generado por el compilador, que es la entrada para el ensamblador.

6. Encuentre la ruta de los siguientes archivos en el equipo de trabajo:

- Scrt1.o o crt1.o
- crti.o
- crtbeginS.o o crtbegin.o
- crtendS.o o crtend.o
- crtn.o

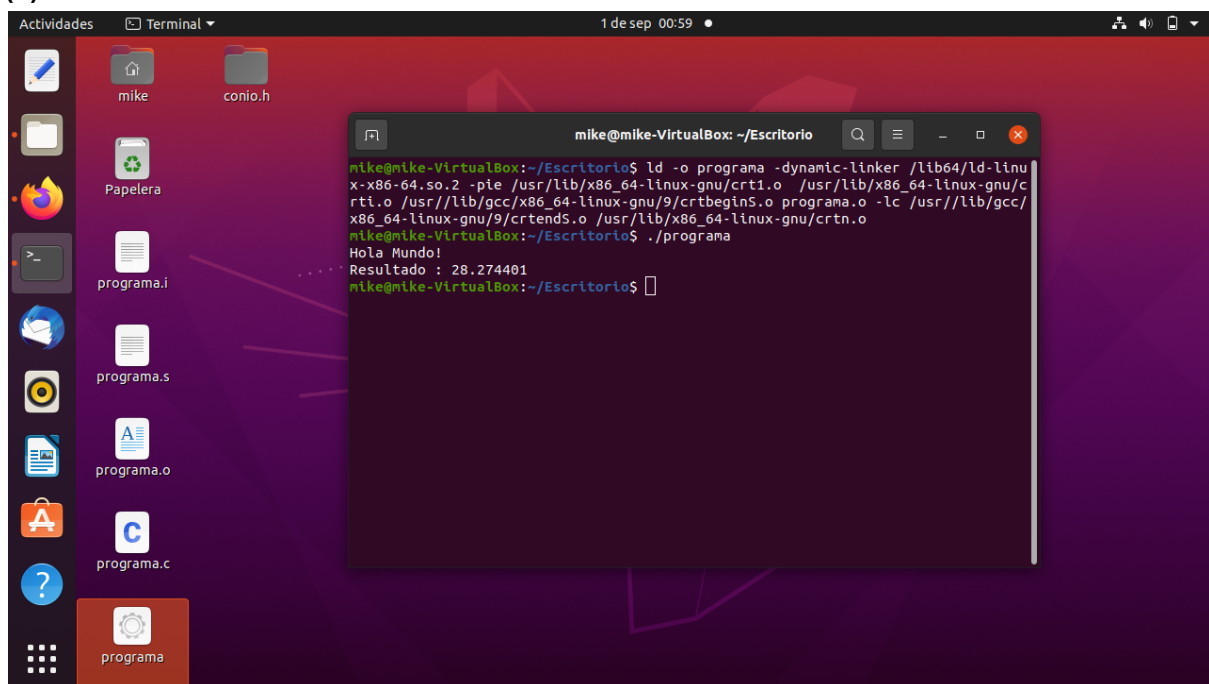
Las primeras dos no las encontré.

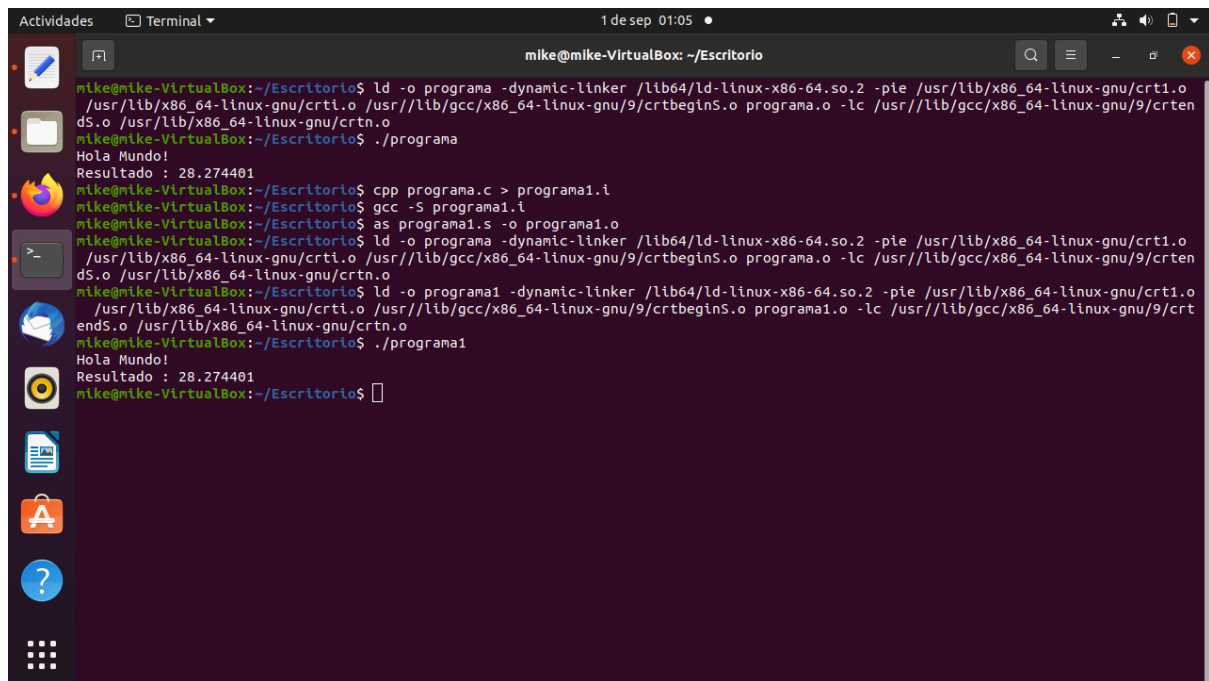
(b) ¿Que se generó al ejecutar el comando anterior?



9. Quite el comentario de la macro #define PI

(a) Genere





```
mike@mike-VirtualBox: ~/Escritorio
mike@mike-VirtualBox:~/Escritorio$ ld -o programa -dynamic-linker /lib64/ld-linux-x86-64.so.2 -pie /usr/lib/x86_64-linux-gnu/crt1.o /usr/lib/x86_64-linux-gnu/crti.o /usr//lib/gcc/x86_64-linux-gnu/9/crtbeginS.o programa.o -lc /usr//lib/gcc/x86_64-linux-gnu/9/crten
ds.o /usr/lib/x86_64-linux-gnu/crtn.o
mike@mike-VirtualBox:~/Escritorio$ ./programa
Hola Mundo!
Resultado : 28.274401
mike@mike-VirtualBox:~/Escritorio$ cpp programa.c > programa1.i
mike@mike-VirtualBox:~/Escritorio$ gcc -S programa1.i
mike@mike-VirtualBox:~/Escritorio$ as programa1.s -o programa1.o
mike@mike-VirtualBox:~/Escritorio$ ld -o programa -dynamic-linker /lib64/ld-linux-x86-64.so.2 -pie /usr/lib/x86_64-linux-gnu/crt1.o /usr/lib/x86_64-linux-gnu/crti.o /usr//lib/gcc/x86_64-linux-gnu/9/crtbeginS.o programa.o -lc /usr//lib/gcc/x86_64-linux-gnu/9/crten
ds.o /usr/lib/x86_64-linux-gnu/crtn.o
mike@mike-VirtualBox:~/Escritorio$ ld -o programa1 -dynamic-linker /lib64/ld-linux-x86-64.so.2 -pie /usr/lib/x86_64-linux-gnu/crt1.o /usr/lib/x86_64-linux-gnu/crti.o /usr//lib/gcc/x86_64-linux-gnu/9/crtbeginS.o programa1.o -lc /usr//lib/gcc/x86_64-linux-gnu/9/crt
ends.o /usr/lib/x86_64-linux-gnu/crtn.o
mike@mike-VirtualBox:~/Escritorio$ ./programa1
Hola Mundo!
Resultado : 28.274401
mike@mike-VirtualBox:~/Escritorio$
```

nuevamente el archivo.i. De preferencia asigne un nuevo nombre.

(b) ¿Cambio algo en la ejecución final?

No

10. Escribe un segundo programa en lenguaje C

En el agregue 4 directivas del preprocesador de C (cpp)*. Las directivas elegidas deben jugar algún papel en el significado del programa, ser distintas entre sí y ser diferentes de las utilizadas en el primer programa (aunque no están prohibidas, si las requieren).

Puede consultar la lista de directivas en su documentación en línea: **CPP - Index of directives**

(<http://gcc.gnu.org/onlinedocs/cpp/Index-of-Directives.html>).

O bien, revisar la entrada para este preprocesador en la herramienta man en Linux : \$ man cpp

(a) Explique la utilidad general de las directivas usadas y su función en particular para su programa.

11. Escriba sus resultados y conclusiones.

Como resultado empecé a imaginar de manera menos abstracta como es que es la comunicación entre el preprocesador, compilador, ensamblador, y el enlazador/cargador, observando cómo en cada paso se genera el archivo visto teóricamente.