Práctica 01

DOCENTE	CARRERA	CURSO
MSc. Vicente Enrique Machaca Arceda	Escuela Profesional de Ingeniería de Software	Compiladores

PRÁCTICA	TEMA	DURACIÓN
01	Introducción	3 horas

1. Datos de los estudiantes

- Grupo: 4
- Integrantes:
  - Gabriela Pacco Huamani
  - Augusto Delgado Bravo
  - Roberto Heredia Garland

2. Ejercicios

1. Redacta el siguiente código, genera el código ensamblador y explica en qué parte (del código ensamblador) se definen las variables c y m. (2 puntos).

```
; 1 Aqui se define la variable "m".
     .LFB0:
    .cfi_startproc
    endbr64
    pushq    %rbp
    .cfi_def_cfa_offset 16
    .cfi_offset 6, -16
    movq     %rsp, %rbp
    .cfi_def_cfa_register 6
    movl     $11148, -4(%rbp)
    movl     $0, %eax
    popq     %rbp
    .cfi_def_cfa 7, 8
    ret
    .cfi_endproc
    ;Aqui se define  la  variable c:
    .LC0:
    .string "abcdef"
    .text
    .globl   main
    .type    main, @function
```

2. Redacta el siguiente código, genera el código ensamblador y explica en qué parte (del código ensamblador) se define la división entre 8. (2 puntos).

```
1
2       .LFE0:
3       .size   main, .-main
4       .ident  "GCC: (Ubuntu 9.3.0-10ubuntu2) 9.3.0"
5       .section        .note.GNU-stack,"",@progbits
6       .section        .note.gnu.property,"a"
7       .align 8
8       .long    1f - 0f
9       .long    4f - 1f
10      .long    5
11  0:
12      .string  "GNU"
13  1:
14      .align 8
15      .long    0xc0000002
16      .long    3f - 2f
17  2:
18      .long    0x3
19  3:
20      .align 8
21  4:
```

3. Redacta el siguiente código, genera el código ensamblador y explica en qué parte (del código ensamblador) se define la división entre 4. (2 puntos).

```
1       cmovs   %edx, %eax
2       sarl    $2, %eax
```

4. Redacta el siguiente código, genera el código ensamblador y explica en qué parte (del código ensamblador) se define la división entre 2. (2 puntos).

```
1       addl    %edx, %eax
2       sarl    %eax
```

5. Redacta el siguiente código, genera el código ensamblador y explica: (4 puntos):

   • En qué parte del código ensamblador se define la función div4.

```
1       Ltext0:
2       .cfi_sections    .debug_frame
3       .globl   __Z4div4i
4       .def     __Z4div4i;   .scl    2;   .type   32; .endef
```

   • En qué parte del código ensamblador se invoca a la función div4.

```
1       call     __Z4div4i
```

   • En qué parte del código ensamblador dentro de la función div4 se procesa la división.

```
1       movl    8(%ebp), %eax
2       leal    3(%eax), %edx
3       testl   %eax, %eax
4       cmovs   %edx, %eax
5       sarl    $2, %eax
```

6. Redacta el siguiente código, genera el código ensamblador y explica: (4 puntos):
   - En qué parte del código ensamblador se define la función div.

```
1       Ltext0:
2       .cfi_sections    .debug_frame
3       .globl   __Z3divii
4       .def     __Z3divii;   .scl    2;   .type    32; .endef
```

   - En qué parte del código ensamblador se invoca a la función div.

```
1   call     __Z3divii
```

   - En qué parte del código ensamblador dentro de la función div se procesa la división.

```
1   idivl    12(%ebp)
```

7. De las preguntas anteriores, se ha generado código, por cada función, ambas dividen entre 4, pero difieren un poco en su implementación. Investigue a qué se debe dicha diferencia y comente cuáles podrían ser las consecuencias. (4 puntos)

   - En la funcion *div* se utiliza *idivl* la cual es una instruccion para dividir signed numbers, por eso se utiliza cltd (lo cual convierte longs a dobuble longs

```
1   cltd
2   idivl    12(%ebp)
```

   - En cambio en *div*4 se utiliza una operacion bitwise al dividir (Shift Arithmetic Right)

```
1       sarl     $2, %eax
```

8. Github

   https://github.com/Robertohg/Compiladores