

# Práctica 01

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

PRÁCTICA	TEMA	DURACIÓN
01	Introducción	3 horas

## 1. Datos de los estudiantes

- **■** Grupo: 1
- Git Hub: https://github.com/CrazyDani17/Practica1-Compiladores
- Integrantes:
  - Guillermo Alemán
  - Marvik Del Carpio
  - Daniel Mendiguri
  - Daniela Vilchez

# 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).

```
int main(){
    char* c = "abcdef";
    int m = 11148;
    return 0;
}
```

Código Assembler:

```
" code.c"
           . file
           . def
                                                                           32;
                     ___main;
                                           . scl
                                                      2;
                                                                .type
           . section . rdata ," dr"
LC0:
           . ascii "abcdef\0"
           . text
           .globl
                     _main
           . def
                     _main; .scl
                                           2;
                                                                32;
                                                                           .\,\mathrm{endef}
                                                      .type
_{\mathtt{main}}:
LFB0:
```

.end



```
.cfi_startproc
         pushl
                  % ebp
         . cfi_def_cfa_offset 8
         . cfi_offset 5, -8
                  \% \exp, \% \exp
         movl
         .\ cfi\_def\_cfa\_register\ 5
         andl
                  -16, % esp
                  $16, \% esp
         subl
         call
                  ___main
                  $LC0, 12(%esp)
         movl
                  $11148, 8(%esp)
         movl
         movl
                  \$0, \% eax
         leave
         .cfi_restore 5
         .cfi_def_cfa 4, 4
         ret
         .cfi_endproc
LFE0:
         .ident "GCC: (MinGW.org GCC-6.3.0-1) 6.3.0"
```

Solución

La variable 'c' se define en las líneas detalladas abajo correspondinetes al código Assembler:

Mientras que la variable 'm' corresponde a la siguiente:

```
movl $11148, 8(%esp)
```

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).

```
int main(){
    char* c = "abcdef";
    int m = 11148;
    int x = m/8;
    return 0;
}
```

Código Assembler:

```
. file "ej2.cpp"
. text
. section . rodata
. type _ZStL19piecewise_construct , @object
. size _ZStL19piecewise_construct , 1
_ZStL19piecewise_construct :
```

MSc. Vicente Machaca Compiladores Página 2



```
.zero
                    _ZStL8__ioinit
          .local
                    _{\rm ZStL8\_ioinit} ,1,1
          .comm
.LC0:
          .string "abcdef"
          .text
          .globl
                    main
                    main, @function
          .type
main:
.LFB1493:
          .cfi_startproc
          pushq
                    % rbp
          .cfi_def_cfa_offset 16
          .cfi_offset 6, -16
                    % rsp, % rbp
          .cfi_def_cfa_register 6
                     .LCO(\%rip), \%rax
          leaq
                    \% \; \mathrm{rax} \; , \; \; -8(\,\%\mathrm{rbp}\,)
          movq
                    11148, -16(\% \text{rbp})
          movl
                    -16(\% \text{rbp}), \% \text{ eax} 7(\% \text{rax}), \% \text{ edx}
          movl
          leal
                    \% eax, \% eax
          testl
                    % edx, % eax
          cmovs
                    $3, \% eax
          sarl
                    \% \operatorname{eax}, -12(\%\operatorname{rbp})
          movl
                    $0, % eax
          movl
          popq
                    % rbp
          .cfi_def_cfa 7, 8
          .cfi_endproc
.LFE1493:
          . size
                    main, .-main
          .type
                    _Z41_static_initialization_and_destruction_0ii, @fu
_Z41_static_initialization_and_destruction_0ii:
.LFB1974:
          .cfi_startproc
                    % rbp
          pushq
          .cfi_def_cfa_offset 16
          .cfi_offset 6, -16
                     \% \operatorname{rsp}, \% \operatorname{rbp}
          movq
          .cfi_def_cfa_register 6
          subq
                    $16, % rsp
                    \% edi, -4(\%rbp)
          movl
                    % esi, -8(%rbp)
          movl
                    $1, -4(\% \text{rbp})
          cmpl
                    . L5
          jne
          cmpl
                    65535, -8(\% \text{rbp})
          jne
                    _ZStL8__ioinit(%rip), %rdi
          leaq
                    _{\rm ZNSt8ios\_base4InitC1Ev@PLT}
          call
                    __dso_handle(%rip), %rdx
          leaq
                    _ZStL8__ioinit(%rip), % rsi
          leaq
```



```
movq
                 _ZNSt8ios_base4InitD1Ev@GOTPCREL(%rip), % rax
                 % rax, % rdi
        movq
        call
                 -cxa\_atexit@PLT
.L5:
        nop
        leave
        .cfi_def_cfa 7, 8
        .cfi_endproc
.LFE1974:
                 _Z41_static_initialization_and_destruction_0ii, .-_
        . size
                 _GLOBAL__sub_I_main, @function
        .type
_GLOBAL__sub_I_main:
.LFB1975:
        .cfi_startproc
        pushq
                 % rbp
        .cfi_def_cfa_offset 16
        .cfi_offset 6, -16
                 % rsp, % rbp
        .cfi_def_cfa_register 6
                 $65535, % esi
        movl
                 $1, % edi
        movl
                 _Z41_static_initialization_and_destruction_0ii
        call
                 % rbp
        popq
        .cfi_def_cfa 7, 8
        .cfi_endproc
.LFE1975:
                 _{GLOBAL\_sub\_I\_main}, .-_{GLOBAL\_sub\_I\_main}
        . size
                         .init_array,"aw"
        .section
        .align 8
                 _{\rm GLOBAL\_sub\_I\_main}
        . quad
        .hidden __dso_handle
        .ident "GCC: (Ubuntu 7.5.0-3ubuntu1~18.04) 7.5.0"
                          .note.GNU-stack,"", @progbits
        .section
```

#### Solución

La división entre 8 se da en las siguientes líneas:

```
movl -16(%rbp), % eax leal 7(%rax), % edx testl % eax, % eax cmovs % edx, % eax sarl $3, % eax movl % eax, -12(%rbp)
```

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).

```
int main(){
    char* c = "abcdef";
```



```
int m = 11148;
                      int x = m/8;
                      int y = m/4;
                      int z = m/2;
                      return 0;
Código Assembler:
                                 . file
                                            "ej3.cpp"
                                 .text
                                 .section
                                                       . rodata
                                            _ZStL19piecewise_construct, @object
                                 .type
                                 . size
                                            _ZStL19piecewise_construct, 1
                      _ZStL19piecewise_construct:
                                 .zero
                                            _ZStL8__ioinit
                                 .local
                                            _{\rm ZStL8\_ioinit} ,1,1
                                 . comm
                                 .text
                                 .globl
                                            main
                                            main, @function
                                 .type
                      main:
                      .LFB1493:
                                 .cfi_startproc
                                 pushq
                                            % rbp
                                 .cfi_def_cfa_offset 16
                                 .cfi_offset 6, -16
                                 movq
                                            \% \operatorname{rsp}, \% \operatorname{rbp}
                                 .cfi_def_cfa_register 6
                                 movl
                                            $11148, -16(\% \text{ rbp})
                                            -16(\%\text{rbp}), \% eax
                                 movl
                                            7(\% \text{rax}), \% \text{edx}
                                 leal
                                            \%~eax~,~~\%~eax
                                 testl
                                            \%~edx~,~~\%~eax
                                 cmovs
                                            $3, \% eax
                                 sarl
                                 movl
                                            \% \text{ eax}, -12(\% \text{rbp})
                                            -16(\%\text{rbp}), \% eax
                                 movl
                                            3(\% \text{rax}), \% \text{edx}
                                 leal
                                            \%\;eax\;,\quad\%\;eax
                                 testl
                                            % edx, % eax
                                 cmovs
                                 sarl
                                            $2, \% eax
                                            \%eax, -8(\%rbp)
                                 movl
                                            -16(\% \text{rbp}), \% \text{ eax}
                                 movl
                                            \%~eax~,~~\%~edx
                                 movl
                                            $31, \% edx
                                 shrl
                                            \% \text{ edx}, \% \text{ eax}
                                 addl
                                 sarl
                                            \% \, \mathrm{eax}
                                            \% \operatorname{eax}, -4(\%\operatorname{rbp})
                                 movl
                                            $0, % eax
                                 movl
                                 popq
                                            \% rbp
                                 .cfi_def_cfa 7, 8
                                 ret
```



```
.cfi_endproc
.LFE1493:
                 main, .-main
        .size
                 _Z41_static_initialization_and_destruction_0ii, @fu
_Z41_static_initialization_and_destruction_0ii:
.LFB1974:
        .cfi_startproc
        pushq
                 % rbp
        .cfi_def_cfa_offset 16
        .cfi_offset 6, -16
        movq
                 % rsp, % rbp
        .cfi_def_cfa_register 6
        subq
                 $16, % rsp
        movl
                 \% edi, -4(\%rbp)
                 \% \text{ esi }, -8(\% \text{rbp})
        movl
                 $1, -4(\% \text{rbp})
        cmpl
                 .L5
        jne
                 65535, -8(\% \text{rbp})
        cmpl
        jne
        leaq
                 _ZStL8__ioinit(%rip), %rdi
                 _ZNSt8ios_base4InitC1Ev@PLT
        call
        leaq
                 __dso_handle(%rip), %rdx
                 _ZStL8__ioinit(%rip), % rsi
        leaq
        movq
                 _ZNSt8ios_base4InitD1Ev@GOTPCREL(% rip), % rax
                 % rax, % rdi
        movq
                 _{-c}cxa_atexit@PLT
        call
.L5:
        nop
        leave
        .cfi_def_cfa 7, 8
         .cfi_endproc
.LFE1974:
                 _Z41_static_initialization_and_destruction_0ii, .-_
        . size
        .type
                 _GLOBAL__sub_I_main, @function
_GLOBAL__sub_I_main:
.LFB1975:
        .cfi_startproc
        pushq
                 % rbp
        .cfi_def_cfa_offset 16
        .cfi_offset 6, -16
                 % rsp, % rbp
        .cfi_def_cfa_register 6
        movl
                 $65535, \% esi
                 $1, % edi
        movl
        call
                 _Z41__static_initialization_and_destruction_0ii
        popq
                 % rbp
        .cfi_def_cfa 7, 8
         .cfi_endproc
.LFE1975:
                 _GLOBAL_sub_I_main , .-_GLOBAL_sub_I_main
         . size
```



#### Solución:

La división entre 4 se da en las siguientes líneas:

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

```
int main(){
    char* c = "abcdef";
    int m = 11148;
    int x = m/8;
    int y = m/4;
    int z = m/2;
    return 0;
}
```

Código Assembler:

```
"ej4.cpp"
          . file
          .text
                             . rodata
          .section
                   _ZStL19piecewise_construct, @object
         .type
                   _ZStL19piecewise_construct, 1
          .size
_ZStL19piecewise_construct:
          .zero
          .local
                   _ZStL8__ioinit
          . comm
                   _{\rm ZStL8\_ioinit} ,1,1
         . text
         . globl
                   main
          .type
                   main, @function
main:
.LFB1493:
          .cfi_startproc
         pushq
                   \% rbp
          .cfi_def_cfa_offset 16
          .cfi_offset 6, -16
                   \% \operatorname{rsp}, \% \operatorname{rbp}
         movq
          .cfi_def_cfa_register 6
                   11148, -16(\% \text{rbp})
         movl
```



```
movl
                     -16(\% \text{rbp}), \% \text{ eax}
                     7(\% rax), \% edx
          leal
                      \%\;\mathrm{eax}\;,\;\;\%\;\mathrm{eax}
           testl
                      % edx, % eax
          cmovs
                      $3, \% eax
          sarl
                      \% \operatorname{eax}, -12(\%\operatorname{rbp})
          movl
          movl
                     -16(\% \text{rbp}), \% \text{ eax}
                     3(\% \text{rax}), \% \text{edx}
          leal
                      \%\; eax\;, \quad \%\; eax
           testl
                      % edx, % eax
          cmovs
                      $2, \% eax
          sarl
          movl
                      \% \text{ eax}, -8(\% \text{rbp})
                      -16(\% \text{rbp}), \% \text{ eax}
          movl
                      \%~eax~,~~\%~edx
          movl
                      $31, \% edx
          shrl
          addl
                      %dx, %eax
                      \% \, \mathrm{eax}
          sarl
                      \% \operatorname{eax}, -4(\%\operatorname{rbp})
          movl
                      $0, % eax
          movl
                      \% rbp
          popq
           .cfi_def_cfa 7, 8
           .cfi_endproc
.LFE1493:
           .size
                     main, .-main
                      _Z41_static_initialization_and_destruction_0ii, @fu
_Z41_static_initialization_and_destruction_0ii:
.LFB1974:
           .cfi_startproc
                      \% rbp
          pushq
           .\ cfi\_def\_cfa\_offset\ 16
           .cfi\_offset 6, -16
                      % rsp , % rbp
          movq
           .cfi_def_cfa_register 6
                      $16, % rsp
          subq
                      \% \ \mathrm{edi} \ , \ -4(\,\%\mathrm{rbp}\,)
          movl
                      \% esi, -8(\%rbp)
          movl
                      $1, -4(\% \text{rbp})
          cmpl
                      . L5
          jne
                      65535, -8(\% \text{rbp})
          cmpl
          jne
          leaq
                      _ZStL8__ioinit(%rip), %rdi
                      _{\rm ZNSt8ios\_base4InitC1Ev@PLT}
          call
          leaq
                      __dso_handle(%rip), %rdx
                      _ZStL8__ioinit(%rip), % rsi
          leaq
                      _ZNSt8ios_base4InitD1Ev@GOTPCREL(%rip), %rax
          movq
                      % rax, % rdi
          movq
          call
                      __cxa_atexit@PLT
.L5:
          nop
          leave
           .cfi_def_cfa 7, 8
```



```
ret
        .\ cfi\_endproc
.LFE1974:
                 _Z41_static_initialization_and_destruction_0ii, .-_
                 _GLOBAL__sub_I_main, @function
        .type
_GLOBAL__sub_I_main:
.LFB1975:
        .cfi_startproc
                 % rbp
        pushq
        .cfi_def_cfa_offset 16
        .cfi_offset 6, -16
                 % rsp, % rbp
        .cfi_def_cfa_register 6
                 \$65535, \% esi
        movl
                 $1, % edi
        movl
                 _Z41__static_initialization_and_destruction_0ii
        call
                 \% rbp
        popq
        .cfi_def_cfa 7, 8
        .cfi_endproc
.LFE1975:
        .size
                 _{GLOBAL\_sub\_I\_main}, .-_{GLOBAL\_sub\_I\_main}
        . section
                          .init_array,"aw"
        .align 8
        . quad
                 _GLOBAL__sub_I_main
        .hidden __dso_handle
                 "GCC: (Ubuntu 7.5.0 – 3 ubuntu1~18.04) 7.5.0"
                          .note.GNU-stack,"", @progbits
        .section
```

### Solución:

La división entre 2 se da en las siguientes líneas:

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

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

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

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

```
int div4(int x){
    return x/4;
}
int main(){
    char* c = "abcdef";
    int m = 11148;
```



```
int x = m/8;
                      int y = m/4;
                      int z = m/2;
                      int rpt = div4(5);
                     return 0;
Código Assembler:
                                 . file
                                           "ej5.cpp"
                                 .text
                                 .section
                                                       . rodata
                                            _ZStL19piecewise_construct, @object
                                 .type
                                 . size
                                            _ZStL19piecewise_construct, 1
                      _ZStL19piecewise_construct:
                                 .zero
                                            _ZStL8__ioinit
                                 .local
                                            _{\rm ZStL8\_ioinit} ,1,1
                                 . comm
                                 .text
                                 .globl
                                           _{\rm Z}4{
m div}4{
m i}
                                            _Z4div4i, @function
                                 .type
                      _{\rm Z}4{\rm div}4{\rm i}:
                      .LFB1493:
                                 .\ cfi\_startproc
                                pushq
                                            % rbp
                                 .cfi_def_cfa_offset 16
                                 .cfi_offset 6, -16
                                movq
                                            \% \operatorname{rsp}, \% \operatorname{rbp}
                                 .cfi_def_cfa_register 6
                                            \% edi , -4(\%rbp)
                                movl
                                           -4(%rbp), % eax 3(%rax), % edx
                                movl
                                 leal
                                            \% eax, \% eax
                                 testl
                                            \% \text{ edx}, \% \text{ eax}
                                cmovs
                                            $2, % eax
                                 sarl
                                            % rbp
                                popq
                                 .cfi_def_cfa 7, 8
                                 .cfi_endproc
                      .LFE1493:
                                 .size
                                            _{\mathsf{Z}}4\operatorname{div}4\mathrm{i}, .-_{\mathsf{Z}}4\operatorname{div}4\mathrm{i}
                                 .globl
                                           main
                                           main, @function
                                 .type
                      main:
                      .LFB1494:
                                 .cfi_startproc
                                pushq
                                            \% rbp
                                 .cfi_def_cfa_offset 16
                                 .cfi_offset 6, -16
                                            \% \operatorname{rsp}, \% \operatorname{rbp}
                                movq
                                 .cfi_def_cfa_register 6
                                subq
                                            $32, \% rsp
```



```
movl
                     $11148, -20(\% \text{rbp})
                     -20(\%rbp), % eax
          movl
                     7(\% rax), \% edx
          leal
                     \% \ \mathrm{eax} \ , \ \ \% \ \mathrm{eax}
          testl
                     \% \text{ edx}, \% \text{ eax}
          cmovs
                     $3, % eax
          sarl
          movl
                     \% \text{ eax}, -16(\% \text{rbp})
                     -20(\%\text{rbp}), \% eax
          movl
                     3(%rax), % edx
          leal
                     \%\;\mathrm{eax}\;,\  \  \, \%\;\mathrm{eax}
          testl
                     % edx, % eax
          cmovs
          sarl
                     $2, \% eax
          movl
                     \% \operatorname{eax}, -12(\%\operatorname{rbp})
                     -20(\%rbp), \% eax
          movl
                     \% eax, \% edx
          movl
                     $31, \% edx
          shrl
          addl
                     % edx, % eax
          sarl
                     \% eax
                     \% \text{ eax}, -8(\% \text{rbp})
          movl
                     $5, \% edi
          movl
                     _{\rm Z}4{
m div}4{
m i}
          call
                     \% \operatorname{eax}, -4(\%\operatorname{rbp})
          movl
                     $0, % eax
          movl
          leave
          .cfi_def_cfa 7, 8
           .cfi_endproc
.LFE1494:
          .size
                     main, .-main
                     _Z41_static_initialization_and_destruction_0ii, @fu
_Z41_static_initialization_and_destruction_0ii:
.LFB1975:
           .cfi_startproc
          pushq
                     % rbp
          .cfi_def_cfa_offset 16
          .cfi_offset 6, -16
                     \% \operatorname{rsp}, \% \operatorname{rbp}
          movq
          .cfi_def_cfa_register 6
          subq
                     $16, % rsp
                     \% edi , -4(\%rbp)
          movl
                     % esi, -8(%rbp)
          movl
                     $1, -4(\% \text{rbp})
          cmpl
                     .L7
          jne
          cmpl
                     $65535, -8(\%rbp)
          jne
          leaq
                     _ZStL8__ioinit(%rip), %rdi
          call
                     _ZNSt8ios_base4InitC1Ev@PLT
          leaq
                     __dso_handle(%rip), %rdx
                     _ZStL8__ioinit(%rip), % rsi
          leaq
                     _ZNSt8ios_base4InitD1Ev@GOTPCREL(%rip), %rax
          movq
                     % rax, % rdi
          movq
                     __cxa_atexit@PLT
          call
```



```
.L7:
        nop
        leave
        .cfi_def_cfa 7, 8
        .cfi_endproc
.LFE1975:
                 _Z41_static_initialization_and_destruction_0ii, .-_
                 _GLOBAL__sub_I__Z4div4i, @function
        .type
_GLOBAL__sub_I__Z4div4i:
.LFB1976:
        .cfi_startproc
        pushq
                \% rbp
        .cfi_def_cfa_offset 16
        .cfi_offset 6, -16
                 % rsp, % rbp
        movq
        .cfi_def_cfa_register 6
                 \$65535, \% esi
        movl
                 $1, % edi
        movl
                 \_Z41\_\_static\_initialization\_and\_destruction\_0ii
        call
                 % rbp
        popq
        .cfi_def_cfa 7, 8
        .cfi_endproc
.LFE1976:
                 \_GLOBAL\_\_sub\_I\_\_Z4div4i \;, \;\; .-\_GLOBAL\_\_sub\_I\_\_Z4div4i
        .size
        .section
                          .init_array,"aw"
        .align 8
        . quad
                 _GLOBAL__sub_I__Z4div4i
        .hidden __dso_handle
        .ident
                 "GCC: (Ubuntu 7.5.0-3ubuntu1~18.04) 7.5.0"
                          .note.GNU-stack,"", @progbits
        .section
```

### Soluciones:

La función div4 se define en las siguientes líneas:

```
_ZStL19piecewise_construct:
.zero
.local
          _ZStL8__ioinit
. comm
          _{\rm ZStL8\_ioinit}, 1, 1
. text
          _Z4div4i
. globl
.type
          _Z4div4i, @function
_{\rm Z}4{\rm div}4{\rm i}:
.LFB1493:
          .cfi_startproc
          pushq
                    \% rbp
          .cfi_def_cfa_offset 16
          .cfi_offset 6, -16
                   \% \ rsp \ , \quad \% \ rbp
          movq
          .cfi_def_cfa_register 6
                    \% \text{ edi }, -4(\% \text{rbp})
          movl
```



```
movl -4(%rbp), % eax leal 3(%rax), % edx testl % eax, % eax cmovs %edx, % eax sarl $2, % eax popq % rbp . cfi_def_cfa 7, 8 ret . cfi_endproc
```

La función div4 se invoca en las siguientes líneas:

```
\begin{array}{ccc} movl & \$5\,, \ \% \ edi \\ call & \_Z4 \ div4i \end{array}
```

La división se proecesa en las siguientes líneas

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

En qué parte del código ensamblador se define la función div.

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

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

```
int div(int x, int y){
    return x/y;
    }
int div4(int x){
    return x/4;
    }
int main(){
    char* c = "abcdef";
    int m = 11148;
    int x = m/8;
    int y = m/4;
    int z = m/2;
    int rpt = div(5,4);
    int rpt2 = div4(5);
    return 0;
}
```

Código Assembler:



```
. file
               "ej6.cpp"
     .text
                         . rodata
     .section
               _ZStL19piecewise_construct, @object
     .type
     . \operatorname{size}
               _ZStL19piecewise_construct, 1
_ZStL19piecewise_construct:
     .zero
     .local
               _ZStL8__ioinit
               _{\rm ZStL8\_ioinit} ,1,1
     . comm
     .text
     . globl
               _Z8divii
     .type
               _Z8divii, @function
_Z8divii:
.LFB1493:
     .cfi_startproc
               % rbp
    pushq
     .cfi_def_cfa_offset 16
     .cfi_offset 6, -16
               % rsp, % rbp
     .cfi_def_cfa_register 6
               \% edi, -4(\%rbp)
    movl
               \% esi, -8(\%rbp)
    movl
               -4(\% \text{rbp}), \% \text{ eax}
    movl
     cltd
               -8(\% \text{rbp})
     idivl
               \% rbp
    popq
     .cfi_def_cfa 7, 8
     .cfi_endproc
.LFE1493:
               _{\mathbf{Z}}8 divii, .-_{\mathbf{Z}}8 divii
     .size
               _{\rm Z}4{
m div}4{
m i}
     .globl
     .type
               _Z4div4i, @function
_{\rm Z}4{
m div}4{
m i}:
.LFB1494:
     .cfi_startproc
    pushq
               % rbp
     .cfi_def_cfa_offset 16
     .cfi_offset 6, -16
               % rsp , % rbp
    movq
     .cfi_def_cfa_register 6
               \% \text{ edi }, -4(\% \text{rbp})
    movl
              -4(\%rbp), % eax 3(\%rax), % edx
    movl
     leal
               \% eax, \% eax
     testl
               % edx, % eax
    cmovs
               $2, % eax
     sarl
               % rbp
    popq
     .cfi_def_cfa 7, 8
     .cfi_endproc
.\,\mathrm{LFE} 1494:
```



```
.size
                _{Z}4div4i, .-_{Z}4div4i
     .globl
                main
                main, @function
     .type
main:
.LFB1495:
     . cfi_startproc
     pushq
                % rbp
     .cfi_def_cfa_offset 16
     .cfi_offset 6, -16
                \% \operatorname{rsp}, \% \operatorname{rbp}
     movq
     .cfi_def_cfa_register 6
     subq
                $32, \% rsp
                11148, -20(\%rbp)
     movl
                -20(\%rbp), \% eax
     movl
                7(%rax), % edx
     leal
                % eax, % eax
     testl
                % edx, % eax
     cmovs
                $3, \% eax
     sarl
                \% \text{ eax}, -16(\% \text{rbp})
     movl
                -20(%rbp), %eax 3(%rax), %edx
     movl
     leal
                \% \ \mathrm{eax} \ , \ \ \% \ \mathrm{eax}
     testl
                % edx, % eax
     cmovs
                $2, %eax
     sarl
                \% \operatorname{eax}, -12(\%\operatorname{rbp})
     movl
                -20(\% \text{rbp}), \% \text{ eax}
     movl
                \% eax, \% edx
     movl
                $31, \% edx
     shrl
                % edx, % eax
     addl
     sarl
                \% eax
     movl
                \% \operatorname{eax}, -8(\%\operatorname{rbp})
                \$4, \% esi \$5, \% edi
     movl
     movl
                _{\rm Z}8\,{\rm divii}
     call
                \% \operatorname{eax}, -4(\%\operatorname{rbp})
     movl
                $0, % eax
     movl
     leave
     .cfi_def_cfa 7, 8
     .cfi_endproc
.LFE1495:
     . size
                _Z41_static_initialization_and_destruction_0ii, @fu
     .type
_Z41_static_initialization_and_destruction_0ii:
.LFB1976:
     .cfi_startproc
     pushq
                % rbp
     .\ cfi\_def\_cfa\_offset\ 16
     .cfi_offset 6, -16
                % rsp, % rbp
     movq
     .cfi_def_cfa_register 6
```

subq

\$16, %rsp



```
movl
             \% edi, -4(\%rbp)
             \% esi, -8(\%rbp)
    movl
             $1, -4(\% \text{rbp})
    cmpl
             . L9
    jne
             $65535, -8(\%rbp)
    cmpl
    jne
    leaq
             _ZStL8__ioinit(%rip), %rdi
    call
             _ZNSt8ios_base4InitC1Ev@PLT
             __dso_handle(%rip), %rdx
    leaq
             _ZStL8__ioinit(%rip), % rsi
    leaq
             _ZNSt8ios_base4InitD1Ev@GOTPCREL(%rip), % rax
    movq
             % rax, % rdi
    movq
             __cxa_atexit@PLT
    call
.L9:
    nop
    leave
    .cfi_def_cfa 7, 8
    .cfi_endproc
.LFE1976:
             _Z41_static_initialization_and_destruction_0ii, .-_
    .size
    .type
             _GLOBAL__sub_I__Z8divii, @function
_GLOBAL__sub_I__Z8divii:
.LFB1977:
    .cfi_startproc
    pushq
             % rbp
    .cfi_def_cfa_offset 16
    .cfi_offset 6, -16
             \%\; rsp\;,\;\; \%\; rbp
    movq
    .cfi_def_cfa_register 6
             $65535, \% esi
    movl
             $1, % edi
    movl
    call
             _Z41__static_initialization_and_destruction_0ii
             % rbp
    popq
    .cfi_def_cfa 7, 8
    .\ cfi\_endproc
. LFE1977:
    .size
             _GLOBAL__sub_I__Z8divii, .-_GLOBAL__sub_I__Z8divii
    .section
                     .init_array,"aw"
    .align 8
             _GLOBAL__sub_I__Z8divii
    . quad
    .hidden __dso_handle
    .ident
            "GCC: (Ubuntu 7.5.0-3ubuntu1~18.04) 7.5.0"
                     .note.GNU-stack,"", @progbits
    . section
```

Soluciones:

La función div se define en las siguientes líneas:

\_ZStL19piecewise\_construct:



```
.zero
         _ZStL8__ioinit
.local
         _{\rm ZStL8\_ioinit} ,1,1
. comm
.text
.globl
         _Z8divii
         _Z8divii, @function
.type
_Z8divii:
.LFB1493:
.\ cfi\_startproc
pushq
         % rbp
.cfi_def_cfa_offset 16
. cfi_offset 6, -16
movq
         % rsp, % rbp
.cfi_def_cfa_register 6
         \% edi, -4(\%rbp)
movl
         % esi, −8(%rbp)
movl
         -4(\%rbp), % eax
movl
cltd
idivl
         -8(\% \text{rbp})
         \% rbp
popq
.cfi_def_cfa 7, 8
.cfi_endproc
```

La función div se invoca en las siguientes líneas:

```
movl $4, % esi
movl $5, % edi
call _Z8divii
```

La división se procesa en las siguientes líneas:

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)

La función Div recibe dos parámetros los cuales representan el divisor y el dividendo y estas son variables o sea que podríamos dividir entre cualquier número, mientras que en la función Div4 sólo recibe un parámetro, el cual representa el divisor y en la función se divide entre una constante 4.

MSc. Vicente Machaca Compiladores Página 17