Practica 1

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

PRÁCTICA	TEMA	DURACIÓN
01	Introducción	3 horas

Integrantes:

Carlos Corrales

Leonardo Deza

1. Ejercicios

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

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

return 0;
}
```

```
        .file   "1.cpp"
        .text
        .section .rdata,"dr"
_ZStL19piecewise_construct:
        .space 1
.lcomm _ZStL8__ioinit,1,1
        .def    __main; .scl    2;      .type   32;     .endef
.LC0:
        .ascii "abcdef\0"
        .text
        .globl  main
        .def    main;   .scl    2;      .type   32;     .endef
        .seh_proc       main
main:
.LFB1573:
        pushq   %rbp
        .seh_pushreg    %rbp
        pushq   %rbx
        .seh_pushreg    %rbx
        subq    $88, %rsp
        .seh_stackalloc 88
        leaq    128(%rsp), %rbp
        .seh_setframe   %rbp, 128
        .seh_endprologue
        call    __main
        leaq    -53(%rbp), %rax
        movq    %rax, %rcx
        call    _ZNSaIcEC1Ev
        leaq    -53(%rbp), %rdx
        leaq    -96(%rbp), %rax
        movq    %rdx, %r8
        leaq    .LC0(%rip), %rdx
        movq    %rax, %rcx
.LEHB0:
        call    _ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEC1EPKcRKS3_
.LEHE0:
        leaq    -53(%rbp), %rax
        movq    %rax, %rcx
        call    _ZNSaIcED1Ev
        movl    $11148, -52(%rbp)
        movl    $0, %ebx
        leaq    -96(%rbp), %rax
        movq    %rax, %rcx
        call    _ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEED1Ev
        movl    %ebx, %eax
        jmp     .L5
.L4:
        movq    %rax, %rbx
        leaq    -53(%rbp), %rax
        movq    %rax, %rcx
        call    _ZNSaIcED1Ev
        movq    %rbx, %rax
        movq    %rax, %rcx
.LEHB1:
        call    _Unwind_Resume
.LEHE1:
.L5:
        addq    $88, %rsp
        popq    %rbx
        popq    %rbp
        ret
        .def    __gxx_personality_seh0; .scl    2;      .type   32;     .endef
        .seh_handler    __gxx_personality_seh0, @unwind, @except
        .seh_handlerdata
.LLSDA1573:
        .byte   0xff
        .byte   0xff
        .byte   0x1
        .uleb128 .LLSDACSE1573-.LLSDACSB1573
```

```
.LLSDACSB1573:
        .uleb128 .LEHB0-.LFB1573
        .uleb128 .LEHE0-.LEHB0
        .uleb128 .L4-.LFB1573
        .uleb128 0
        .uleb128 .LEHB1-.LFB1573
        .uleb128 .LEHE1-.LEHB1
        .uleb128 0
        .uleb128 0
.LLSDACSE1573:
        .text
        .seh_endproc
        .def    __tcf_0;        .scl    3;      .type   32;     .endef
        .seh_proc       __tcf_0
__tcf_0:
.LFB2058:
        pushq   %rbp
        .seh_pushreg    %rbp
        movq    %rsp, %rbp
        .seh_setframe   %rbp, 0
        subq    $32, %rsp
        .seh_stackalloc 32
        .seh_endprologue
        leaq    _ZStL8__ioinit(%rip), %rcx
        call    _ZNSt8ios_base4InitD1Ev
        nop
        addq    $32, %rsp
        popq    %rbp
        ret
        .seh_endproc
        .def    _Z41__static_initialization_and_destruction_0ii;        .scl    3;      .type   32;     .endef
        .seh_proc       _Z41__static_initialization_and_destruction_0ii
_Z41__static_initialization_and_destruction_0ii:
.LFB2057:
        pushq   %rbp
        .seh_pushreg    %rbp
        movq    %rsp, %rbp
        .seh_setframe   %rbp, 0
        subq    $32, %rsp
        .seh_stackalloc 32
        .seh_endprologue
        movl    %ecx, 16(%rbp)
        movl    %edx, 24(%rbp)
        cmpl    $1, 16(%rbp)
        jne     .L9
        cmpl    $65535, 24(%rbp)
        jne     .L9
        leaq    _ZStL8__ioinit(%rip), %rcx
        call    _ZNSt8ios_base4InitC1Ev
        leaq    __tcf_0(%rip), %rcx
        call    atexit
.L9:
        nop
        addq    $32, %rsp
        popq    %rbp
        ret
        .seh_endproc
        .def    _GLOBAL__sub_I_main;    .scl    3;      .type   32;     .endef
        .seh_proc       _GLOBAL__sub_I_main
_GLOBAL__sub_I_main:
.LFB2059:
        pushq   %rbp
        .seh_pushreg    %rbp
        movq    %rsp, %rbp
        .seh_setframe   %rbp, 0
        subq    $32, %rsp
        .seh_stackalloc 32
        .seh_endprologue
        movl    $65535, %edx
        movl    $1, %ecx
        call    _Z41__static_initialization_and_destruction_0ii
        nop
        addq    $32, %rsp
        popq    %rbp
        ret
        .seh_endproc
        .section        .ctors,"w"
        .align 8
        .quad   _GLOBAL__sub_I_main
        .ident  "GCC: (x86_64-posix-seh-rev0, Built by MinGW-W64 project) 8.1.0"
        .def    _ZNSaIcEC1Ev;   .scl    2;      .type   32;     .endef
        .def    _ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEC1EPKcRKS3_;       .scl    2;      .type   32;     .endef
        .def    _ZNSaIcED1Ev;   .scl    2;      .type   32;     .endef
        .def    _ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEED1Ev;      .scl    2;      .type   32;     .endef
        .def    _Unwind_Resume; .scl    2;      .type   32;     .endef
        .def    _ZNSt8ios_base4InitD1Ev;        .scl    2;      .type   32;     .endef
        .def    _ZNSt8ios_base4InitC1Ev;        .scl    2;      .type   32;     .endef
        .def    atexit; .scl    2;      .type   32;     .endef
```

Respuesta

```
.ascii "abcdef\0"      movl    $11148, -52(%rbp)
```

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

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

return 0;
}
```

```
.LLSDA1573:
        .byte   0xff
        .byte   0xff
        .byte   0x1
        .uleb128 .LLSDACSE1573-.LLSDACSB1573
.LLSDACSB1573:
        .uleb128 .LEHB0-.LFB1573
        .uleb128 .LEHE0-.LEHB0
        .uleb128 .L4-.LFB1573
        .uleb128 0
        .uleb128 .LEHB1-.LFB1573
        .uleb128 .LEHE1-.LEHB1
        .uleb128 0
        .uleb128 0
.LLSDACSE1573:
        .text
        .seh_endproc
        .def    __tcf_0;        .scl    3;      .type   32;     .endef
        .seh_proc       __tcf_0
__tcf_0:
.LFB2058:
        pushq   %rbp
        .seh_pushreg    %rbp
        movq    %rsp, %rbp
        .seh_setframe   %rbp, 0
        subq    $32, %rsp
        .seh_stackalloc 32
        .seh_endprologue
        leaq    _ZStL8__ioinit(%rip), %rcx
        call    _ZNSt8ios_base4InitD1Ev
        nop
        addq    $32, %rsp
        popq    %rbp
        ret
        .seh_endproc
        .def    _Z41__static_initialization_and_destruction_0ii;        .scl    3;      .type   32;     .endef
        .seh_proc       _Z41__static_initialization_and_destruction_0ii
_Z41__static_initialization_and_destruction_0ii:
.LFB2057:
        pushq   %rbp
        .seh_pushreg    %rbp
        movq    %rsp, %rbp
        .seh_setframe   %rbp, 0
        subq    $32, %rsp
        .seh_stackalloc 32
        .seh_endprologue
        movl    %ecx, 16(%rbp)
        movl    %edx, 24(%rbp)
        cmpl    $1, 16(%rbp)
        jne     .L9
        cmpl    $65535, 24(%rbp)
        jne     .L9
        leaq    _ZStL8__ioinit(%rip), %rcx
        call    _ZNSt8ios_base4InitC1Ev
        leaq    __tcf_0(%rip), %rcx
        call    atexit
.L9:
        nop
        addq    $32, %rsp
        popq    %rbp
        ret
        .seh_endproc
        .def    _GLOBAL__sub_I_main;    .scl    3;      .type   32;     .endef
        .seh_proc       _GLOBAL__sub_I_main
_GLOBAL__sub_I_main:
.LFB2059:
        pushq   %rbp
        .seh_pushreg    %rbp
        movq    %rsp, %rbp
        .seh_setframe   %rbp, 0
        subq    $32, %rsp
        .seh_stackalloc 32
        .seh_endprologue
        movl    $65535, %edx
        movl    $1, %ecx
        call    _Z41__static_initialization_and_destruction_0ii
        nop
        addq    $32, %rsp
        popq    %rbp
        ret
        .seh_endproc
        .section        .ctors,"w"
        .align 8
        .quad   _GLOBAL__sub_I_main
        .ident  "GCC: (x86_64-posix-seh-rev0, Built by MinGW-W64 project) 8.1.0"
        .def    _ZNSaIcEC1Ev;   .scl    2;      .type   32;     .endef
        .def    _ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEC1EPKcRKS3_;       .scl    2;      .type   32;     .endef
        .def    _ZNSaIcED1Ev;   .scl    2;      .type   32;     .endef
        .def    _ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEED1Ev;      .scl    2;      .type   32;     .endef
        .def    _Unwind_Resume; .scl    2;      .type   32;     .endef
        .def    _ZNSt8ios_base4InitD1Ev;        .scl    2;      .type   32;     .endef
        .def    _ZNSt8ios_base4InitC1Ev;        .scl    2;      .type   32;     .endef
        .def    atexit; .scl    2;      .type   32;     .endef
```

```
        .file   "2.cpp"
        .text
        .section .rdata,"dr"
_ZStL19piecewise_construct:
        .space 1
.lcomm _ZStL8__ioinit,1,1
        .def    __main; .scl    2;      .type   32;     .endef
.LC0:
        .ascii "abcdef\0"
        .text
        .globl  main
        .def    main;   .scl    2;      .type   32;     .endef
        .seh_proc       main
main:
.LFB1573:
        pushq   %rbp
        .seh_pushreg    %rbp
        pushq   %rbx
        .seh_pushreg    %rbx
        subq    $88, %rsp
        .seh_stackalloc 88
        leaq    128(%rsp), %rbp
        .seh_setframe   %rbp, 128
        .seh_endprologue
        call    __main
        leaq    -57(%rbp), %rax
        movq    %rax, %rcx
        call    _ZNSaIcEC1Ev
        leaq    -57(%rbp), %rdx
        leaq    -96(%rbp), %rax
        movq    %rdx, %r8
        leaq    .LC0(%rip), %rdx
        movq    %rax, %rcx
.LEHB0:
        call    _ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEC1EPKcRKS3_
.LEHE0:
        leaq    -57(%rbp), %rax
        movq    %rax, %rcx
        call    _ZNSaIcED1Ev
        movl    $11148, -52(%rbp)
        movl    -52(%rbp), %eax
        leal    7(%rax), %edx
        testl   %eax, %eax
        cmovs   %edx, %eax
        sarl    $3, %eax
        movl    %eax, -56(%rbp)
        movl    $0, %ebx
        leaq    -96(%rbp), %rax
        movq    %rax, %rcx
        call    _ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEED1Ev
        movl    %ebx, %eax
        jmp     .L5
.L4:
        movq    %rax, %rbx
        leaq    -57(%rbp), %rax
        movq    %rax, %rcx
        call    _ZNSaIcED1Ev
        movq    %rbx, %rax
        movq    %rax, %rcx
.LEHB1:
        call    _Unwind_Resume
.LEHE1:
.L5:
        addq    $88, %rsp
        popq    %rbx
        popq    %rbp
        ret
        .def    __gxx_personality_seh0; .scl    2;      .type   32;     .endef
        .seh_handler    __gxx_personality_seh0, @unwind, @except
        .seh_handlerdata
```

Respuesta

```
movl    -52(%rbp), %eax
leal    7(%rax), %edx
testl   %eax, %eax
cmovs   %edx, %eax
sarl    $3, %eax
movl    %eax, -56(%rbp)
```

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

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

```
        .file   "3.cpp"
        .text
        .section .rdata,"dr"
_ZStL19piecewise_construct:
        .space 1
.lcomm _ZStL8__ioinit,1,1
        .def    __main; .scl    2;      .type   32;     .endef
.LC0:
        .ascii "abcdef\0"
        .text
        .globl  main
        .def    main;   .scl    2;      .type   32;     .endef
        .seh_proc       main
main:
.LFB1573:
        pushq   %rbp
        .seh_pushreg    %rbp
        pushq   %rbx
        .seh_pushreg    %rbx
        subq    $104, %rsp
        .seh_stackalloc 104
        leaq    128(%rsp), %rbp
        .seh_setframe   %rbp, 128
        .seh_endprologue
        call    __main
        leaq    -49(%rbp), %rax
        movq    %rax, %rcx
        call    _ZNSaIcEC1Ev
        leaq    -49(%rbp), %rdx
        leaq    -96(%rbp), %rax
        movq    %rdx, %r8
        leaq    .LC0(%rip), %rdx
        movq    %rax, %rcx
.LEHB0:
        call    _ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEC1EPKcRKS3_
.LEHE0:
        leaq    -49(%rbp), %rax
        movq    %rax, %rcx
        call    _ZNSaIcED1Ev
        movl    $11148, -36(%rbp)
        movl    -36(%rbp), %eax
        leal    7(%rax), %edx
        testl   %eax, %eax
        cmovs   %edx, %eax
        sarl    $3, %eax
        movl    %eax, -40(%rbp)
        movl    -36(%rbp), %eax
        leal    3(%rax), %edx
        testl   %eax, %eax
        cmovs   %edx, %eax
        sarl    $2, %eax
        movl    %eax, -44(%rbp)
        movl    -36(%rbp), %eax
        movl    %eax, %edx
        shrl    $31, %edx
        addl    %edx, %eax
        sarl    %eax
        movl    %eax, -48(%rbp)
        movl    $0, %ebx
        leaq    -96(%rbp), %rax
        movq    %rax, %rcx
        call    _ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEED1Ev
        movl    %ebx, %eax
        jmp     .L5
.L4:
        movq    %rax, %rbx
        leaq    -49(%rbp), %rax
        movq    %rax, %rcx
        call    _ZNSaIcED1Ev
        movq    %rbx, %rax
        movq    %rax, %rcx
.LEHB1:
        call    _Unwind_Resume
```

```
.LEHE1:
.L5:
        addq    $104, %rsp
        popq    %rbx
        popq    %rbp
        ret
        .def    __gxx_personality_seh0; .scl    2;      .type   32;     .endef
        .seh_handler    __gxx_personality_seh0, @unwind, @except
        .seh_handlerdata
.LLSDA1573:
        .byte   0xff
        .byte   0xff
        .byte   0x1
        .uleb128 .LLSDACSE1573-.LLSDACSB1573
.LLSDACSB1573:
        .uleb128 .LEHB0-.LFB1573
        .uleb128 .LEHE0-.LEHB0
        .uleb128 .L4-.LFB1573
        .uleb128 0
        .uleb128 .LEHB1-.LFB1573
        .uleb128 .LEHE1-.LEHB1
        .uleb128 0
        .uleb128 0
.LLSDACSE1573:
        .text
        .seh_endproc
        .def    __tcf_0;        .scl    3;      .type   32;     .endef
        .seh_proc       __tcf_0
__tcf_0:
.LFB2058:
        pushq   %rbp
        .seh_pushreg    %rbp
        movq    %rsp, %rbp
        .seh_setframe   %rbp, 0
        subq    $32, %rsp
        .seh_stackalloc 32
        .seh_endprologue
        leaq    _ZStL8__ioinit(%rip), %rcx
        call    _ZNSt8ios_base4InitD1Ev
        nop
        addq    $32, %rsp
        popq    %rbp
        ret
        .seh_endproc
        .def    _Z41__static_initialization_and_destruction_0ii;        .scl    3;      .type   32;     .endef
        .seh_proc       _Z41__static_initialization_and_destruction_0ii
_Z41__static_initialization_and_destruction_0ii:
.LFB2057:
        pushq   %rbp
        .seh_pushreg    %rbp
        movq    %rsp, %rbp
        .seh_setframe   %rbp, 0
        subq    $32, %rsp
        .seh_stackalloc 32
        .seh_endprologue
        movl    %ecx, 16(%rbp)
        movl    %edx, 24(%rbp)
        cmpl    $1, 16(%rbp)
        jne     .L9
        cmpl    $65535, 24(%rbp)
        jne     .L9
        leaq    _ZStL8__ioinit(%rip), %rcx
        call    _ZNSt8ios_base4InitC1Ev
        leaq    __tcf_0(%rip), %rcx
        call    atexit
```

```
.L9:
        nop
        addq    $32, %rsp
        popq    %rbp
        ret
        .seh_endproc
        .def    _GLOBAL__sub_I_main;    .scl    3;      .type   32;     .endef
        .seh_proc       _GLOBAL__sub_I_main
_GLOBAL__sub_I_main:
.LFB2059:
        pushq   %rbp
        .seh_pushreg    %rbp
        movq    %rsp, %rbp
        .seh_setframe   %rbp, 0
        subq    $32, %rsp
        .seh_stackalloc 32
        .seh_endprologue
        movl    $65535, %edx
        movl    $1, %ecx
        call    _Z41__static_initialization_and_destruction_0ii
        nop
        addq    $32, %rsp
        popq    %rbp
        ret
        .seh_endproc
        .section        .ctors,"w"
        .align 8
        .quad   _GLOBAL__sub_I_main
        .ident  "GCC: (x86_64-posix-seh-rev0, Built by MinGW-W64 project) 8.1.0"
        .def    _ZNSaIcEC1Ev;   .scl    2;      .type   32;     .endef
        .def    _ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEC1EPKcRKS3_;       .scl    2;      .type   32;     .endef
        .def    _ZNSaIcED1Ev;   .scl    2;      .type   32;     .endef
        .def    _ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEED1Ev;      .scl    2;      .type   32;     .endef
        .def    _Unwind_Resume; .scl    2;      .type   32;     .endef
        .def    _ZNSt8ios_base4InitD1Ev;        .scl    2;      .type   32;     .endef
        .def    _ZNSt8ios_base4InitC1Ev;        .scl    2;      .type   32;     .endef
        .def    atexit; .scl    2;      .type   32;     .endef
```

Respuesta

movl -36(%rbp), %eax

leal 3(%rax), %edx

testl %eax, %eax

cmovs %edx, %eax

sarl $2, %eax

movl %eax, -44(%rbp)

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

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

```
        .file   "3.cpp"
        .text
        .section .rdata,"dr"
_ZStL19piecewise_construct:
        .space 1
.lcomm _ZStL8__ioinit,1,1
        .def    __main; .scl    2;      .type   32;     .endef
.LC0:
        .ascii "abcdef\0"
        .text
        .globl  main
        .def    main;   .scl    2;      .type   32;     .endef
        .seh_proc       main
main:
.LFB1573:
        pushq   %rbp
        .seh_pushreg    %rbp
        pushq   %rbx
        .seh_pushreg    %rbx
        subq    $104, %rsp
        .seh_stackalloc 104
        leaq    128(%rsp), %rbp
        .seh_setframe   %rbp, 128
        .seh_endprologue
        call    __main
        leaq    -49(%rbp), %rax
        movq    %rax, %rcx
        call    _ZNSaIcEC1Ev
        leaq    -49(%rbp), %rdx
        leaq    -96(%rbp), %rax
        movq    %rdx, %r8
        leaq    .LC0(%rip), %rdx
        movq    %rax, %rcx
.LEHB0:
        call    _ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEC1EPKcRKS3_
.LEHE0:
        leaq    -49(%rbp), %rax
        movq    %rax, %rcx
        call    _ZNSaIcED1Ev
        movl    $11148, -36(%rbp)
        movl    -36(%rbp), %eax
        leal    7(%rax), %edx
        testl   %eax, %eax
        cmovs   %edx, %eax
        sarl    $3, %eax
        movl    %eax, -40(%rbp)
        movl    -36(%rbp), %eax
        leal    3(%rax), %edx
        testl   %eax, %eax
        cmovs   %edx, %eax
        sarl    $2, %eax
        movl    %eax, -44(%rbp)
        movl    -36(%rbp), %eax
        movl    %eax, %edx
        shrl    $31, %edx
        addl    %edx, %eax
        sarl    %eax
        movl    %eax, -48(%rbp)
        movl    $0, %ebx
        leaq    -96(%rbp), %rax
        movq    %rax, %rcx
        call    _ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEED1Ev
        movl    %ebx, %eax
        jmp     .L5
.L4:
        movq    %rax, %rbx
        leaq    -49(%rbp), %rax
        movq    %rax, %rcx
        call    _ZNSaIcED1Ev
        movq    %rbx, %rax
        movq    %rax, %rcx
.LEHB1:
        call    _Unwind_Resume
```

```
.LEHE1:
.L5:
        addq    $104, %rsp
        popq    %rbx
        popq    %rbp
        ret
        .def    __gxx_personality_seh0; .scl    2;      .type   32;     .endef
        .seh_handler    __gxx_personality_seh0, @unwind, @except
        .seh_handlerdata
.LLSDA1573:
        .byte   0xff
        .byte   0xff
        .byte   0x1
        .uleb128 .LLSDACSE1573-.LLSDACSB1573
.LLSDACSB1573:
        .uleb128 .LEHB0-.LFB1573
        .uleb128 .LEHE0-.LEHB0
        .uleb128 .L4-.LFB1573
        .uleb128 0
        .uleb128 .LEHB1-.LFB1573
        .uleb128 .LEHE1-.LEHB1
        .uleb128 0
        .uleb128 0
.LLSDACSE1573:
        .text
        .seh_endproc
        .def    __tcf_0;        .scl    3;      .type   32;     .endef
        .seh_proc       __tcf_0
__tcf_0:
.LFB2058:
        pushq   %rbp
        .seh_pushreg    %rbp
        movq    %rsp, %rbp
        .seh_setframe   %rbp, 0
        subq    $32, %rsp
        .seh_stackalloc 32
        .seh_endprologue
        leaq    _ZStL8__ioinit(%rip), %rcx
        call    _ZNSt8ios_base4InitD1Ev
        nop
        addq    $32, %rsp
        popq    %rbp
        ret
        .seh_endproc
        .def    _Z41__static_initialization_and_destruction_0ii;        .scl    3;      .type   32;     .endef
        .seh_proc       _Z41__static_initialization_and_destruction_0ii
_Z41__static_initialization_and_destruction_0ii:
.LFB2057:
        pushq   %rbp
        .seh_pushreg    %rbp
        movq    %rsp, %rbp
        .seh_setframe   %rbp, 0
        subq    $32, %rsp
        .seh_stackalloc 32
        .seh_endprologue
        movl    %ecx, 16(%rbp)
        movl    %edx, 24(%rbp)
        cmpl    $1, 16(%rbp)
        jne     .L9
        cmpl    $65535, 24(%rbp)
        jne     .L9
        leaq    _ZStL8__ioinit(%rip), %rcx
        call    _ZNSt8ios_base4InitC1Ev
        leaq    __tcf_0(%rip), %rcx
        call    atexit
```

```
.L9:
        nop
        addq    $32, %rsp
        popq    %rbp
        ret
        .seh_endproc
        .def    _GLOBAL__sub_I_main;    .scl    3;      .type   32;     .endef
        .seh_proc       _GLOBAL__sub_I_main
_GLOBAL__sub_I_main:
.LFB2059:
        pushq   %rbp
        .seh_pushreg    %rbp
        movq    %rsp, %rbp
        .seh_setframe   %rbp, 0
        subq    $32, %rsp
        .seh_stackalloc 32
        .seh_endprologue
        movl    $65535, %edx
        movl    $1, %ecx
        call    _Z41__static_initialization_and_destruction_0ii
        nop
        addq    $32, %rsp
        popq    %rbp
        ret
        .seh_endproc
        .section        .ctors,"w"
        .align 8
        .quad   _GLOBAL__sub_I_main
        .ident  "GCC: (x86_64-posix-seh-rev0, Built by MinGW-W64 project) 8.1.0"
        .def    _ZNSaIcEC1Ev;   .scl    2;      .type   32;     .endef
        .def    _ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEC1EPKcRKS3_;       .scl    2;      .type   32;     .endef
        .def    _ZNSaIcED1Ev;   .scl    2;      .type   32;     .endef
        .def    _ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEED1Ev;      .scl    2;      .type   32;     .endef
        .def    _Unwind_Resume; .scl    2;      .type   32;     .endef
        .def    _ZNSt8ios_base4InitD1Ev;        .scl    2;      .type   32;     .endef
        .def    _ZNSt8ios_base4InitC1Ev;        .scl    2;      .type   32;     .endef
        .def    atexit; .scl    2;      .type   32;     .endef
```

Respuesta

movl -36(%rbp), %eax

movl %eax, %edx

shrl $31, %edx

addl %edx, %eax

sarl %eax

movl %eax, -48(%rbp)

1.5 Redacta el siguiente código, genera el código ensamblador y explica:

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.

```c
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;
}
```

```
        .file   "5.cpp"
        .text
        .section .rdata,"dr"
_ZStL19piecewise_construct:
        .space 1
.lcomm _ZStL8__ioinit,1,1
        .text
        .globl  _Z4div4i
        .def    _Z4div4i;       .scl    2;      .type   32;     .endef
        .seh_proc       _Z4div4i
_Z4div4i:
.LFB1573:
        pushq   %rbp
        .seh_pushreg    %rbp
        movq    %rsp, %rbp
        .seh_setframe   %rbp, 0
        .seh_endprologue
        movl    %ecx, 16(%rbp)
        movl    16(%rbp), %eax
        leal    3(%rax), %edx
        testl   %eax, %eax
        cmovs   %edx, %eax
        sarl    $2, %eax
        popq    %rbp
        ret
        .seh_endproc
        .def    __main; .scl    2;      .type   32;     .endef
        .section .rdata,"dr"
.LC0:
        .ascii "abcdef\0"
        .text
        .globl  main
        .def    main;   .scl    2;      .type   32;     .endef
        .seh_proc       main
main:
.LFB1574:
        pushq   %rbp
        .seh_pushreg    %rbp
        pushq   %rbx
        .seh_pushreg    %rbx
        subq    $104, %rsp
        .seh_stackalloc 104
        leaq    128(%rsp), %rbp
        .seh_setframe   %rbp, 128
        .seh_endprologue
        call    __main
        leaq    -53(%rbp), %rax
        movq    %rax, %rcx
        call    _ZNSaIcEC1Ev
        leaq    -53(%rbp), %rdx
        leaq    -96(%rbp), %rax
        movq    %rdx, %r8
        leaq    .LC0(%rip), %rdx
        movq    %rax, %rcx
.LEHB0:
        call    _ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEC1EPKcRKS3_
.LEHE0:
        leaq    -53(%rbp), %rax
        movq    %rax, %rcx
        call    _ZNSaIcED1Ev
        movl    $11148, -36(%rbp)
        movl    -36(%rbp), %eax
        leal    7(%rax), %edx
        testl   %eax, %eax
        cmovs   %edx, %eax
        sarl    $3, %eax
        movl    %eax, -40(%rbp)
        movl    -36(%rbp), %eax
        leal    3(%rax), %edx
        testl   %eax, %eax
        cmovs   %edx, %eax
        sarl    $2, %eax
        movl    %eax, -44(%rbp)
        movl    -36(%rbp), %eax
        movl    %eax, %edx
```

```
        movl    %eax, %edx
        shrl    $31, %edx
        addl    %edx, %eax
        sarl    %eax
        movl    %eax, -48(%rbp)
        movl    $5, %ecx
        call    _Z4div4i
        movl    %eax, -52(%rbp)
        movl    $0, %ebx
        leaq    -96(%rbp), %rax
        movq    %rax, %rcx
        call    _ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEED1Ev
        movl    %ebx, %eax
        jmp     .L7
.L6:
        movq    %rax, %rbx
        leaq    -53(%rbp), %rax
        movq    %rax, %rcx
        call    _ZNSaIcED1Ev
        movq    %rbx, %rax
        movq    %rax, %rcx
.LEHB1:
        call    _Unwind_Resume
.LEHE1:
.L7:
        addq    $104, %rsp
        popq    %rbx
        popq    %rbp
        ret
        .def    __gxx_personality_seh0; .scl    2;      .type   32;     .endef
        .seh_handler    __gxx_personality_seh0, @unwind, @except
        .seh_handlerdata
.LLSDA1574:
        .byte   0xff
        .byte   0xff
        .byte   0x1
        .uleb128 .LLSDACSE1574-.LLSDACSB1574
.LLSDACSB1574:
        .uleb128 .LEHB0-.LFB1574
        .uleb128 .LEHE0-.LEHB0
        .uleb128 .L6-.LFB1574
        .uleb128 0
        .uleb128 .LEHB1-.LFB1574
        .uleb128 .LEHE1-.LEHB1
        .uleb128 0
        .uleb128 0
.LLSDACSE1574:
        .text
        .seh_endproc
        .def    __tcf_0;        .scl    3;      .type   32;     .endef
        .seh_proc       __tcf_0
__tcf_0:
.LFB2059:
        pushq   %rbp
        .seh_pushreg    %rbp
        movq    %rsp, %rbp
        .seh_setframe   %rbp, 0
        subq    $32, %rsp
        .seh_stackalloc 32
        .seh_endprologue
        leaq    _ZStL8__ioinit(%rip), %rcx
        call    _ZNSt8ios_base4InitD1Ev
        nop
        addq    $32, %rsp
        popq    %rbp
        ret
        .seh_endproc
        .def    _Z41__static_initialization_and_destruction_0ii;        .scl    3;      .type   32;     .endef
        .seh_proc       _Z41__static_initialization_and_destruction_0ii
_Z41__static_initialization_and_destruction_0ii:
.LFB2058:
        pushq   %rbp
        .seh_pushreg    %rbp
        movq    %rsp, %rbp
```

```
        .seh_setframe   %rbp, 0
        subq    $32, %rsp
        .seh_stackalloc 32
        .seh_endprologue
        movl    %ecx, 16(%rbp)
        movl    %edx, 24(%rbp)
        cmpl    $1, 16(%rbp)
        jne     .L11
        cmpl    $65535, 24(%rbp)
        jne     .L11
        leaq    _ZStL8__ioinit(%rip), %rcx
        call    _ZNSt8ios_base4InitC1Ev
        leaq    __tcf_0(%rip), %rcx
        call    atexit
.L11:
        nop
        addq    $32, %rsp
        popq    %rbp
        ret
        .seh_endproc
        .def    _GLOBAL__sub_I__Z4div4i;        .scl    3;      .type   32;     .endef
        .seh_proc       _GLOBAL__sub_I__Z4div4i
_GLOBAL__sub_I__Z4div4i:
.LFB2060:
        pushq   %rbp
        .seh_pushreg    %rbp
        movq    %rsp, %rbp
        .seh_setframe   %rbp, 0
        subq    $32, %rsp
        .seh_stackalloc 32
        .seh_endprologue
        movl    $65535, %edx
        movl    $1, %ecx
        call    _Z41__static_initialization_and_destruction_0ii
        nop
        addq    $32, %rsp
        popq    %rbp
        ret
        .seh_endproc
        .section        .ctors,"w"
        .align 8
        .quad   _GLOBAL__sub_I__Z4div4i
        .ident  "GCC: (x86_64-posix-seh-rev0, Built by MinGW-W64 project) 8.1.0"
        .def    _ZNSaIcEC1Ev;   .scl    2;      .type   32;     .endef
        .def    _ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEC1EPKcRKS3_;       .scl    2;      .type   32;     .endef
        .def    _ZNSaIcED1Ev;   .scl    2;      .type   32;     .endef
        .def    _ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEED1Ev;      .scl    2;      .type   32;     .endef
        .def    _Unwind_Resume; .scl    2;      .type   32;     .endef
        .def    _ZNSt8ios_base4InitD1Ev;        .scl    2;      .type   32;     .endef
        .def    _ZNSt8ios_base4InitC1Ev;        .scl    2;      .type   32;     .endef
        .def    atexit; .scl    2;      .type   32;     .endef
```

Respuesta:

1.-

lcomm ZStL8_ioinit,1,1

.text

.globl _Z4div4i

.def _Z4div4i; .scl 2; .type 32; .endef

.seh_proc _Z4div4i

2.-

call _Z4div4i

3.-

.LFB1573:

pushq %rbp

```
.seh_pushreg    %rbp
movq    %rsp, %rbp
.seh_setframe    %rbp, 0
.seh_endprologue
movl    %ecx, 16(%rbp)
movl    16(%rbp), %eax
leal    3(%rax), %edx
testl   %eax, %eax
cmovs   %edx, %eax
sarl    $2, %eax
popq    %rbp
ret
.seh_endproc
.def    __main;         .scl    2;      .type   32;     .endef
.section .rdata,"dr"
```

1.6 Redacta el siguiente código, genera el código ensamblador y explica:

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;
}
```

```
        .file   "Ejercicio6.cpp"
        .text
        .globl  __Z3divii
        .def    __Z3divii;      .scl    2;      .type   32;     .endef
__Z3divii:
LFB0:
        .cfi_startproc
        pushl   %ebp
        .cfi_def_cfa_offset 8
        .cfi_offset 5, -8
        movl    %esp, %ebp
        .cfi_def_cfa_register 5
        movl    8(%ebp), %eax
        cltd
        idivl   12(%ebp)
        popl    %ebp
        .cfi_restore 5
        .cfi_def_cfa 4, 4
        ret
        .cfi_endproc
LFE0:
        .globl  __Z4div4i
        .def    __Z4div4i;      .scl    2;      .type   32;     .endef
__Z4div4i:
LFB1:
        .cfi_startproc
        pushl   %ebp
        .cfi_def_cfa_offset 8
        .cfi_offset 5, -8
        movl    %esp, %ebp
        .cfi_def_cfa_register 5
        movl    8(%ebp), %eax
        cltd
        andl    $3, %edx
        addl    %edx, %eax
        sarl    $2, %eax
        popl    %ebp
        .cfi_restore 5
        .cfi_def_cfa 4, 4
        ret
        .cfi_endproc
LFE1:
        .def    ___main;        .scl    2;      .type   32;     .endef
        .section .rdata,"dr"
LC0:
        .ascii "abcdef\0"
        .text
        .globl  _main
        .def    _main;  .scl    2;      .type   32;     .endef
_main:
```

```
LFB2:
        .cfi_startproc
        pushl   %ebp
        .cfi_def_cfa_offset 8
        .cfi_offset 5, -8
        movl    %esp, %ebp
        .cfi_def_cfa_register 5
        andl    $-16, %esp
        subl    $48, %esp
        call    ___main
        movl    $LC0, 44(%esp)
        movl    $11148, 40(%esp)
        movl    40(%esp), %eax
        cltd
        andl    $7, %edx
        addl    %edx, %eax
        sarl    $3, %eax
        movl    %eax, 36(%esp)
        movl    40(%esp), %eax
        cltd
        andl    $3, %edx
        addl    %edx, %eax
        sarl    $2, %eax
        movl    %eax, 32(%esp)
        movl    40(%esp), %eax
        movl    %eax, %edx
        shrl    $31, %edx
        addl    %edx, %eax
        sarl    %eax
        movl    %eax, 28(%esp)
        movl    $4, 4(%esp)
        movl    $5, (%esp)
        call    __Z3divii
        movl    %eax, 24(%esp)
        movl    $5, (%esp)
        call    __Z4div4i
        movl    %eax, 20(%esp)
        movl    $0, %eax
        leave
        .cfi_restore 5
        .cfi_def_cfa 4, 4
        ret
        .cfi_endproc
LFE2:
        .ident  "GCC: (MinGW.org GCC-6.3.0-1) 6.3.0"
```

Respuesta

1.-

.globl __Z3divii

.def __Z3divii; .scl 2; .type 32; .endef

__Z3divii:

2.-

call __Z3divii

3.-

```
__Z3divii:
LFB0:
.cfi_startproc
pushl  %ebp
.cfi_def_cfa_offset 8
.cfi_offset 5, -8
movl  %esp, %ebp
.cfi_def_cfa_register 5
movl  8(%ebp), %eax
cltd
idivl   12(%ebp)
popl   %ebp
.cfi_restore 5
.cfi_def_cfa 4, 4
ret
.cfi_endproc
```

1.7 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.

Primero existe el hecho de que además de dividir entre 4, puede dividir entre cualquier otro número, esto causa que reserve una variable extra, lo que causa una pérdida de memoria para cosas pequeñas, por otro lado, la función div4 divide únicamente entre 4, en ensamblador esta orden la toma simplemente como desplazamiento. Pero volviendo al otro caso este desplazamiento puede ser cualquier numero por lo que antes reserva una variable para esto y luego recién realiza el desplazamiento