2. Escreva um procedimento double cos(double x), em assembly para o MIPS, para calcular o cosseno de um ângulo x, dado em radianos. O procedimento calcula o cosseno usando uma série de Taylor expandida em x = 0 (veja a equação 1). No procedimento, trunque a a série em n = 7 (até o termo x14/14!).
Crie um programa em assembly para o MIPS. O programa permite a entrada de um ângulo x em graus (∘), converte o ângulo para radianos, calcula o cosseno do ângulo usando o procedimento chama o procedimento cos() e imprime o resultado. Use o programa para calcular o seno de 40,67∘. Mostre a saída da execução do seu programa no programa MARS. Verifique se o resultado apresentado pelo seu programa está correto.

Código do programa:

```
.data:
    zero: .double 0.0
    one: .double 1.0
    neg_one: .double -1.0
    pi: .double 3.1415926535897932
    one_80: .double 180.0

.text:

init:
    jal main
    addi $a0, $v0, 0
    addi $v0, $zero, 17
    syscall

#double pot(double x, int pot)
#reg:e
#f16 = x
#f18 = res
#t0 = i
#t1 = pot
pot:
    # prologo:
    mov.d $f16, $f12 #f16 = x
    # $f18 <- 1
    la $t1, one
    lwc1 $f18, 0($t1)
    lwc1 $f19, 4($t1)
    addi $t0, $zero, 0 #i = 0
    addi $t1, $a0, 0

    # corpo:
    j pot_for_teste

pot_for_corpo:
    mul.d $f18, $f16, $f18
    addi $t0, $t0, 1

pot_for_teste:
```

```
    bne $t0, $t1, pot_for_corpo

pot_fim:
    #epilogo
    mov.d $f0, $f18
    jr $ra



#int fat(int n)->int
#reg:
#t0 -> n
#t1 -> i
#t2 -> n2
fat:
    #epilogo:
    addi $t0, $a0, 0 #t0 = n
    addi $t1, $zero, 1 #t1 = 1
    addi $t2, $t0, 0 #t2 = n

    #corpo:
    beq $t0, $zero, fat_zero
    j fat_for_teste

fat_zero:
    addi $v0, $zero, 1
    jr $ra

fat_for_corpo:
    mul $t2, $t2, $t1
    addi $t1, $t1, 1

fat_for_teste:
    bne $t1, $t0, fat_for_corpo

fat_fim:
    addi $v0, $t2, 0
    jr $ra



#double cos(double x): double
#pilha: -48
# $sp+00: x
# $sp+08: num
# $sp+16: den
# $sp+24: mult
# $sp+32: res
# $sp+40: i
# $sp+44: $ra
```

```
#reg:
# $f4: num
# $f6: (float)den
# $f8: mult
# $f10: res
# $f16: x
# $t0: i
# $t2: den

cos:
    # prologo:
    addi $sp, $sp, -48
    #sp+00 <- x
    swc1 $f12, 0($sp)
    swc1 $f13, 4($sp)
    addi $t0, $zero, 0
    #sp+32 <- 0
    la $t1, zero
    lwc1 $f10, 0($t1)
    lwc1 $f11, 4($t1)
    swc1 $f10, 32($sp)
    swc1 $f11, 36($sp)
    #sp+44 = $ra
    sw $ra, 44($sp)

    # corpo:
    j cos_for_teste

cos_for_corpo:
    #salvando tudo na pilha (redundancia)
    swc1 $f4, 8($sp)
    swc1 $f5, 12($sp)
    sw $t2, 16($sp)
    swc1 $f7, 20($sp)
    swc1 $f8, 24($sp)
    swc1 $f9, 28($sp)
    swc1 $f10, 32($sp)
    swc1 $f11, 36($sp)
    sw $t0, 40($sp)

    #num = pot(-1, i)
    la $t1, neg_one
    lwc1 $f12, 0($t1)
    lwc1 $f13, 4($t1)
    lw $a0, 40($sp)
    jal pot
    swc1 $f0, 8($sp)
    swc1 $f1, 12($sp)

    #den = fat(2*i)
```

```
    lw $a0, 40($sp)
    sll $a0, $a0, 1
    jal fat
    sw $v0, 16($sp)

    #mult = pot(x, 2*i)
    lwc1 $f12, 0($sp)
    lwc1 $f13, 4($sp)
    lw $a0, 40($sp)
    sll $a0, $a0, 1
    jal pot
    swc1 $f0, 24($sp)
    swc1 $f1, 28($sp)

    #num = num/den
    lwc1 $f4, 8($sp)
    lwc1 $f5, 12($sp)
    lwc1 $f18, 16($sp)
    cvt.d.w $f6, $f18
    div.d $f4, $f4, $f6

    #num = num * mult
    lwc1 $f8, 24($sp)
    lwc1 $f9, 28($sp)
    mul.d $f4, $f4, $f8

    #res = res + num
    lwc1 $f10, 32($sp)
    lwc1 $f11, 36($sp)
    add.d $f10, $f10, $f4

    #i++
    lw $t0, 40($sp)
    addi $t0, $t0, 1

cos_for_teste:
    slti $t1, $t0, 8
    bne $t1, $zero, cos_for_corpo

cos_fim:
    #epilogo:
    mov.d $f0, $f10 #f0 = res
    lw $ra, 44($sp) #$ra = sp+44
    addi $sp, $sp, 48
    jr $ra


#double grad(double x): rad(x)
#reg:
# $f4: x
```

```
# $f6: pi
# $f8: 180
grad:
    # prologo:
    #$f4 = x
    mov.d $f4, $f12
    #$f6 = pi
    la $t0, pi
    lwc1 $f6, 0($t0)
    lwc1 $f7, 4($t0)
    #$f8 = 180
    la $t0, one_80
    lwc1 $f8, 0($t0)
    lwc1 $f9, 4($t0)

    # corpo:
    div.d $f4, $f4, $f8
    mul.d $f4, $f4, $f6

    #epilogo:
    mov.d $f0, $f4
    jr $ra


#int main()->0
#pilha: -12
#sp+00: x
#sp+08: $ra
main:
    # prologo:
    addi $sp, $sp, -12
    sw $ra, 8($sp)

    # corpo:
    # scanf("%lf", &x)
    addi $v0, $zero, 7
    syscall
    swc1 $f0, 0($sp)
    swc1 $f1, 4($sp)

    # x = grad(x)
    lwc1 $f12, 0($sp)
    lwc1 $f13, 4($sp)
    jal grad
    swc1 $f0, 0($sp)
    swc1 $f1, 4($sp)

    #printf("%lf", x);
    #lwc1 $f12, 0($sp)
    #lwc1 $f13, 4($sp)
```

```
    #addi $v0, $zero, 3
    #syscall

    # x = cos(x)
    lwc1 $f12, 0($sp)
    lwc1 $f13, 4($sp)
    jal cos
    swc1 $f0, 0($sp)
    swc1 $f1, 4($sp)

    #printf("%lf", x);
    lwc1 $f12, 0($sp)
    lwc1 $f13, 4($sp)
    addi $v0, $zero, 3
    syscall

    # epilogo:
    addi $v0, $zero, 0
    lw $ra, 8($sp)
    addi $sp, $sp, 12
    jr $ra
```

Saída do programa (copiada do programa MARS):

```
40.67
0.7584756701781008
-- program is finished running --
```