PROBLEM 1. [10 points] Consider the following MIPS code:

```
addi $8, $0, 1
addi $9, $0, 2
addi $10, $0, 3
beq $16, $8, ONE
beq $16, $9, TWO
beq $16, $10, THREE
addi $2, $0, 0
j EXIT

ONE:    addi $2, $0, 10
        j EXIT
TWO:    addi $2, $0, 50
        j EXIT
THREE:  addi $2, $0, 100

EXIT:
    jr $31
```

1.1 In C-style pseudo-code, describe what this MIPS code computes.

```c
int problemOne(int s0) {
    int t0 = 1;
    int t1 = 2;
    int t2 = 3;
    if (s0 == t0) {
        return 10;
    }
    if (s0 == t1) {
        return 50;
    }
    if (s0 == t2) {
        return 100;
    }
    return 0;
}
```

PROBLEM 2. [20 points] Consider the following MIPS code:
Count:
    addi $sp, $sp, -8
    sw $s0, 4($sp)
    sw $ra, 0($sp)
    bne $a0, $0, else
    add $v0, $0, $0
    addi $sp, $sp, 8
    jr $ra
else:
    andi $s0, $a0, 1
    srl $a0, $a0, 1
    jal Count
    add $v0, $v0, $s0
    lw $ra, 0($sp)
    lw $s0, 4($sp)
    addi $sp, $sp, 8
    jr $ra

2.1 If register $a0 stores the integer 11, what is the value in $v0 after the execution of the above MIPS code?

3

2.2 In C-style pseudo-code, describe what this MIPS code computes.

```c
int Count(int a0) {
    if (a0 == 0) {
        return 0;
    }

    int s0 = a0 & 1;
    a0 = a0 >> 1;
    return Count(a0) + s0;
}
```

PROBLEM 3. [30 points] Assume that, in MIPS assembly, the base address of an array v in stored in $a0 and that the length of that array is stored in $a1.

3.1 Regarding the consecutive words contained in v as signed integers, implement a MIPS function returning the maximum and minimum elements of v in $v0 and $v1, respectively.

```
# Find max and min of an array where a0 is array address, a1 is array length
# returns max in v0, min in v1.
maxMin:
```
addi $sp, $sp, -20  # make room for 5 ints on stack
sw $s4, 16($sp)
sw $s3, 12($sp)
sw $s2, 8($sp)
sw $s1, 4($sp)
sw $s0, 0($sp)

beq $a1, $0, exit1  # if count is 0, done.

add $s2, $a0, $zero  # s2 = a0 (address of v array)
add $s3, $a1, $zero  # s3 = a1 (length of v array)
addi $s4, $zero, 1  # s4 = i = 1

lw $s0 0($s2)       # min = a[0]
lw $s1 0($s2)       # max = a[0]

for1tst:
  slt $t0, $s4, $s3  # t0 = (s0 < s3)
  beq $t0, $zero, exit2

  sll $t1, $s4, 2  # t1 = i * 4
  add $t2, $s2, $t1  # t2 = s2 + (i*4)
lw $t3, 0($t2)  # t3 = a[i];

  slt $t4, $t3, $s0  # t3 < min?
  beq $t4, $zero, testmax
  add $s0, $t3, $zero

  testmax:
    slt $t4, $s1, $t3  # max < t3?
    beq $t4, $zero, doLoop
    add $s1, $t3, $zero

  doLoop:
    addi $s4, $s4, 1
    j for1tst

  exit2:
    add $v0, $s1, $zero  # v0 = max
    add $v1, $s0, $zero  # v1 = min

exit1:
  lw $s0, 0($sp)
lw $s1, 4($sp)
lw $s2, 8($sp)
lw $s3, 12($sp)
lw $s4, 16($sp)
addi $sp, $sp, 20
jr $ra

3.2 Write a complete MIPS program testing your function. Similarly to the program W18_bubble_sort.asm seen in class, the test-array would be defined in the program
itself and you can use:

\[ 5, 18, 9, 1, 3, 14, 10, 4, 6, 7 \]

Your program should work correctly with SPIM. Here are additional references about MIPS: a [tutorial](https://example.com/tutorial), lecture notes, a [MIPS Instruction cheat sheet](https://example.com/cheat-sheet).

SPIM file can be found in `min_max.asm`

**PROBLEM 4.** [40 points] Consider the following definition of the combination numbers, for integers \(0 \leq k \leq n\)

\[
\binom{n}{k} = \frac{n(n-1) \cdots (n-k+1)}{k(k-1) \cdots 1} \tag{1}
\]

see [https://en.wikipedia.org/wiki/Combination](https://en.wikipedia.org/wiki/Combination)

Recall also the recurrence relation:

\[
\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k} \tag{2}
\]

4.1 Implement a function computing \(\binom{n}{k}\), for \(0 \leq k \leq n\), based on Equation (1) in MIPS assembly. To perform the division use `div`, see multiplication and division in MIPS. Try to use a minimum number of MIPS instructions.

```assembly
####
# Find n choose k, given n in a0 and k in a1.
# n >= k
####
combOne:
	slt $t0, $a1, $a0
	beq $t0, $zero, combOneRetZero
	sub $t0, $a0, $a1 #t0 = (n - k + 1) - 1
	add $t1, $a0, $zero #t1 = n
	add $t2, $a1, $zero #t2 = k
	addi $t3, $zero, 1 #numerator prod = 1
	addi $t4, $zero, 1 #denominator prod = 1

for1tst:
	slt $t5, $t0, $t1
	beq $t5, $zero, for1exit
	mult $t3, $t1;
	mflo $t3
	add $t1, $t1, -1
	j for1tst
```

4
for1exit:
for2tst:
    beq $t2, $zero, for2exit
    mult $t4, $t2
    mflo $t4
    addi $t2, $t2, -1
    j for2tst

for2exit:
exitCombOne:
    div $t3, $t4
    mflo $v0
    jr $ra

combOneRetZero:
    add $v0 $zero $zero
    jr $ra

4.2 Implement a function computing \( \binom{n}{k} \), for \( 0 \leq k \leq n \), based on Equation (2) (together with the base-cases \( \binom{n}{0} = 1 \) and \( \binom{n}{n} = 1 \)) in MIPS assembly. Try to use a minimum number of MIPS instructions.

####
# Find n choose k, recursively, given n in a0 and k in a1.
# n >= k
####
combTwo:
    beq $a0, $a1, return1
    beq $a1, $zero, return1
    addi $sp, $sp, -12  # make room for 3 ints on stack
    sw $ra, 8($sp)
    sw $s1, 4($sp)
    sw $s0, 0($sp)
    addi $a0, $a0, -1;  # n = n - 1
    add $s0, $a0, $zero  # s0 = n-1
    add $s1, $a1, $zero  # s1 = k
    jal combTwo
    add $a0, $s0, $zero  # a0 = n - 1
    add $s0, $v0, $zero  # s0 = combTwo(n-1, k);
    addi $a1, $s1, -1;  # a1 = k-1
    jal combTwo
    add $v0, $s0, $v0  # v0 = combTwo(n-1, k) + combTwo(n-1, k-1)
    lw $s0, 0($sp)
    lw $s1, 4($sp)
lw $ra, 8($sp)
addi $sp, $sp, 12
jr $ra

return1:
addi $v0, $zero, 1
jr $ra

4.3 For each of the above functions, write a complete MIPS program testing those functions. Your program should read in the values of \( n \) and \( k \), and print \( \binom{n}{k} \).

SPIM files can be found in `combinationMult.asm` and `combinationRecur.asm`. 