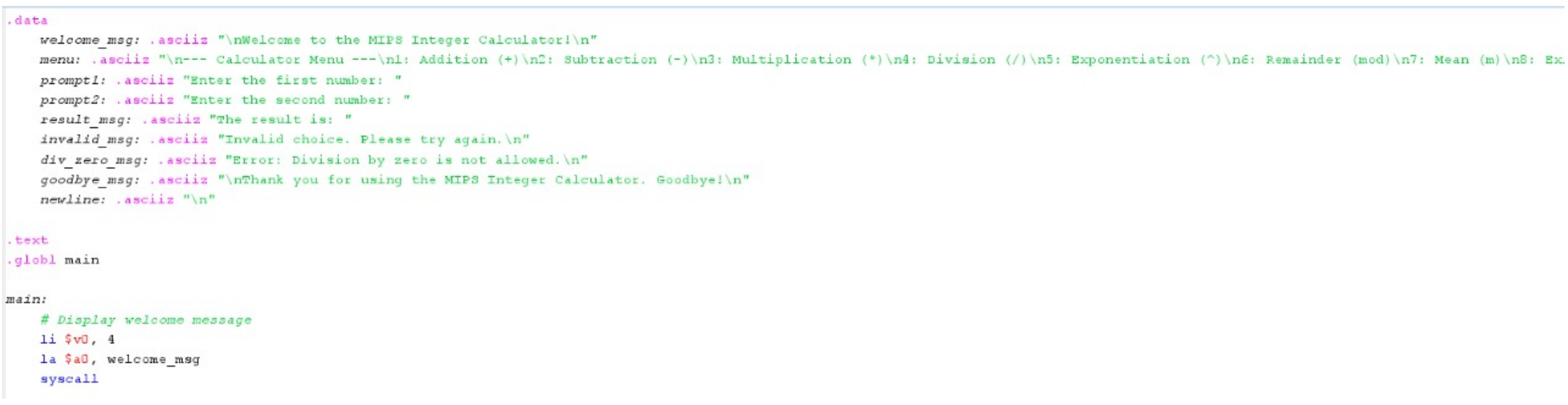
MIPS Calculator.

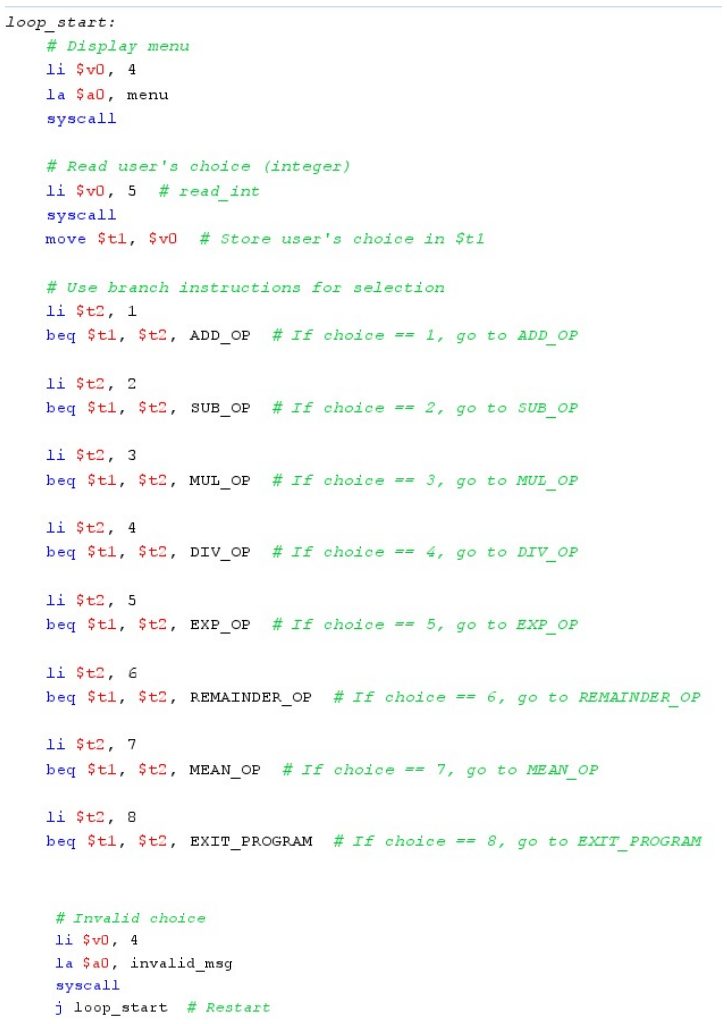
A simple MIPS program to perform basic arithmetic operations on integers.

CCCS-217 DL2
Rimas Almuntashiri - 2311631
Sadeem Awak - 2301922

-Definition of the text data used when intracting with the user and display a welcome message when the program is started.

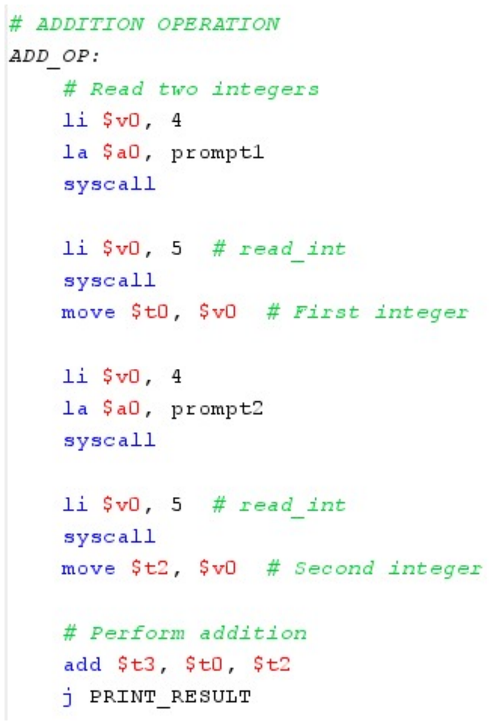


```
.data
    welcome_msg: .asciiz "\nWelcome to the MIPS Integer Calculator!\n"
    menu: .asciiz "\n--- Calculator Menu ---\n1: Addition (+)\n2: Subtraction (-)\n3: Multiplication (*)\n4: Division (/)\n5: Exponentiation (^)\n6: Remainder (mod)\n7: Mean (m)\n8: Ex
    prompt1: .asciiz "Enter the first number: "
    prompt2: .asciiz "Enter the second number: "
    result_msg: .asciiz "The result is: "
    invalid_msg: .asciiz "Invalid choice. Please try again.\n"
    div_zero_msg: .asciiz "Error: Division by zero is not allowed.\n"
    goodbye_msg: .asciiz "\nThank you for using the MIPS Integer Calculator. Goodbye!\n"
    newline: .asciiz "\n"

.text
.globl main

main:
    # Display welcome message
    li $v0, 4
    la $a0, welcome_msg
    syscall
```



```
loop_start:
    # Display menu
    li $v0, 4
    la $a0, menu
    syscall

    # Read user's choice (integer)
    li $v0, 5  # read_int
    syscall
    move $t1, $v0  # Store user's choice in $t1

    # Use branch instructions for selection
    li $t2, 1
    beq $t1, $t2, ADD_OP  # If choice == 1, go to ADD_OP

    li $t2, 2
    beq $t1, $t2, SUB_OP  # If choice == 2, go to SUB_OP

    li $t2, 3
    beq $t1, $t2, MUL_OP  # If choice == 3, go to MUL_OP

    li $t2, 4
    beq $t1, $t2, DIV_OP  # If choice == 4, go to DIV_OP

    li $t2, 5
    beq $t1, $t2, EXP_OP  # If choice == 5, go to EXP_OP

    li $t2, 6
    beq $t1, $t2, REMAINDER_OP  # If choice == 6, go to REMAINDER_OP

    li $t2, 7
    beq $t1, $t2, MEAN_OP  # If choice == 7, go to MEAN_OP

    li $t2, 8
    beq $t1, $t2, EXIT_PROGRAM  # If choice == 8, go to EXIT_PROGRAM


     # Invalid choice
     li $v0, 4
     la $a0, invalid_msg
     syscall
     j loop_start  # Restart
```

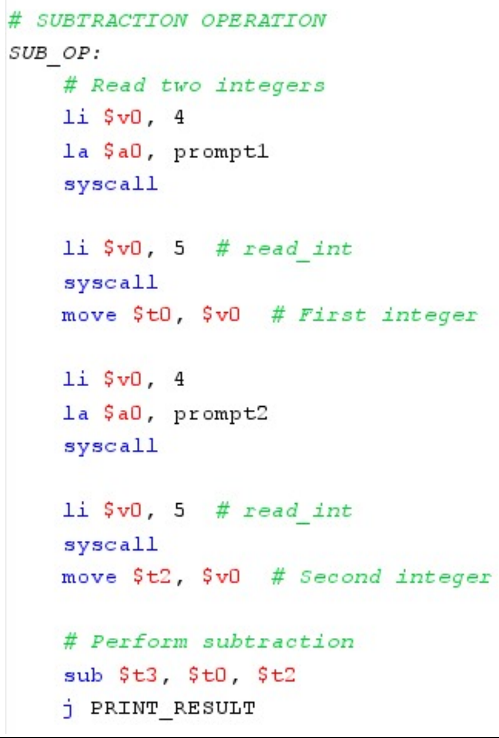
-A loob that displays the menu to the user, where the program reads the user's choice and directs to the appropriate operation.



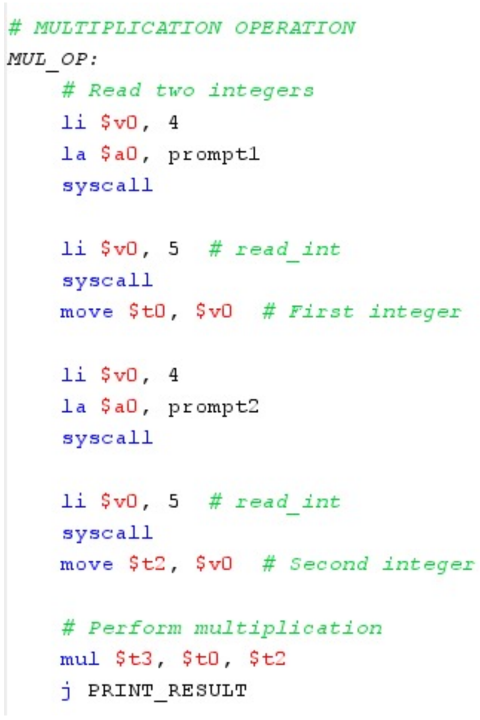
```
# ADDITION OPERATION
ADD_OP:
    # Read two integers
    li $v0, 4
    la $a0, prompt1
    syscall

    li $v0, 5   # read_int
    syscall
    move $t0, $v0   # First integer

    li $v0, 4
    la $a0, prompt2
    syscall

    li $v0, 5   # read_int
    syscall
    move $t2, $v0   # Second integer

    # Perform addition
    add $t3, $t0, $t2
    j PRINT_RESULT
```

-The addition operation:
Reads two integer , adds them and displays the results.

-The subtraction operation:
Reads two integers, subtractacts them and displays the result.



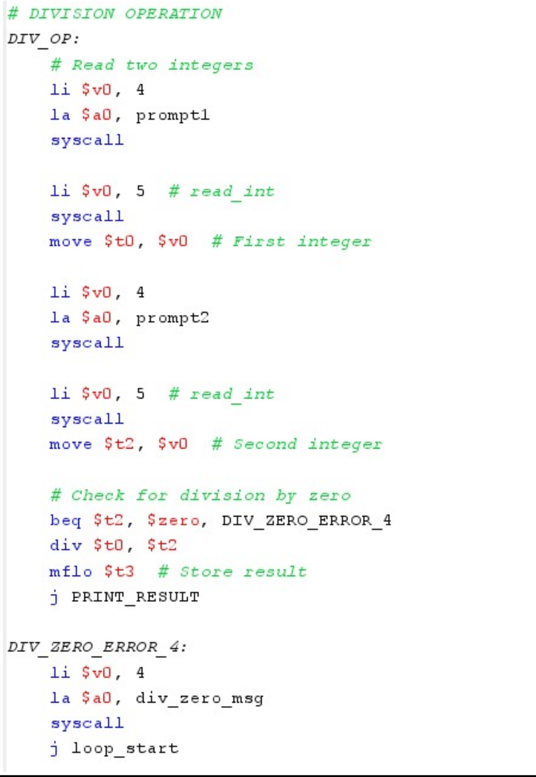
```
# SUBTRACTION OPERATION
SUB_OP:
    # Read two integers
    li $v0, 4
    la $a0, prompt1
    syscall

    li $v0, 5   # read_int
    syscall
    move $t0, $v0   # First integer

    li $v0, 4
    la $a0, prompt2
    syscall

    li $v0, 5   # read_int
    syscall
    move $t2, $v0   # Second integer

    # Perform subtraction
    sub $t3, $t0, $t2
    j PRINT_RESULT
```



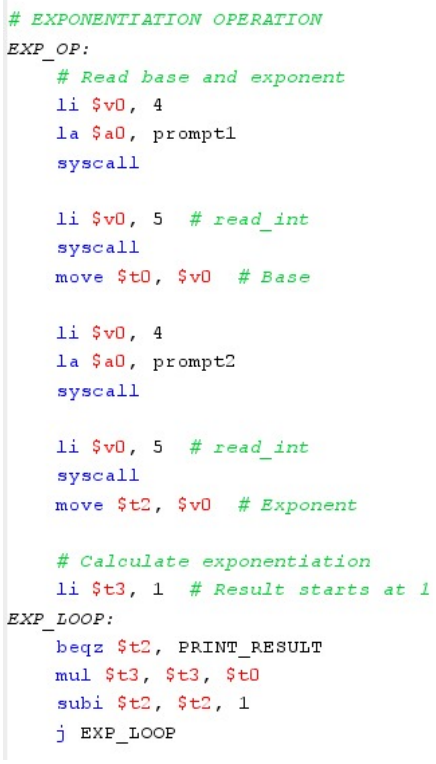
```
# MULTIPLICATION OPERATION
MUL_OP:
    # Read two integers
    li $v0, 4
    la $a0, prompt1
    syscall

    li $v0, 5   # read_int
    syscall
    move $t0, $v0   # First integer

    li $v0, 4
    la $a0, prompt2
    syscall

    li $v0, 5   # read_int
    syscall
    move $t2, $v0   # Second integer

    # Perform multiplication
    mul $t3, $t0, $t2
    j PRINT_RESULT
```

The multiplication operation:
Reads two integers, multiplies them and displays the result.

-The division operation:
Reads two integers, divides the first integer by the second integer and displays the result.



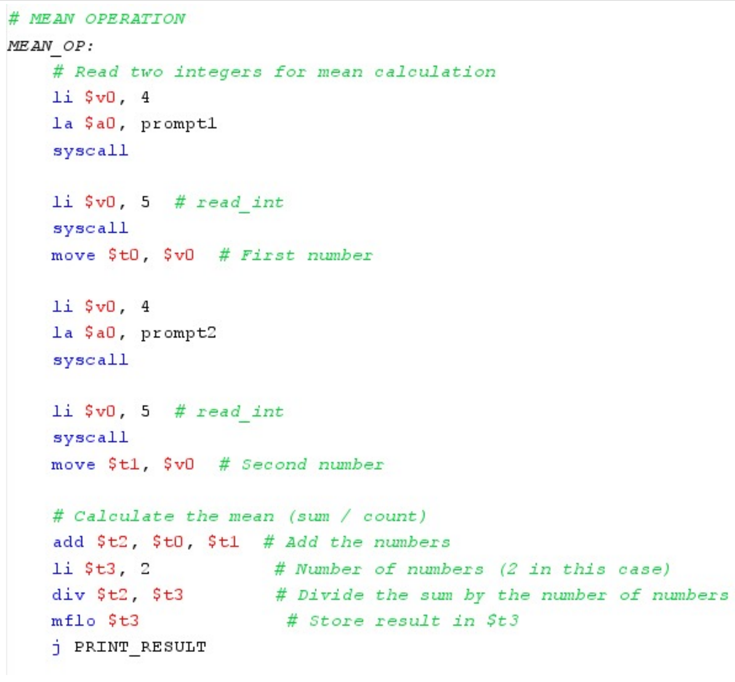
```
# DIVISION OPERATION
DIV_OP:
    # Read two integers
    li $v0, 4
    la $a0, prompt1
    syscall

    li $v0, 5   # read_int
    syscall
    move $t0, $v0   # First integer

    li $v0, 4
    la $a0, prompt2
    syscall

    li $v0, 5   # read_int
    syscall
    move $t2, $v0   # Second integer

    # Check for division by zero
    beq $t2, $zero, DIV_ZERO_ERROR_4
    div $t0, $t2
    mflo $t3   # Store result
    j PRINT_RESULT

DIV_ZERO_ERROR_4:
    li $v0, 4
    la $a0, div_zero_msg
    syscall
    j loop_start
```



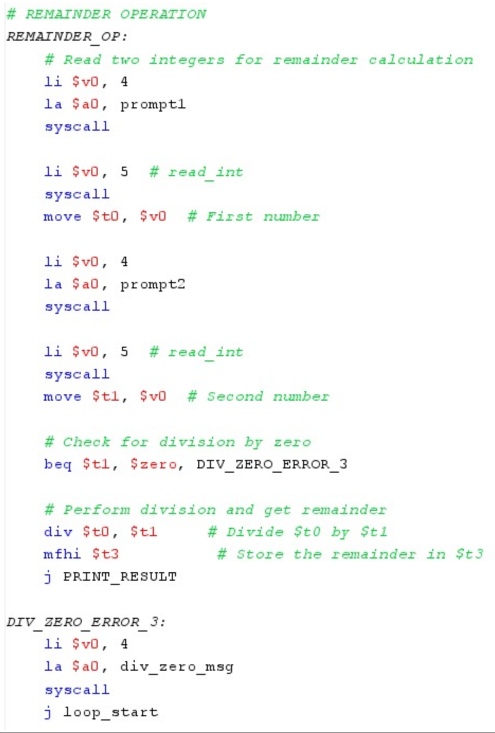
```
# EXPONENTIATION OPERATION
EXP_OP:
    # Read base and exponent
    li $v0, 4
    la $a0, prompt1
    syscall

    li $v0, 5   # read_int
    syscall
    move $t0, $v0   # Base

    li $v0, 4
    la $a0, prompt2
    syscall

    li $v0, 5   # read_int
    syscall
    move $t2, $v0   # Exponent

    # Calculate exponentiation
    li $t3, 1   # Result starts at 1
EXP_LOOP:
    beqz $t2, PRINT_RESULT
    mul $t3, $t3, $t0
    subi $t2, $t2, 1
    j EXP_LOOP
```

-The exponentiation operation:
Reads two integers, raises the second number to the power of the first and displays the result.

-The mean operation:
Calculates their arithmetic mean and displays the result.



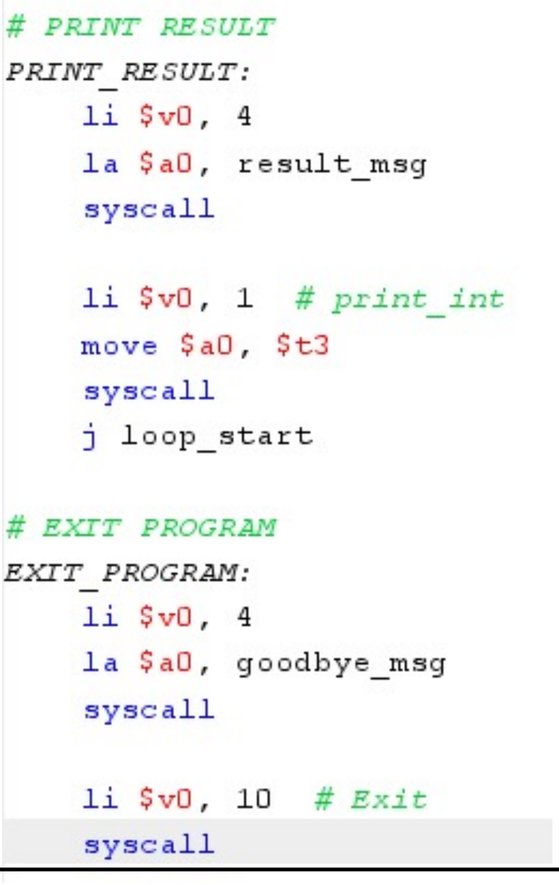
```
# MEAN OPERATION
MEAN_OP:
    # Read two integers for mean calculation
    li $v0, 4
    la $a0, prompt1
    syscall

    li $v0, 5   # read_int
    syscall
    move $t0, $v0   # First number

    li $v0, 4
    la $a0, prompt2
    syscall

    li $v0, 5   # read_int
    syscall
    move $t1, $v0   # Second number

    # Calculate the mean (sum / count)
    add $t2, $t0, $t1   # Add the numbers
    li $t3, 2            # Number of numbers (2 in this case)
    div $t2, $t3         # Divide the sum by the number of numbers
    mflo $t3              # Store result in $t3
    j PRINT_RESULT
```



```
# REMAINDER OPERATION
REMAINDER_OP:
    # Read two integers for remainder calculation
    li $v0, 4
    la $a0, prompt1
    syscall

    li $v0, 5  # read_int
    syscall
    move $t0, $v0  # First number

    li $v0, 4
    la $a0, prompt2
    syscall

    li $v0, 5  # read_int
    syscall
    move $t1, $v0  # Second number

    # Check for division by zero
    beq $t1, $zero, DIV_ZERO_ERROR_3

    # Perform division and get remainder
    div $t0, $t1      # Divide $t0 by $t1
    mfhi $t3           # Store the remainder in $t3
    j PRINT_RESULT

DIV_ZERO_ERROR_3:
    li $v0, 4
    la $a0, div_zero_msg
    syscall
    j loop_start
```

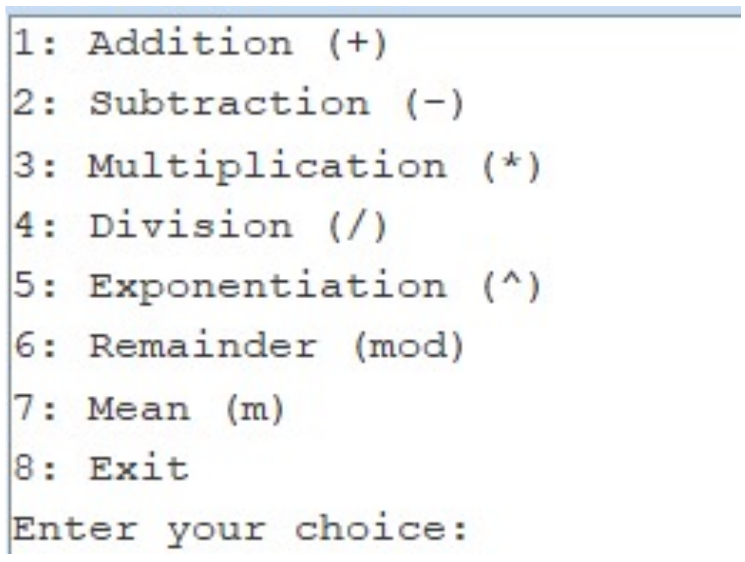
-The remainder operation:
Reads two integers, divides the first by the second and displays the remainder of the division.

-Displays the result to the user and terminates the program.



```
# PRINT RESULT
PRINT_RESULT:
    li $v0, 4
    la $a0, result_msg
    syscall

    li $v0, 1  # print_int
    move $a0, $t3
    syscall
    j loop_start

# EXIT PROGRAM
EXIT_PROGRAM:
    li $v0, 4
    la $a0, goodbye_msg
    syscall

    li $v0, 10  # Exit
    syscall
```

-The output:



```
1: Addition (+)
2: Subtraction (-)
3: Multiplication (*)
4: Division (/)
5: Exponentiation (^)
6: Remainder (mod)
7: Mean (m)
8: Exit
Enter your choice:
```

Thank you!