

Programming Assignment, Pgm1

1. Problem Statement:

The first programming assignment, Pgm1 for the course Computer Science 2 (CS5330.501) is a MIPS Assembly Language program that is intended to read two integers (A and B) from the user and perform several arithmetic operations with them. The arithmetic operations include the sum of the integers (**A+B**), differences of the integers (**A-B** and **B-A**), a product of the integers (**A*B**), and at last the quotient & the remainder from dividing the integers (**A/B** and **B/A**). The result of these arithmetic operations must be displayed back into the console for the user.

2. Approach:

The MIPS code for Pgm1 can be divided into two sections. The first section of the program could accept two integers (A and B) from the user and store the values in registers for future computations. The second section of the program could include the procedures to do the required arithmetic operations and display the results back to the user. Since the problem statement involves two division operations (**A/B** and **B/A**), the program could run into runtime error if the divisor is zero (as anything divided by zero is not defined), therefore the program must include a conditional statement which would check for the value of divisor and proceed with the computation accordingly.

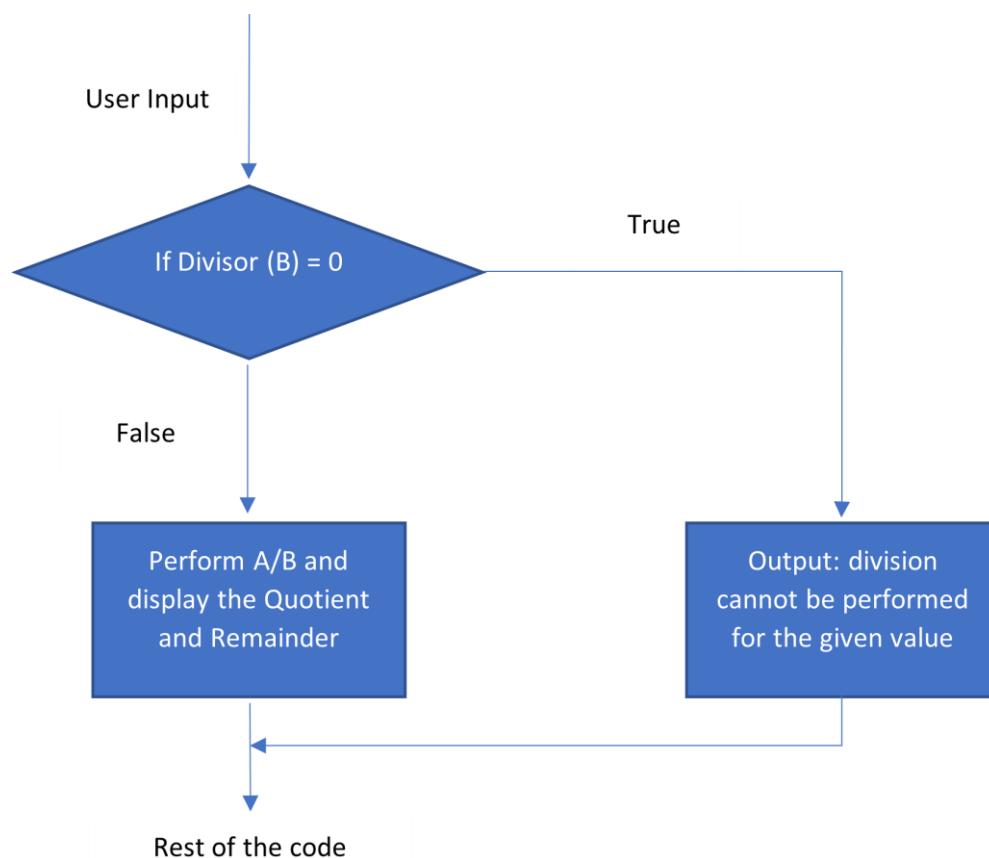


Figure 1 Flow of conditional statement for division operation

Figure 1 shows the flow of conditional statement for the division operation that must be followed to avoid runtime error. For the operation A/B , if the value of B (divisor) is zero, the flow branches out to true and displays a message. If the value of B is not zero, the flow branches out to false and performs the division operation. The same process flow can be used for the operation B/A wherein A is the divisor, and the process checks for the value of A.

3. Solution:

The problem statement is to read two integers from the user and perform a set of arithmetic operations. Figure 2 shows the code being successfully assembled.

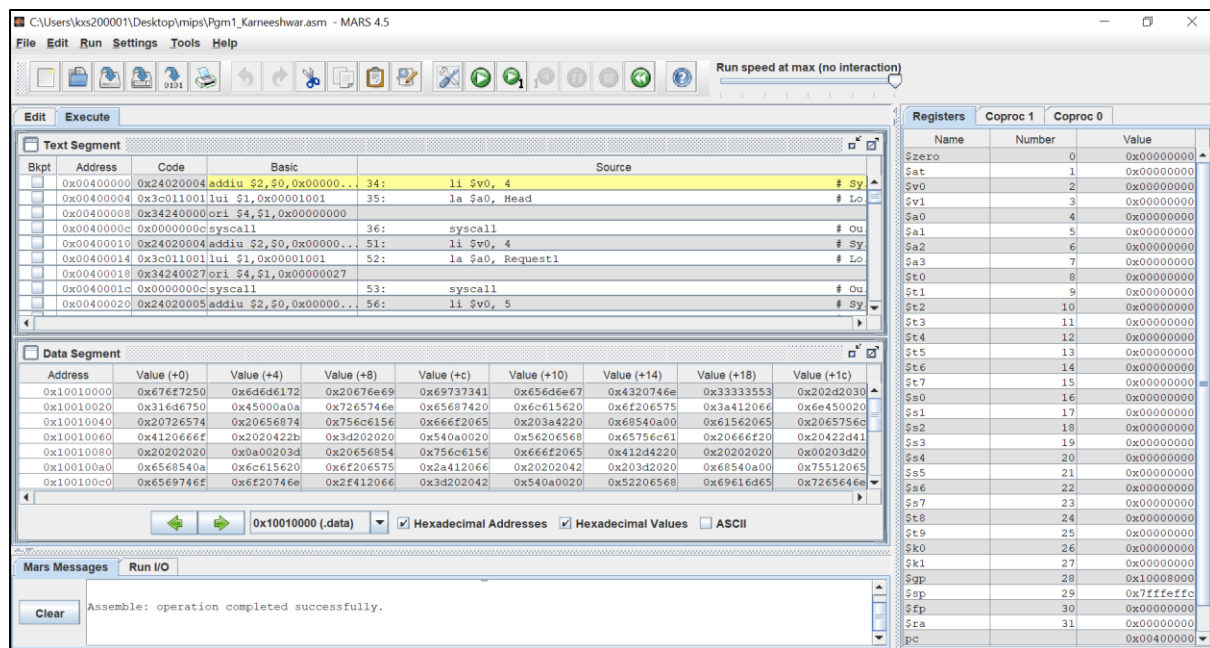


Figure 2 Screen capture of assembled code

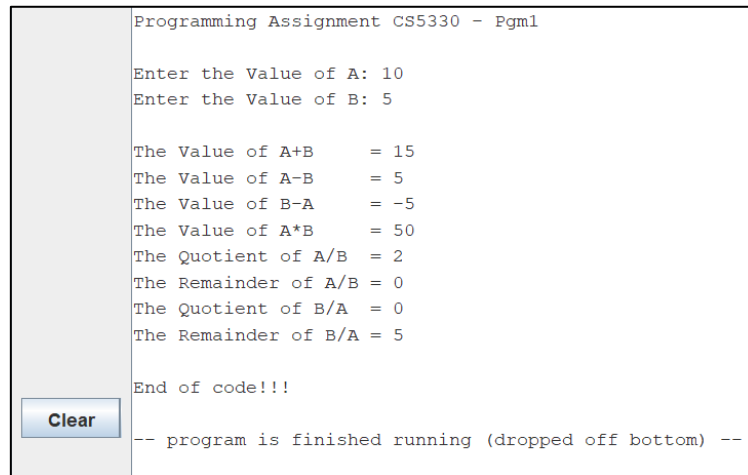
The first section of the MIPS code (*UserI*, refer to the appendix for code snippet 1) is created to accept the integers A and B. The user is prompted (Figure 3) by a message to enter the values of A and B individually using the standard system call, and these values are stored in registers so that the upcoming section of the program can use them for the arithmetic operations.



Figure 3 Prompt requesting the value of A from the user

Now that the values of A and B are available the program enters into the second section (*Calc*, refer to the appendix for code snippet 2) where all the arithmetic operations are done and displayed to the user in a sequence. Firstly, the sum of integers ($A+B$) is performed using the

“add” keyword and the result is displayed to the user. Followed by the difference operations (**A-B** and **B-A**) using “sub” keyword, product operation (**A*B**) using “mul” keyword, and division operations (**A/B** and **B/A**) using “div” keyword. The results of each arithmetic operation are stored in individual registers and displayed to the user right after each operation, a sample output with A = 10 and B = 5 can be seen in Figure 4.



```
Programming Assignment CS5330 - Pgm1

Enter the Value of A: 10
Enter the Value of B: 5

The Value of A+B      = 15
The Value of A-B      = 5
The Value of B-A      = -5
The Value of A*B      = 50
The Quotient of A/B   = 2
The Remainder of A/B  = 0
The Quotient of B/A   = 0
The Remainder of B/A  = 5

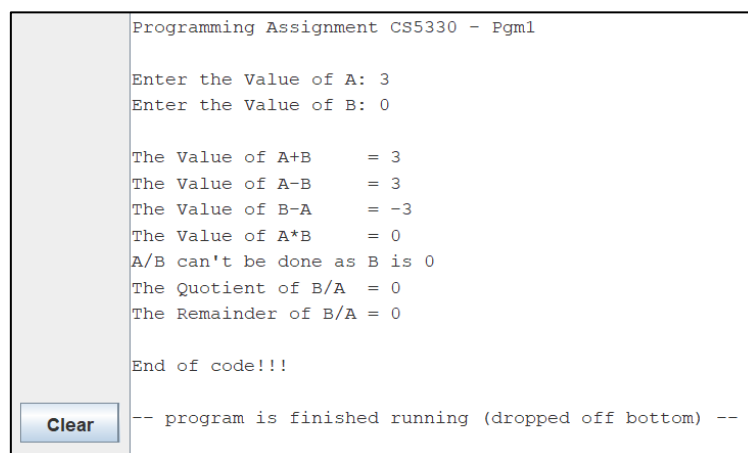
End of code!!!

-- program is finished running (dropped off bottom) --
```

Clear

Figure 4 Sample output for A = 10 and B = 5

The division operation is performed inside a block of a conditional statement. To avoid run time error, the divisors of each division operation (**A/B** and **B/A**) are checked for zero. If the value of the divisor is zero, the program is coded to jump to an else label (*else1 and else2*, refer to the appendix for code snippet 3) where the user is prompted by a message that the division operation cannot be performed. If the divisors happen to be non-zero numbers, the program continues to perform the division operation and displays the results to the user. Figure 5 shows a sample output for A = 3 and B = 0 (note that for the operation **A/B**, B being the divisor has a value of 0). Similarly, Figure 6 shows a sample output for A = 0 and B = 7.



```
Programming Assignment CS5330 - Pgm1

Enter the Value of A: 3
Enter the Value of B: 0

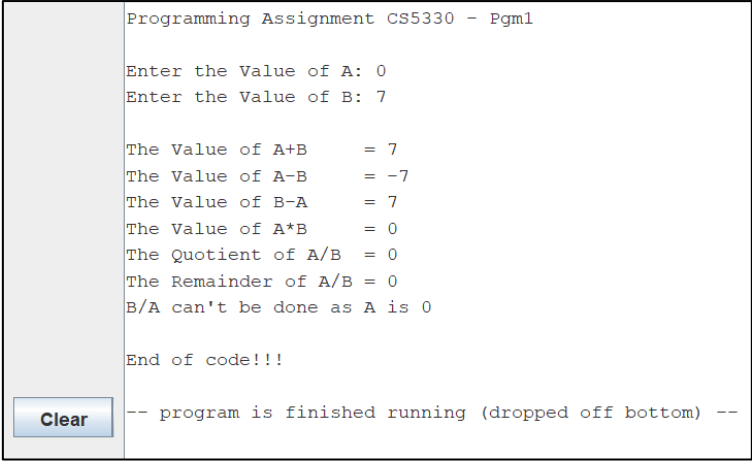
The Value of A+B      = 3
The Value of A-B      = 3
The Value of B-A      = -3
The Value of A*B      = 0
A/B can't be done as B is 0
The Quotient of B/A   = 0
The Remainder of B/A  = 0

End of code!!!

-- program is finished running (dropped off bottom) --
```

Clear

Figure 5 Sample Output for A = 3 and B = 0



```
Programming Assignment CS5330 - Pgm1

Enter the Value of A: 0
Enter the Value of B: 7

The Value of A+B      = 7
The Value of A-B      = -7
The Value of B-A      = 7
The Value of A*B      = 0
The Quotient of A/B   = 0
The Remainder of A/B  = 0
B/A can't be done as A is 0

End of code!!!

-- program is finished running (dropped off bottom) --
```

Clear

Figure 6 Sample Output for A = 0 and B = 7

4. Conclusion:

The given problem statement is analyzed, and the required outputs were delivered using the MIPS programming language. The code has been created to accept two integer values from the user and several arithmetic operations such as addition, subtraction, multiplication, and division were performed using the user's input. The results of these operations were displayed back to the user in a clear format.

Appendix

Snippet 1: First Section of MIPS code (*User1*) for getting the input from the user:

User1:

Request the user to enter the value of A

```
li $v0, 4          # System call for printing string Request1
la $a0, Request1   # Load address of Request1 prompt
syscall           # Outputs string Request1
```

Get the value of A

```
li $v0, 5          # System call for interger input from user
syscall           # Inputs the Value of A from the user and stores it in $v0
move $t0, $v0      # Storing the Value of A to $t0
```

Where, Request1 is a string with ascii value of "Enter the Value of A: "

Snippet 2: Second Section of MIPS code (*Calc*) for performing arithmetic operations:

Calc:

#A+B

```
li $v0, 4          # System call for printing string AplusB
la $a0, AplusB     # Load address of AplusB String
syscall           # Outputs string AplusB
add $t2, $t0, $t1   # $t2 = $t0 + $t1 ( => A+B) sum of A and B
li $v0, 1          # System call for printing integer A+B
move $a0, $t2       # Storing the Value of A+B to $a0 for printing
syscall           # Outputs the value of AplusB
```

Where, AplusB is a string with ascii value of "The Value of A+B = "

Snippet 3: MIPS code to display a message to user when divisor is zero:

#A/B

beq \$t1, \$zero, else1 # if B = 0, branch to else1

.

.

. # block of code if B is not 0

.

.

else1: li \$v0, 4 # System call for printing string noAbyB

la \$a0, noAbyB # Load address of noAbyB String

syscall # Outputs string noAbyB

#B/A

beq \$t0, \$zero, else2 # if A = 0, branch to else2

.

.

. # block of code if A is not 0

.

.

else2: li \$v0, 4 # System call for printing string noBbyA

la \$a0, noBbyA # Load address of noBbyA String

syscall # Outputs string noBbyA

Where,

noAbyB is a string with ascii value of "A/B can't be done as B is 0"

noBbyA is a string with ascii value of "B/A can't be done as A is 0"