CLEVELAND STATE UNIVERSITY

Class: *CIS580(Computer Architecture)*
Project Title: *MIPS programming (project#02)*
Name: *Daniel Izadnegahdar / 2420596*
Submission LoginID#: *daizadne*

1. Summary of code:

1.1. This program is written in MIPS to calculate a sequence of Fibonacci numbers. It first prints general information about the code such as author, title, and description. It then prompts the user for a name and an integer. The integer will continue looping until the user provides a number between 1 and 47 (inclusive). This integer is used to count the number of Fibonacci numbers to print. Finally, the Fibonacci sequence and a farewell message is printed, and the program ends. The extra credit function is also included in the program, it runs after the Fibonacci program is completed.

1.1.1. Program summary:

1.1.1.1. Data block: The program begins by defining the main data variables. This includes the strings of the program, which are mainly stored inside the .asciiz type, except for the input string which is stored in a .space type with a 64 byte limitation.

1.1.1.2. Text block: Next, the .text block is called, where it is divided into the *main*, *whileFormat*, *checkLowerBound*, *repeat*, *startFibonacci*, *whileFibonacci*, *firstFibonacci*, *secondFibonacci*, *thirdFibonacci*, *newline*, and *exit* functions.

1.1.1.3. whileFormat: The *whileFormat* block is used to run continuously until the user integer is inside the allowable range. If the user integer is lower than 1 or higher than 47, the block will loop. It does this by using a blt instruction to check the upper bound. If the upper bound is met, then it will run the *checkLowerBound* block. If both of these are met, then the loop will exit, otherwise it will continue to loop through the *repeat* block.

1.1.1.4. whileFibonacci: Once the Fibonacci block is ready to run, it starts by printing the 1st 3 Fibonacci numbers through the first, second, and third Fibonacci function blocks. Afterwards, it begins to calculate the sequence by adding the last 2 numbers. To achieve this, it swaps the registers $s1, $s2, and $s3 so that it can continuously calculate Fibonacci numbers without running out of registers. Afterwards, it prints 4 spaces after the Fibonacci number. Finally, the whileFibonacci sequence prints a new line with "\n", only if the modulus of the counter with 5, is equal to 0. This is to maintain the format requested per the assignment, where only 5 numbers must show per row.

1.1.1.5. exit: The program then sends a farewell message to the user, it prints the user's name, and proceeds with the extra credit code. The program then exits on its own using the #10 MIPS command.

1.1.2. General data map:

- ***1.1.2.1.*** **$s0** = user max
- ***1.1.2.2.*** **$s1** = 1st value
- ***1.1.2.3.*** **$s2** = 2nd value
- ***1.1.2.4.*** **$s3** = sum
- ***1.1.2.5.*** **$s5** = remainder
- ***1.1.2.6.*** **$s7** = counter

---

2. Run instructions:

2.1. Keyboard method:

- ***2.1.1. Open MARS:*** Run MARS and open the provided branch.asm file.
- ***2.1.2. Keyboard:*** Press F3 to assemble and F5 to run.
- ***2.1.3. GUI (Alternative):*** Press F3 to assemble and F5 to run.

---

3. Sample test run:

```
************************************************
Author: Daniel Izadnegahdar
Title: Fibonacci Numbers
Description: MIPS program that takes a user integer to calculate n many fibonacci numbers.
************************************************
Fibonacci Numbers by, Daniel Izadnegahdar

What is your name? John Smith
Hi, John Smith

How many Fibonacci numbers should I display?
Enter an integer in the range [1..47]: 0
That number was out of range, try again.

How many Fibonacci numbers should I display?
Enter an integer in the range [1..47]: 48
That number was out of range, try again.

How many Fibonacci numbers should I display?
Enter an integer in the range [1..47]: 47

0    1    1    2    3
5    8    13    21    34
55    89    144    233    377
610    987    1597    2584    4181
6765    10946    17711    28657    46368
75025    121393    196418    317811    514229
832040    1346269    2178309    3524578    5702887
9227465    14930352    24157817    39088169    63245986
102334155    165580141    267914296    433494437    701408733
1134903170    1836311903

Goodbye, John Smith
************************************************
```

4. Experiences in debugging and testing:

4.1. Development:

4.1.1. The development began with research on *MIPS programming* using the *MARS IDE*. This required reviewing the documentation and the provided class material. One of the key discoveries of assembly programming was how to program AND & IF statements. As opposed to high-level programming, in assembly, much of these statements are achieved through nested code blocks using the branch functions.

4.2. Debugging& Testing:

4.2.1. To debug, the code errors were diagnosed by reviewing the output notifications. The run, step, and stop tools of MARS were frequently used for troubleshooting.