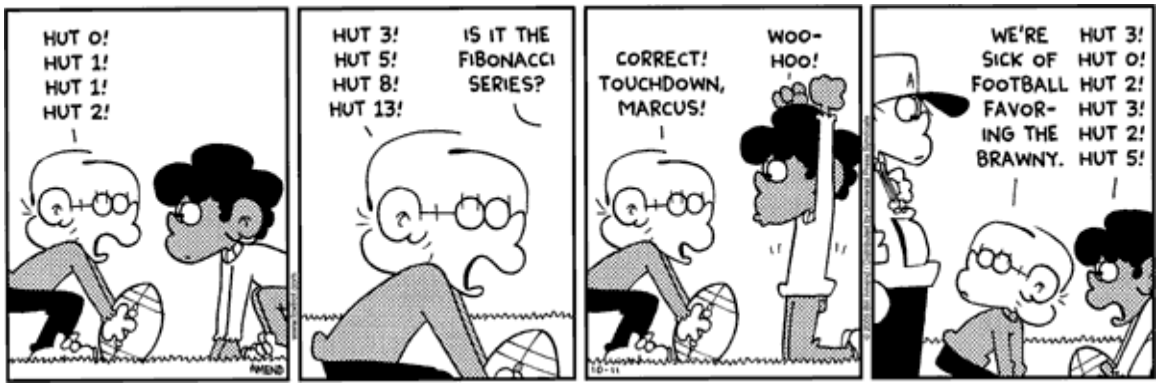
CS 218 – Assignment #4

Purpose: Become familiar with the MIPS Instruction Set, and the MIPS procedure calling convention, and basic recursion.
Points: 75

Assignment:
Write an assembly language program to compute the Fibonacci and Perrin sequences.



The provided main program will call the following functions:

- Value returning function, ***fibonacci()***, that recursively computes the Fibonacci number. Must be recursive for credit. The Fibonacci Sequence is defined as follows:

$$fibonacci(n) = \begin{cases} n & \text{if } n=0 \text{ or } n=1 \\ fibonacci(n-2) + fibonacci(n-1) & \text{if } n \geq 2 \end{cases}$$

- Value returning function, ***perrin()***, that recursively computes the Perrin number. Must be recursive for credit. The Perrin sequence is defined as follows:

$$perrin(n) = \begin{cases} 3 & \text{if } n=0 \\ 0 & \text{if } n=1 \\ 2 & \text{if } n=2 \\ perrin(n-2) + perrin(n-3) & \text{if } n>2 \end{cases}$$

- Value returning function, ***getNumber()***, that prompts for and reads the N value from the user. The program must verify that N is between 3 and 45 (inclusive). Based on the N value, a display a message.
  - If N <= 25 display the message "**This should be quick**".
  - If N >25, display "**This is going to take a while (n>25)**".
  - If N >30, display "**This is going to take a long time (n>30)**".
  - If N >35, display "**This is going to take a very long time (> 30 minutes)**".
  - If N < 3 or N > 45, display "**Error, value of range. Please re-enter.**" and re-prompt.

  All message and error strings are predefined. *Note*, the QtSpim system service to read an integer ($v0=5) does not provide error handling for non-digit input. As such, that is not required on this assignment.

- Void function, ***prtLine()***, that displays the information in the specified format, including right justifying the numbers. Note the ***N*** values will have up to 2 digits (max) and the Fibonacci and numbers will have up to 10 digits (max) and the Perrin numbers will have up to 7 digits (max). The formatting must accommodate these sizes and match the example output (below).

  The line should look like:

  ```
  num =  0   fibonacci =          0   perrin = 3
  ```

  With a general format as follows:

  ```
  numB=BnnBBBfibonacciB=Bfffffffffff BBBperrinB=Bppppppp
  ```

  Where
  - `B` = blank space
  - `f` = actual Fibonacci number (123…)
  - `p` = actual Perrin number (123...)

  Note;
  - The number value will always be 1-2 digits.
  - The Fibonacci value will always b 1-10 digits.
  - The Perrin value will always be 1-7 digits.

  This function should call the ***prtBlnks()*** function for print the appropriate number of blanks preceding the number (based on the size of the number).

- Void function, ***prtBlnks()***, that will print an appropriate number of blanks based on the size of the number. This is how the right justification is accomplished. The function should accept the number and the field size in order to determine the correct number of blanks to print. Once the appropriate number of blanks to print is determined, a single blank can be printed in a loop.

  Note, this should be a short function, overly complex solutions will be marked off.

Do not change the data types of the provided data. You may define additional variables as required.

Submission:

- All source files must assemble and execute with QtSpim/SPIM MIPS simulator.
- Submit source file
  - Submit a copy of the program source file via the on-line submission
- Once you submit, the system will score the project and provide feedback.
  - If you do not get full score, you can (and should) correct and resubmit.
  - You can re-submit an unlimited number of times before the due date/time (at a maximum rate of 5 submissions per hour).
- Late submissions will be accepted for a period of 24 hours after the due date/time for any given lab. Late submissions will be subject to a ~2% reduction in points per an hour late. If you submit 1 minute - 1 hour late -2%, 1-2 hours late -4%, … , 23-24 hours late -50%. This means after 24 hours late submissions will receive an automatic 0.

Program Header Block

All source files must include your name, section number, assignment, NSHE number, and program description. The required format is as follows:

```
#  Name: <your name>
#  NSHE ID: <your id>
#  Section: <section>
#  Assignment: <assignment number>
#  Description: <short description of program goes here>
```

Failure to include your name in this format will result in a reduction of points.

Scoring Rubric

Scoring will include functionality, code quality, and documentation. Below is a summary of the scoring rubric for this assignment.

Criteria	Weight	Summary
Assemble	-	Failure to assemble will result in a score of 0.
Program Header	3%	Must include header block in the required format (see above).
General Comments	7%	Must include an appropriate level of program documentation.
Program Functionality (and on-time)	90%	Program must meet the functional requirements as outlined in the assignment. Must be submitted on time for full score.

Example Output:

Note: For formatting, the ***N*** value has a max of 2 digits, Fibonacci numbers have a max of 10 digits, and Perrin numbers have a max of 7 digits.

An example, an execution might look like:

```
MIPS Assignment #4
Fibonacci and Perrin Numbers Program


Enter N (3-45, 0 to terminate): 2

Error, value of range.  Please re-enter.



Enter N (3-45, 0 to terminate): 46

Error, value of range.  Please re-enter.



Enter N (3-45, 0 to terminate): 19

This should be quick.

num =  0   fibonacci =         0   perrin =       3
num =  1   fibonacci =         1   perrin =       0
num =  2   fibonacci =         1   perrin =       2
num =  3   fibonacci =         2   perrin =       3
num =  4   fibonacci =         3   perrin =       2
num =  5   fibonacci =         5   perrin =       5
num =  6   fibonacci =         8   perrin =       5
num =  7   fibonacci =        13   perrin =       7
num =  8   fibonacci =        21   perrin =      10
num =  9   fibonacci =        34   perrin =      12
num = 10   fibonacci =        55   perrin =      17
num = 11   fibonacci =        89   perrin =      22
num = 12   fibonacci =       144   perrin =      29
num = 13   fibonacci =       233   perrin =      39
num = 14   fibonacci =       377   perrin =      51
num = 15   fibonacci =       610   perrin =      68
num = 16   fibonacci =       987   perrin =      90
num = 17   fibonacci =      1597   perrin =     119
num = 18   fibonacci =      2584   perrin =     158
num = 19   fibonacci =      4181   perrin =     209



Enter N (3-45, 0 to terminate): 0


Game Over, thank you for playing.
```

And additional output might look as follows:

```
MIPS Assignment #4
Fibonacci and Perrin Numbers Program


Enter N (3-45, 0 to terminate): 1

Error, value of range.  Please re-enter.



Enter N (3-45, 0 to terminate): 26

This is going to take a while (n>20).

num =  0   fibonacci =        0   perrin =      3
num =  1   fibonacci =        1   perrin =      0
num =  2   fibonacci =        1   perrin =      2
num =  3   fibonacci =        2   perrin =      3
num =  4   fibonacci =        3   perrin =      2
num =  5   fibonacci =        5   perrin =      5
num =  6   fibonacci =        8   perrin =      5
num =  7   fibonacci =       13   perrin =      7
num =  8   fibonacci =       21   perrin =     10
num =  9   fibonacci =       34   perrin =     12
num = 10   fibonacci =       55   perrin =     17
num = 11   fibonacci =       89   perrin =     22
num = 12   fibonacci =      144   perrin =     29
num = 13   fibonacci =      233   perrin =     39
num = 14   fibonacci =      377   perrin =     51
num = 15   fibonacci =      610   perrin =     68
num = 16   fibonacci =      987   perrin =     90
num = 17   fibonacci =     1597   perrin =    119
num = 18   fibonacci =     2584   perrin =    158
num = 19   fibonacci =     4181   perrin =    209
num = 20   fibonacci =     6765   perrin =    277
num = 21   fibonacci =    10946   perrin =    367
num = 22   fibonacci =    17711   perrin =    486
num = 23   fibonacci =    28657   perrin =    644
num = 24   fibonacci =    46368   perrin =    853
num = 25   fibonacci =    75025   perrin =   1130
num = 26   fibonacci =   121393   perrin =   1497


Enter N (3-45, 0 to terminate): 0


Game Over, thank you for playing.
```

And a final output might look as follows:

```
MIPS Assignment #4
Fibonacci and Perrin Numbers Program


Enter N (3-45, 0 to terminate): 31

This is going to take a long time (n>30).

num =  0   fibonacci =        0   perrin =      3
num =  1   fibonacci =        1   perrin =      0
num =  2   fibonacci =        1   perrin =      2
num =  3   fibonacci =        2   perrin =      3
num =  4   fibonacci =        3   perrin =      2
num =  5   fibonacci =        5   perrin =      5
num =  6   fibonacci =        8   perrin =      5
num =  7   fibonacci =       13   perrin =      7
num =  8   fibonacci =       21   perrin =     10
num =  9   fibonacci =       34   perrin =     12
num = 10   fibonacci =       55   perrin =     17
num = 11   fibonacci =       89   perrin =     22
num = 12   fibonacci =      144   perrin =     29
num = 13   fibonacci =      233   perrin =     39
num = 14   fibonacci =      377   perrin =     51
num = 15   fibonacci =      610   perrin =     68
num = 16   fibonacci =      987   perrin =     90
num = 17   fibonacci =     1597   perrin =    119
num = 18   fibonacci =     2584   perrin =    158
num = 19   fibonacci =     4181   perrin =    209
num = 20   fibonacci =     6765   perrin =    277
num = 21   fibonacci =    10946   perrin =    367
num = 22   fibonacci =    17711   perrin =    486
num = 23   fibonacci =    28657   perrin =    644
num = 24   fibonacci =    46368   perrin =    853
num = 25   fibonacci =    75025   perrin =   1130
num = 26   fibonacci =   121393   perrin =   1497
num = 27   fibonacci =   196418   perrin =   1983
num = 28   fibonacci =   317811   perrin =   2627
num = 29   fibonacci =   514229   perrin =   3480
num = 30   fibonacci =   832040   perrin =   4610
num = 31   fibonacci =  1346269   perrin =   6107


Enter N (3-45, 0 to terminate): 0


Game Over, thank you for playing.
```